

SIMATIC

FM 357-2 Multi-Axis Module for Servo and Stepper Drives

Manual

Preface, Contents	
Product Summary	1
Fundamental Principles of Motion Control	2
Installation and Removal	3
Wiring	4
Parameterization	5
Programming of Standard Function Blocks	6
Start-Up	7
Human Machine Interface	8
Description of Functions	9
NC Programming	10
Application Example	11
Troubleshooting	12
Handlings Transformation	13
Appendices	
Technical Specifications	A
List of Abbreviations	B
Index	

Safety Guidelines

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



Danger

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



Warning

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



Caution

with a safety alert symbol indicates that minor personal injury can result if proper precautions are not taken.

Caution

without a safety alert symbol indicates that property damage can result if proper precautions are not taken.

Notice

indicates that an unintended result or situation can occur if the corresponding notice is not taken into account.

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

Qualified personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by qualified personnel. Within the context of the safety notices in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Prescribes Usage

Note the following:



Warning

The device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

Trademarks

All names identified by ® are registered trademarks of SIEMENS AG.

The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

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

Preface

Purpose of this documentation

This Manual contains all information about the FM 357-2 module, i.e.

- Hardware and functions
- Parameter definition
- Man-machine interface
- Technology blocks
- NC programming
- Safe setup

Information blocks of the Manual

The following information blocks describe the purpose and uses of the Manual.

- Product overview of the module (Chapter 1)
This section explains the purpose and possible applications of the module. It provides introductory information about the FM 357-2 and its functions.
- Fundamentals of motion control (Chapter 2)
This section contains an elementary description of the principles of controlling the motion of single axes and axis groupings and includes an explanation of terminology.
- Installation and removal (Chapter 3)
Explains the installation and removal of the FM 357-2.
- Wiring (Chapter 4)
Describes the connection and wiring of drives, encoders and digital input/output modules.
- Parameterization (Chapter 5)
Describes the parameterization and functions of "Parameterize FM 357-2".
- Programming of standard function blocks (Chapter 6)
Describes how technology functions can be programmed with STEP 7.
- Starting up (Chapter 7)
Describes startup procedures for the FM 357-2.
- Human-machine interface (Chapter 8)
Describes the available options for controlling and monitoring the FM 357-2 and which data/signals can be controlled and monitored.

- Reference information and appendices for looking up factual information (module functions, NC programming guide, interface signals, parameter lists, error treatment, technical data, standard HMI, user data blocks).
- List of abbreviations and index for looking up information.

What you need to know to understand this Manual

This Manual describes the hardware and functionality of the FM 357-2 module.

To set up, program and start up a SIMATIC S7-300 with the FM 357-2, you will need a knowledge of:

- The SIMATIC S7-300
 - Manual *S7-300 Assembly, CPU data*
 - Manual *S7-300 M7-300, Module data*
 - List of operations
- Your programming device (PG)
- How to program with STEP 7
- Configuring the interface of an operator panel (e.g. OP 17)

FM 357-2 users

The information in this Manual is structured and represented in accordance with the field of application of the FM 357-2 and the relevant activity of the user.

The subject matter is divided into the following areas:

- Installation and wiring
- Parameterizing and programming
- Troubleshooting and diagnostics
- Human-machine interface

Standards and approvals

Our products meet the requirements of EU Directive 89/336/EEC Electromagnetic Compatibility and the harmonized European Standards (EN) listed there.

You can find detailed information on approvals and standards in the appendix.

The current Declaration of Conformity is on the Internet at

<http://support.automation.siemens.com/WW/view/en/15257461>

Recycling and disposal

The SIMATIC S7-300 is an environmentally compatible product! Its features are:

- The plastic material of the housing features halogen-free flame protection despite high fire resistance.
- Labelling by means of laser (i.e. no labels)
- Marking of plastic materials according to DIN 54840
- Less use of materials due to smaller size construction, fewer components through the integration in ASIC.

The SIMATIC S7-300 can be recycled thanks to its construction with low contaminating materials.

For recycling in an environmentally compatible manner and for the disposal of your old SIMATIC in line with present state of technology, please contact your appropriate Siemens contact partner:

<http://www.automation.siemens.com/partner>

Tips for the user

The following table gives an overview of the firmware products, the standard function blocks and the parameterizing tools for the relevant hardware.

Compatibility list

Hard-ware FM ...	Firm-ware	Parameterization package				
		Param. tool	Standard-FB/FCs	Surface for OP 17/27	Manual (Order No.)	First steps (Order No.)
357	V1.x	<ul style="list-style-type: none"> • V1.3, V2.3 • V3.x 	V1.3	V1.3	<ul style="list-style-type: none"> • GWE-570093101798 04.98 Edition • GWE-570093102081 03.2000 Edition 	–
357	V2.x	<ul style="list-style-type: none"> • V2.3 • V3.x 	V2.3	V1.3	<ul style="list-style-type: none"> • GWE-571093101798 04.98 Edition • GWE-570093102081 03.2000 Edition 	GWE-570093101872
357-2	V3.x	V3.x	V3.x	V3.x	GWE-570093102081 03.2000 Edition	03.2000 Edition
357-2	V3.x	V3.x	V3.x	V3.x	6ES7 357-4AH00-8BG0 01.01 Edition	03.2000Edition
357-2	V4.x	V4.x	V4.x	V3.x	6ES7 357-4AH00-8BG0 10.01 Edition	03.2000Edition
357-2	V5.x	V5.x	V5.x	V5.x	6ES7 357-4AH00-8BG0 05.02 Edition	03.2000 Edition
357-2	V5.3	V5.3	V5.2	V5.1	6ES7 357-4AH00-8AG0 01.03 Edition	03.2000 Edition
357-2	V5.3	V5.3, incl. SP1	V5.2, incl. SP1	V5.1, incl. HF1	6ES7 357-4AH00-8AG0 11/2008 Edition	11/2008Edition
357-2	V5.3	V5.3, incl. SP2	V5.2, incl. SP2	V5.1, incl. HF2	6ES7 357-4AH00-8AG0 10/2012 Edition	10/2012 Edition

Contact partners

Should you encounter any problems or questions when dealing with this Manual, then you are kindly asked to contact the branch specified on the response form at the end of this Manual.

Hotline

If you have any technical questions, please contact our hotline:

Europe / Africa (Nuremberg)

Phone: +49 (180) 50 50 222

Fax: +49 (180) 50 50 223

Internet: <http://support.automation.siemens.com>

Americas (Johnson City)

Phone: +1 423 461-2522

Fax: +1 423 461-2289

E-Mail: techsupport.sea@siemens.com

Asia / Pacific (Singapore)

Phone: +65 740-7000

Fax: +65 740-7001

E-Mail: adsupport.asia@siemens.com

Further support

To facilitate you the entry into the way of working with a SIMATIC S7 automation system, training courses are offered.

Please contact either your regional training center or the central training center in D-90027 Nuremberg, Tel. +49 (0911) 895-3202.

Information about training courses on offer can be found at:

www.sitrain.com

Siemens Internet address

The latest information about SIMATIC products can be found on the Internet at:

<http://www.siemens.com/simatic>



Contents

1	Product Summary	1-1
1.1	The FM 357-2 in the S7-300 programmable controller system	1-3
1.2	Module description	1-9
1.3	Overview of module functions	1-12
2	Fundamental Principles of Motion Control	2-1
3	Installation and Removal	3-1
3.1	Installing the FM 357-2	3-3
3.2	Installing firmware/firmware update	3-4
3.3	Removing and replacing the FM 357-2	3-6
4	Wiring	4-1
4.1	Wiring an FM 357-2	4-3
4.2	Connecting the power supply	4-7
4.3	PROFIBUS DP drive interface	4-11
4.4	Description of the drive interface for analog and stepper drives	4-12
4.5	Connecting the drive units	4-18
4.6	Description of the measuring system interface	4-22
4.7	Connecting the encoders	4-27
4.8	Description of the I/O interface	4-29
4.9	Wiring up the front connector	4-33
4.10	Inserting and replacing the backup battery	4-37
5	Parameterization	5-1
5.1	Installation of "Parameterize FM 357-2"	5-3
5.2	Configuring	5-5
5.2.1	Sample project with PROFIBUS-DP drive	5-6
5.2.2	Sample project PROFIBUS-DP drive and PG/PC connected to the MPI interface of the CPU	5-12
5.2.3	Sample project PROFIBUS-DP drive and PG/PC connected to the DP interface of the CPU	5-15
5.2.4	Sample project "Coupling the PG/PC via Ethernet"	5-18
5.3	Entry to "Parameterizing the FM 357-2"	5-19
5.4	Adaptation to firmware	5-20
5.5	Parameter data	5-21
5.5.1	Machine data (parameters)	5-23

5.5.2	R parameters	5-37
5.5.3	Zero offset	5-37
5.5.4	Tool offset values	5-38
5.5.5	NC programs	5-38
5.6	Settings of parameterization interface	5-39
6	Programming of Standard Function Blocks	6-1
6.1	Programming fundamentals	6-3
6.1.1	FM357_2L library	6-3
6.1.2	Interface, user data blocks (user DB)	6-4
6.1.3	Standard function blocks, overview	6-5
6.1.4	CPU/FM 357-2 communication	6-7
6.1.5	Information about symbolic programming	6-8
6.1.6	Structure of a user program	6-9
6.1.7	Procedure for writing the user program	6-14
6.2	Startup with the parameterization tool "Parameterize FM 357-2"	6-15
6.3	Description of the standard function blocks	6-16
6.3.1	FC 1: RUN_UP – Startup/initialization	6-17
6.3.2	FC 5: BF_DIAG – Diagnostic alarm and FM restart	6-19
6.3.3	FC 22: BFCT – Basic functions and operating modes	6-22
6.3.4	FB 2: FM_GET – Read FM variable	6-24
6.3.5	FB 3: FM_PUT – Write FM variable	6-32
6.3.6	FB 4: FM_PI – General services	6-38
6.3.7	DB 121: VAR_ADDR – Extract from FM variable list	6-45
6.4	Distributed installations	6-47
6.5	User handling, function sequences	6-48
6.5.1	Axis control from the CPU	6-49
6.5.2	Auxiliary functions	6-57
6.5.3	ASUB – Start asynchronous subprograms	6-58
6.6	User data blocks (user DB) of the interface to the FM 357-2	6-59
6.6.1	User data block "FMx"	6-61
6.6.2	User data block "AXy"	6-72
6.6.3	Description of signals	6-75
6.7	Timing diagrams, communication	6-99
6.8	Application examples	6-100
6.8.1	Example 1, axes moving in "Reference point approach" submode	6-101
6.8.2	Example 2, axes moving in "Jog" mode	6-101
6.8.3	Example 3, starting an axis with axis control from CPU	6-102
6.8.4	Example 4, "Automatic mode" with program selection (with FB 4)	6-103
6.8.5	Example 5, reading and writing FM variables (with FB 2 and FB 3)	6-104
6.8.6	Example 6, reading and writing R parameters (with FB 2 and FB 3)	6-106
6.8.7	Example 7, error diagnosis	6-108
6.9	Technical data, runtime specifications	6-110

7	Start-Up	7-1
7.1	Installation and wiring	7-2
7.2	Powering up the FM 357-2	7-3
7.3	Procedure for parameterization	7-5
7.4	Testing and optimization	7-7
8	Human Machine Interface	8-1
8.1	HMI sample interface for the OP 17	8-3
8.2	Troubleshooting on the OP 17 (configuration example)	8-8
8.3	HMI sample interface for SIMATIC HMI devices	8-10
9	Description of Functions	9-1
9.1	Configuration	9-3
9.1.1	System rates	9-4
9.1.2	Axis configuration	9-6
9.1.3	Channel configuration	9-14
9.1.4	Parameters for configuration	9-15
9.2	Encoders	9-18
9.2.1	Incremental encoders	9-20
9.2.2	Absolute encoders (SSI)	9-22
9.2.3	Stepper motor	9-26
9.3	Position control	9-27
9.4	DSC (Dynamic Servo Control)	9-37
9.5	Velocities and acceleration rates	9-39
9.6	Monitoring	9-45
9.6.1	Monitoring of movements	9-45
9.6.2	Encoder monitoring	9-51
9.6.3	Hardware and software limit switches	9-54
9.7	Referencing and alignment	9-56
9.7.1	Referencing with incremental encoders	9-58
9.7.2	Referencing with stepper motors without encoders	9-64
9.7.3	Alignment with absolute encoders	9-65
9.8	Event-controlled program calls	9-68
9.9	Output of M, T and H functions	9-73
9.10	Inputs/outputs	9-76
9.10.1	Digital on-board I/Os	9-76
9.10.2	Digital I/Os on local P bus	9-79
9.10.3	Analog I/Os on the local P bus	9-82
9.11	Limit switching signals (software cams)	9-85
9.11.1	Parameterization	9-85
9.11.2	Activation and output of the limit switching signals	9-90
9.11.3	Limit switching signals with separated output	9-92
9.11.4	Limit switching signals with linked output	9-95
9.11.5	Hot-spot measurement	9-98
9.12	Operating modes	9-102

9.13	NC program execution	9-105
9.14	Asynchronous subprogram (ASUB)	9-107
9.15	Protection zones	9-109
9.16	Motion coupling	9-114
9.16.1	Coupled motion	9-114
9.16.2	Gantry	9-117
9.16.3	Master value coupling	9-124
9.16.4	Electronic gear	9-131
9.16.5	Tangential control	9-137
9.16.6	Overlaid motion in synchronized actions	9-140
9.17	Measurement	9-142
9.18	Travel to fixed stop	9-146
9.18.1	Parameter definition	9-147
9.18.2	Analog drive	9-149
9.18.3	Functional sequence	9-150
9.18.4	Further information	9-154
9.19	EMERGENCY STOP	9-155
9.20	Controlling	9-157
9.21	Axis replacement	9-163
10	NC Programming	10-1
10.1	Basic principles of NC programming	10-3
10.1.1	Program structure and program name	10-3
10.1.2	Statements	10-4
10.1.3	Block structure	10-6
10.1.4	Character set of control	10-9
10.2	Coordinate systems and dimensions	10-10
10.2.1	Coordinate systems	10-10
10.2.2	Axis types	10-11
10.2.3	Absolute dimensions and incremental dimensions (G90, G91, AC, IC)	10-13
10.2.4	Absolute dimensions for rotary axes (DC, ACP, ACN)	10-15
10.2.5	Polar coordinates (G110, G111, G112, RP, AP)	10-17
10.2.6	Meters and inches (G70, G71)	10-20
10.2.7	Plane selection (G17, G18, G19)	10-21
10.3	Zero offsets (frames)	10-22
10.3.1	Settable zero offsets (G54, G55, G56, G57, G500, G53)	10-22
10.3.2	Programmable zero offsets (TRANS, ATRANS, ROT, AROT, RPL, MIRROR, AMIRROR)	10-25
10.4	Set actual value (PRESETON)	10-30
10.5	Programming axis movements	10-31
10.5.1	Programming feeds (F, FA, FL)	10-32
10.5.2	Feed interpolation (FNORM, FLIN, FCUB)	10-33
10.5.3	Path group (FGROUP)	10-36
10.5.4	Linear interpolation with rapid traverse (G0)	10-37
10.5.5	Positioning motion at rapid traverse (G0, RTLION, RTLIOF)	10-38
10.5.6	Linear interpolation with feed (G1)	10-39
10.5.7	Positioning movements (POS, POSA, WAITP)	10-40

10.5.8	Programmable block change for positioning axes (FINEA, COARSEA, IPOENDA, IPOBRAKE)	10-41
10.5.9	Circular interpolation (G2, G3, I, J, K, CR)	10-44
10.5.10	Spline (ASPLINE, CSPLINE, BSPLINE)	10-47
10.5.11	Polynomial interpolation (POLY)	10-54
10.5.12	Involute interpolation (INVCW, INVCCW)	10-58
10.5.13	Chamfer and rounding (CHF, CHR, RND, RNDM, FRC, FRCM)	10-62
10.6	Path response	10-64
10.6.1	Exact stop (G60, G9), target range (G601, G602, G603)	10-65
10.6.2	Continuous-path mode (G64, G641, ADIS, ADISPOS)	10-67
10.6.3	Acceleration response (BRISK, SOFT, DRIVE)	10-70
10.6.4	Programmable dynamic limitation	10-72
10.7	Dwell (G4)	10-73
10.8	Coupled motion (TRAILON, TRAILOF)	10-74
10.9	Tangential control (TANG, TANGON, TANGOF)	10-76
10.10	Measurement	10-78
10.10.1	Block-specific measurement (MEAS, MEAW)	10-78
10.10.2	Axial measurement (MEASA, MEAWA, MEAC)	10-80
10.11	Travel to fixed stop (FXST, FXSW, FXS)	10-84
10.12	Stop preprocessor (STOPRE)	10-86
10.13	Working area limitations (G25, G26, WALIMON, WALIMOF)	10-86
10.14	M functions	10-88
10.15	H functions	10-90
10.16	Tool offset values (T functions)	10-91
10.17	Protection zones (NPROTDEF, EXECUTE, NPROT)	10-93
10.18	Fundamentals of variable NC programming	10-97
10.19	R parameters (arithmetic parameters)	10-103
10.20	System variables (\$P_, \$A_, \$AC_, \$AA_)	10-105
10.21	FIFO variable (\$AC_FIFO1[...] to \$AC_FIFO10[...])	10-113
10.22	User variables	10-118
10.23	Program jumps (GOTOF, GOTOB, GOTO, GOTOC, LABEL, IF) ...	10-124
10.24	Control structures	10-126
10.25	Axis variable	10-131
10.26	String operations	10-132
10.27	Reading, writing and deleting a file	10-134
10.28	Program coordination (INIT, START, WAITE , WAITM, WAITMC, SETM, CLEARM)	10-139
10.29	Subroutine technique	10-143
10.30	Asynchronous subroutines (ASUB)	10-149

10.31	Activating machine data (NEWCONF)	10-153
10.32	Synchronized actions	10-154
10.33	Oscillation	10-176
10.34	Master value coupling	10-181
10.35	Electronic gear	10-187
10.36	Axis replacement	10-192
10.37	Speed feedforward control (FFWON, FFWOF)	10-195
10.38	Overview of statements	10-196
11	Application Example	11-1
11.1	On-the-fly curve table switching	11-1
11.2	Conveyor tracking	11-6
12	Troubleshooting	12-1
12.1	LED indicators	12-2
12.2	Error messages and their effect	12-6
12.3	Error lists	12-8
13	Handlings Transformation	13-1
13.1	Parameterization of “Handling transformation”	13-5
13.2	Programming the standard function blocks for “Handling transformation” with the HPU or HT 6	13-10
13.2.1	FC 21: HPUHT6 – Transfer of the HPU / HT 6 signals to and from the interface (user DB)	13-13
13.2.2	User data block (user DB 21)	13-17
13.3	User alarms and user messages	13-22
13.3.1	FC 20: AL_MSG – user alarms and user messages	13-24
13.3.2	Message signals in DB 20	13-25
13.3.3	Configuring alarm and message texts for PHG	13-29
13.3.4	Configuring alarm and message texts for HT 6	13-31
13.3.5	Diagnostic buffer of the CPU	13-33
13.4	Changing the global data communication (SDB 210)	13-34
13.5	Functions	13-35
13.5.1	Definition of terms	13-35
13.5.2	Configuring the kinematic transformation	13-38
13.5.3	Kinematic descriptions	13-48
13.5.4	Operating modes	13-67
13.6	NC programming	13-69
13.6.1	Tool orientation	13-69
13.6.2	Singular positions and their handling	13-70
13.6.3	Calling the handling transformation	13-71
13.6.4	Tool programming	13-72
13.6.5	Cartesian PTP traversing	13-73
13.6.6	Example program for handling transformation	13-77
13.7	Error handling	13-78

A	Technical Specifications	A-1
B	List of Abbreviations	B-1
	Index	Index-1

Product Summary

1

Section overview

Section	Title	Page
1.1	The FM 357-2 in the S7-300 programmable controller system	1-3
1.2	Module description	1-9
1.3	Overview of module functions	1-12

What can the FM 357-2 do?

The FM 357-2 is a microprocessor-based multi-axis module for the control of servo (analog) and/or stepper drives and/or SIMODRIVE 611-U drives via PROFIBUS-DP.

The module has a maximum of four channels and can control up to four axes.

It is a high-performance module for positioning independent or synchronized axes.

It can operate rotary and linear axes.

The FM 357-2 has a variety of operating modes.

The following firmware versions are available on memory card for the FM 357-2 product version 3 and later:

- FM 357-2 with license L
- FM 357-2 with license LX (extended functional scope)
- FM 357-2 with license H (Handling)

You will need the following documentation in addition to this manual for the “Handling” function:

- *Operator Components* manual, Order No.: 6FC5 297-6AA50-0BP0
- *Handheld Programming Device Operator’s Guide*, Order No.: 6FC5 298-5AD20-0BP1
- *Handheld Terminal HT 6 Operator’s Guide*, Order No.: 6FC5 298-4AD60-0BP0

It can be linked and adapted to user circumstances by parameterizing it as required by the system.

The module has a non-volatile memory for the storage of parameter data.

- Data backup with backup battery
- Data backup on memory card

Where can the FM 357-2 be used?

The FM 357-2 can be used for both simple positioning and complex traversing profiles with high-precision interpolation and synchronization of axis combinations.

Typical uses for the multi-axis module might include:

- Conveyor equipment
- Transfer lines
- Assembly lines
- Special-purpose machines
- Food industry
- Handling equipment
- Loaders
- Packaging machines

1.1 The FM 357-2 in the S7-300 programmable controller system

How is the FM 357-2 integrated in the S7-300?

The FM 357-2 has been designed as a function module for the programmable controller system S7-300.

The S7-300 programmable controller consists of a CPU and a variety of I/O modules mounted on a mounting rail.

The configuration may have one or more tiers.

A SIMATIC S7-300 CPU can control up to four tiers (subracks) with as many as eight bus stations each (see Figure 1-1).

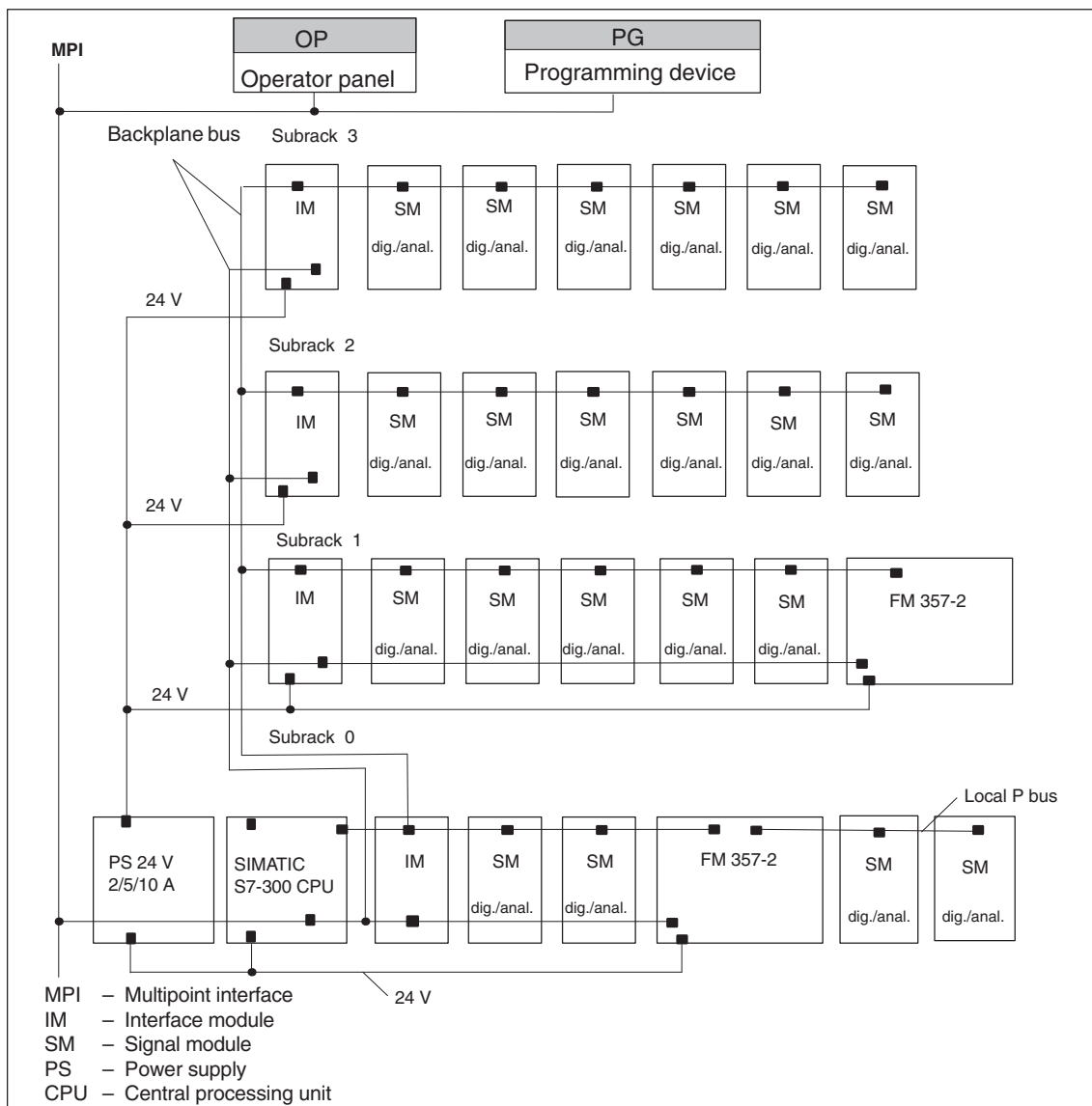


Fig. 1-1 Centralized multi-tier configuration of a SIMATIC S7-300 with FM 357-2 (example)

Single-tier configuration

A single-tier configuration consists of the S7-300 CPU, the FM 357-2 and up to seven further modules (SM, FM).

The SIMATIC S7-300 CPU drives all eight bus nodes, and provides the logic power supply for its signal modules.

A maximum of three FM 357-2 (amongst them, max. one FM 357-2H) modules may be connected to a CPU.

Hybrid operation of the FM 357 and FM 357-2 is not possible.

The FM 357-2 uses a separate connection for its logic power supply.

Multi-tier configuration

In a multi-tier configuration, an interface module (IM) must be installed in subrack 0 to the right of the S7-300 CPU. Eight modules (SMs, FMs and the FM 357-2) can be installed alongside the interface module.

Subrack 1 and each further subrack begins with an interface module (IM), and can contain a further eight modules (SMs, FMs, FM 357-2). The logic power is supplied by the IM, which has a separate power supply connection.

A maximum of three FM 357-2 (amongst them, max. one FM 357-2H) modules may be connected to a CPU.

Hybrid operation of the FM 357 and FM 357-2 is not possible.

In configuring the mechanical layout of your controller, you should note the following properties of the modules:

- Mounting dimensions
- Power consumption from 24 V
- Power consumption from the 5 V P bus supply

For further information about multi-tier configurations and configuring instructions, please refer to manual *S7-300 Programmable Controller, Hardware and Installation*

Distribution of equipment

The multi-tier arrangement allows you to distribute the equipment by linking individual tiers by means of 10 m long IM connecting cables.

In a two-tier configuration, for example, signal modules can be positioned at a maximum of 10 m from the FM 357-2 and, in a four-tier configuration, at up to 30 m.

Local P bus

The FM 357-2 is capable of bridging a local bus segment. All modules installed to the right of the FM 357-2 (the maximum number is 2) can then be addressed from the FM 357-2 as high-speed I/Os once the latter has been powered up.

The local P bus can only be used in centralized installations.

Distributed installation via PROFIBUS DP

The FM 357-2 can be operated as a distributed module on S7-300/400 systems via PROFIBUS DP with the ET 200M (with IM 153-2) or with a CPU 31x-2 DP.

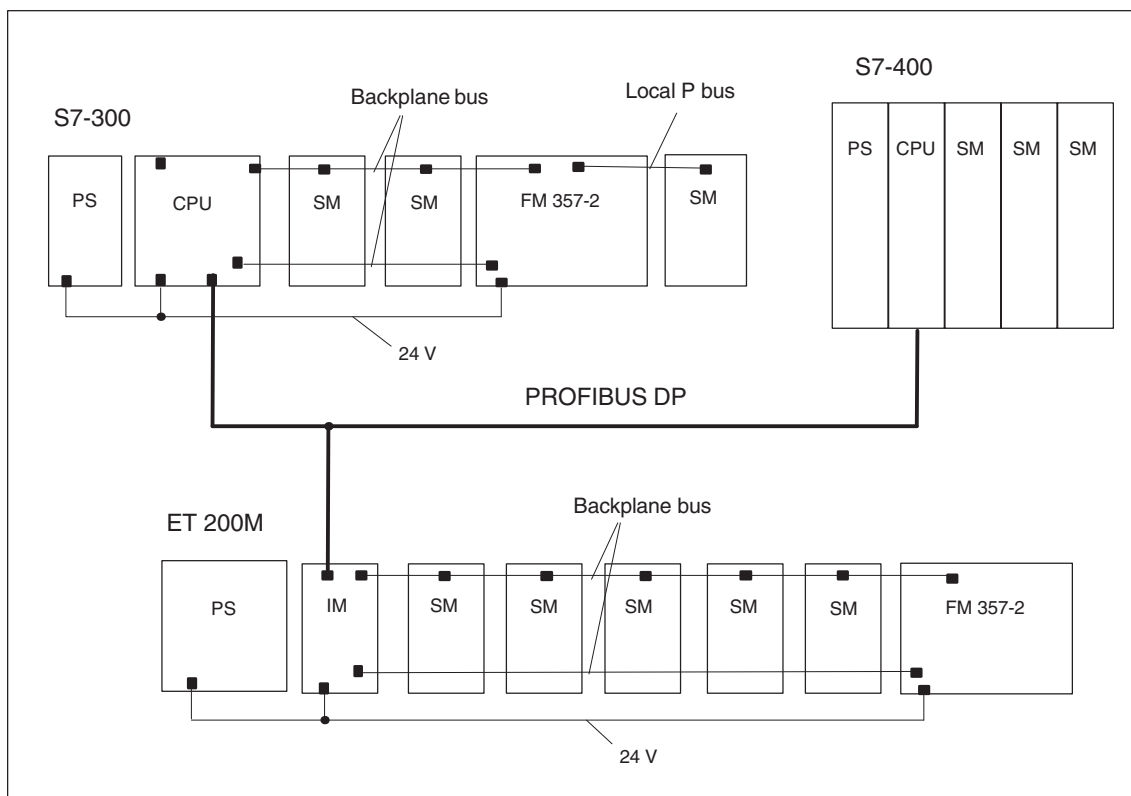


Fig. 1-2 Distributed configuration (example)

System overview

A positioning control system with the FM 357-2 consists of various individual components which are shown in Figure 1-3.

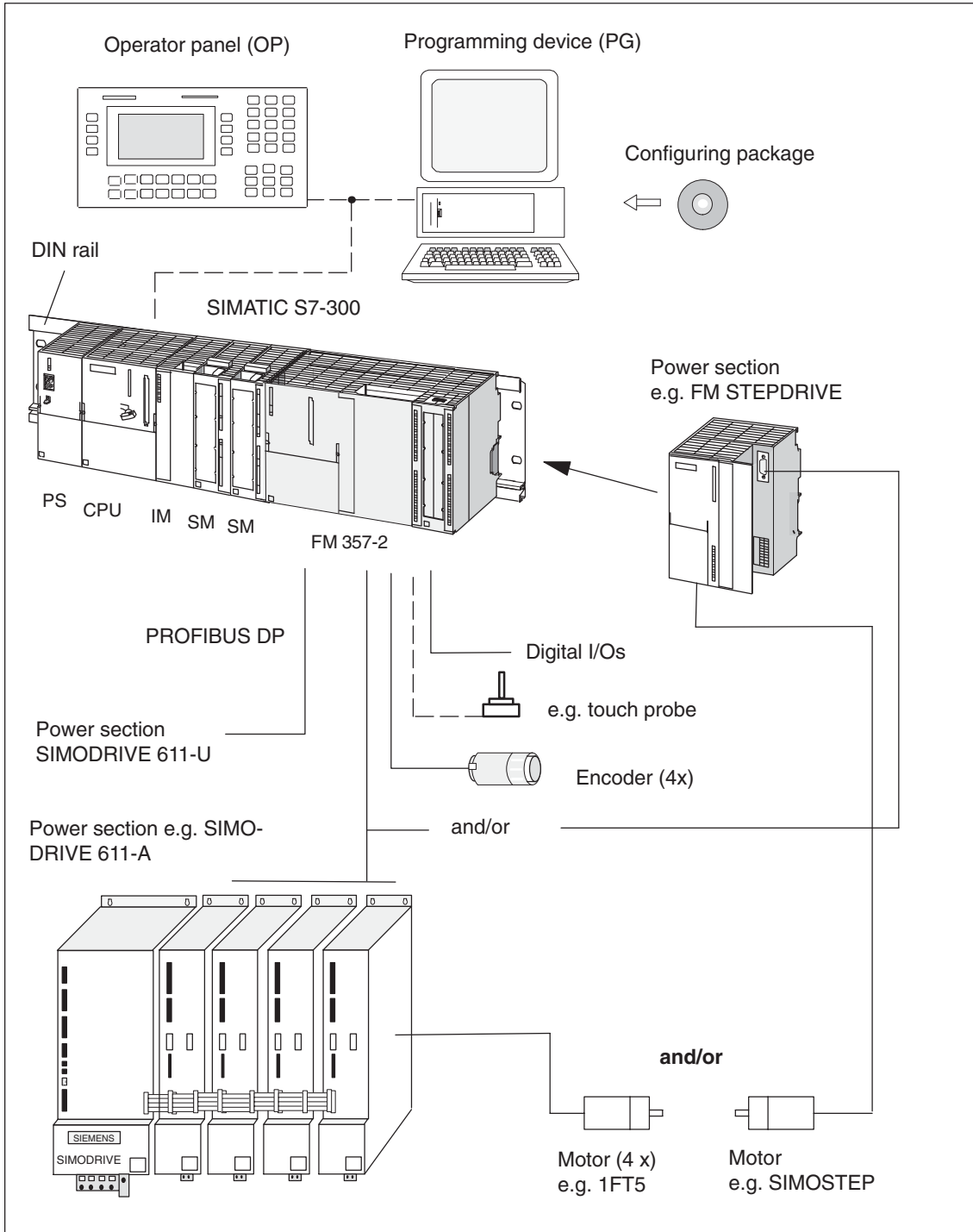


Fig. 1-3 System overview (diagrammatic)

Components

The main components and their functions are listed in Table 1-1.

Table 1-1 Components of a positioning controller

Component	Function
DIN rail	... is the module mounting rack for the S7-300.
FM 357-2	... is the multi-axis module. It is controlled by the S7 CPU.
CPU	... executes the user program; powers the S7-300 backplane bus at 5 V; communicates with the programming device and operator panel via the MPI interface and with the FM 357-2 via the P bus.
Power supply (PS)	... converts line voltage (120/230 V AC) to 24 V DC operating voltage to power the S7-300.
Signal modules (SM)	... adapts various process-signal levels to the S7-300.
Interface module (IM)	... connects the individual tiers of an S7-300 with one another (applies to multi-tier configuration; see Figure 1-1).
Programming device (PG)	... configures, parameterizes, programs and tests the S7-300 and the FM 357-2.
Operator panel (OP)	... acts as the human-machine interface (operator control and monitoring). It is not absolutely essential for the operation of an FM 357-2 .
Power section	... actuates the motor.
Motor	... drives the axis.
Encoder	... the path measurement system that detects the current position of the axis.
Configuring package	<p>... includes CD ROM with:</p> <ul style="list-style-type: none"> • FM 357-2 standard function blocks • FM 357 standard function blocks • "Parameterize FM 357-2" parameterization tool • Preconfigured user interfaces for OP 17/27 for the FM 357-2 • Preconfigured user interfaces for OP 17 for the FM 357 • Manual in PDF format • Getting started, short start-up guide in PDF format • NC variable selector • Firmware update for FM 357-2

System overview of data handling

The following diagram provides an overview of the data storage strategy.

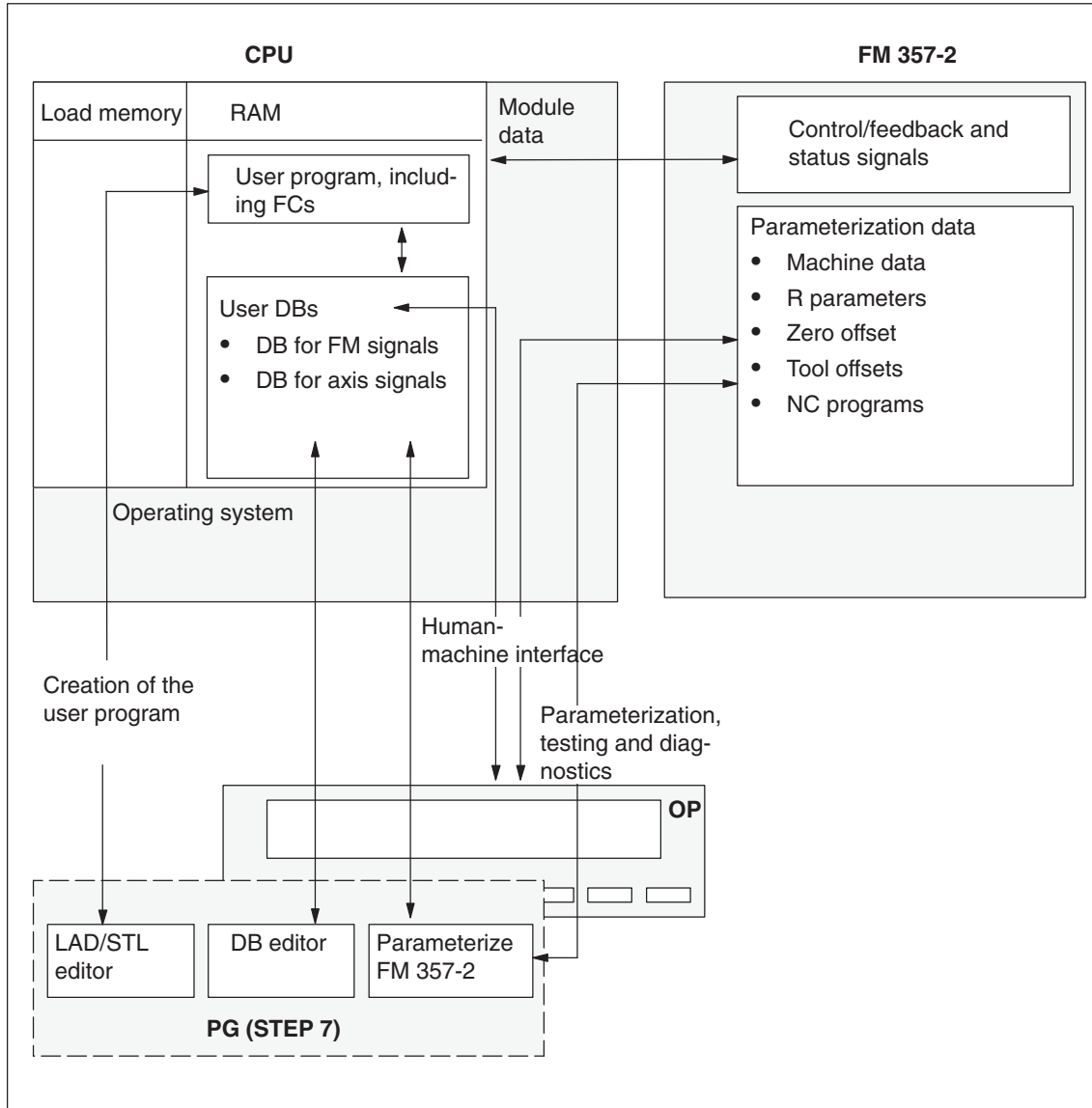


Fig. 1-4 Data storage concept

1.2 Module description

View of FM 357-2

Figure 1-5 shows the FM 357-2 module with its interfaces and front-panel elements (error and status displays).

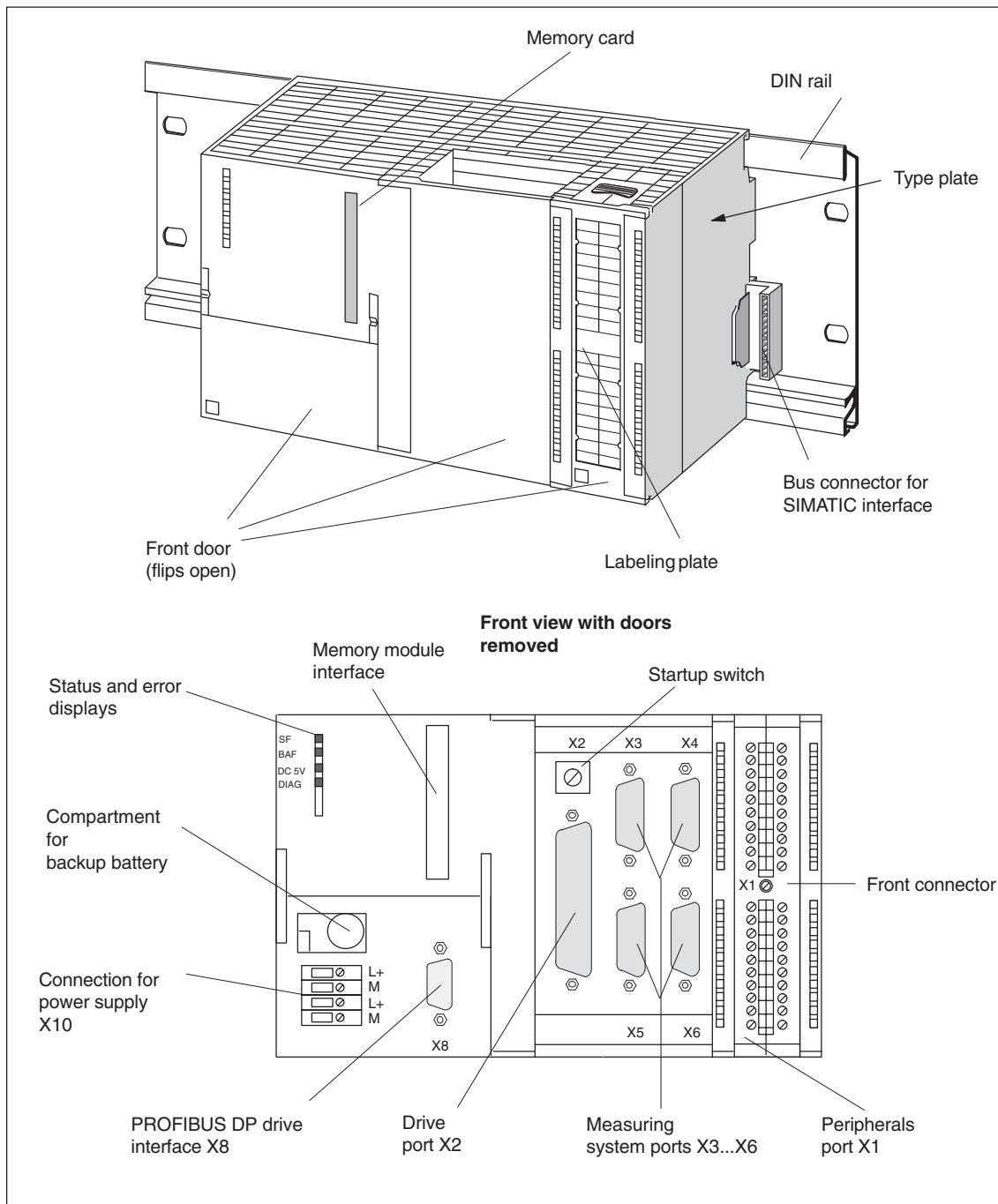


Fig. 1-5 Location of ports and front panel elements

Interfaces

Table 1-2 describes the interfaces and their meaning.

Table 1-2 Interfaces

Interfaces	Description
Bus connector for SIMATIC interface	Rear-panel connector for connecting the FM 357-2 to other S7 modules via the S7 backplane bus
Drive interface	50-pin male sub D connector (X2) for connecting analog and/or stepper drives (max. 4 axes)
Measuring system interface	15-pin female sub-D connector (X3 to X6) for connecting encoders (max. 4)
I/O interface	40-pin male sub D connector (X1) for connecting the high-speed digital I/Os (including the probe) and for wiring the FM-READY relay
Power supply port	4-pin screw-type terminal connection (X10) for connecting the 24 V load power supply
Memory module interface	68-pin connector for memory card
PROFIBUS DP drive interface	9-pin female sub-D connector (X8) for connecting digital servo drives (up to 4 axes)

LED indicators

Four LED indicators are arranged on the front panel of the FM 357-2. Table 1-3 describes these LEDs and what they mean.

Table 1-3 Status and error displays

LED	Significance
SF (red) – group fault	This LED indicates an error condition in the FM 357-2. (see Troubleshooting, Chapter 12)
DC 5V (green) – logic power supply	This LED indicates that the hardware is ready for operation. (see Troubleshooting, Chapter 12)
DIAG (yellow) – diagnostics	This LED indicates various diagnostic states (flashing). (see Troubleshooting, Chapter 12)
BAF (red) – battery fault	This LED indicates a battery failure (you need to change the battery) (see Troubleshooting, Chapter 12)

Operator elements

Start-up switch (rotary switch)

The rotary switch is used during start-up.

Battery compartment

For connection of a lithium battery, with reassembled connectors.

Type plate

Figure 1-6 explains all the information displayed on the type plate.

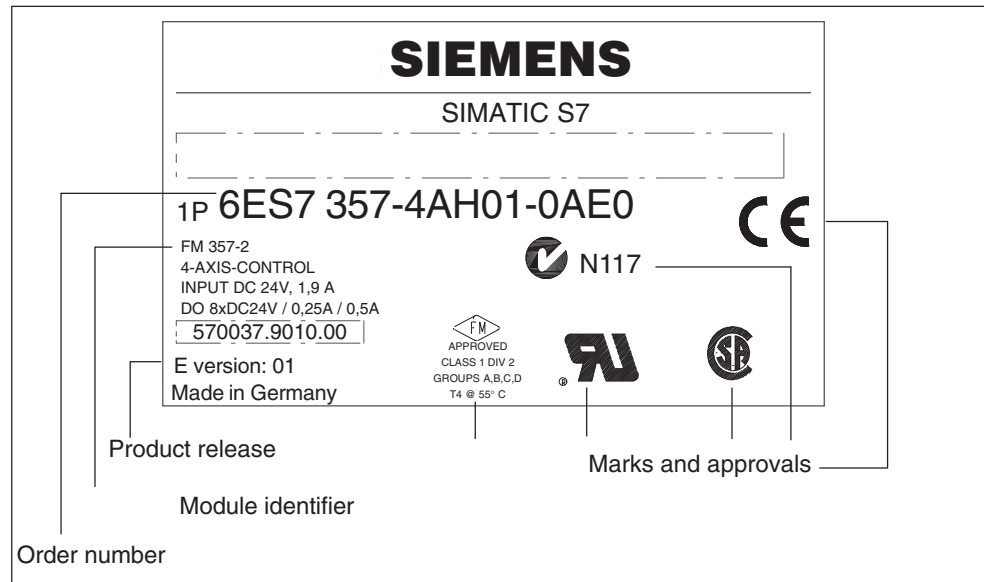


Fig. 1-6 Nameplate of FM 357-2

1.3 Overview of module functions

Overview

The following main functions are implemented on the FM 357-2:

- Mode control
- Actual-value capture
- CL position control
- Stepper motor control
- Multi-axis positioning
- Interpolation and synchronization functionality
- Digital I/Os
- Software limit switches, protection zones and working area limitations
- Block sequence control
- Diagnostics and troubleshooting
- Data storage on the FM 357-2
- Data storage on memory card
- Local P bus

Mode control

The mode must be transferred to the FM from the OP, parameterization tool or the user program.

The FM 357-2 has the following operating modes and submodes.

- Jogging
- Incremental Relative
- Reference-point approach
- MDI (Manual Data Input)
- Automatic
- Automatic Single Block

Encoder

An incremental encoder or absolute encoder (SSI) can be connected to the measuring system interface.

CL position control

The position controller performs the following tasks:

- Control of the drive at the right speed while a movement is being performed.
- Accurate approach by axis into programmed target position
- Maintenance of the axis in position in the face of interfering factors.

Stepper motor control

In addition to servo drives, the FM 357-2 is also capable of operating up to four stepper drives via a pulse interface, under open-loop control (without encoder) or closed-loop control (with encoder).

Multi-axis positioning

Up to four axes can be positioned in mutual independence. The movement commands are issued by the NC program or the CPU.

Interpolation and synchronization functionality

In an axis grouping, a maximum of four axes can interpolate to execute a linear, circular or spline motion. Synchronization functions link one or more following axes to a leading axis.

Digital I/Os (on-board)

You can use the digital I/Os freely to suit your own application.

You might connect:

- Switch for reference point approach
- Touch probes
- I/Os available for use in NC program

Software limit switches, protection zones and working area limitation

The working area is automatically monitored after axis synchronization. The range can be defined by software limit switches, with reference to an axis, or by protection zones, with reference to an area.

Block sequence control

Autonomous processing of NC programs, including subroutines, stored in non-volatile memory on the module.

Diagnostics and troubleshooting

Module startup and operation are monitored by error and diagnostic alarms. Faults or errors are reported to the system and displayed by the LEDs on the module.

Data storage on the FM 357-2

Parameter data (machine data, R parameters, tool offset data, zero offsets and NC programs) are stored in non-volatile memory on the FM 357-2 .

Data storage on memory card

A memory card with a valid license must always be plugged in before you can use the FM 357-2.

You can use the memory card for:

- Backing up your system software and parameter data
- Series startup of an FM 357-2
- Replacing modules without a programming device

Local P bus (in centralized installations only)

The local P bus is used to extend the number of digital I/Os of the FM 357-2. Digital and analog I/O modules (a maximum of 2 modules) can be connected.

MPI connection

Up to 3 MPI connections are possible with the FM 357-2.



Fundamental Principles of Motion Control

2

Position-controlled motion control for servo axes (analog)

The FM 357-2 allows the motion of a maximum of four axes to be controlled as a function of position. The FM 357-2 achieves this by assigning an analog output to each axis for the speed setpoint and assigning an encoder input to each axis for cyclical reading of the actual position.

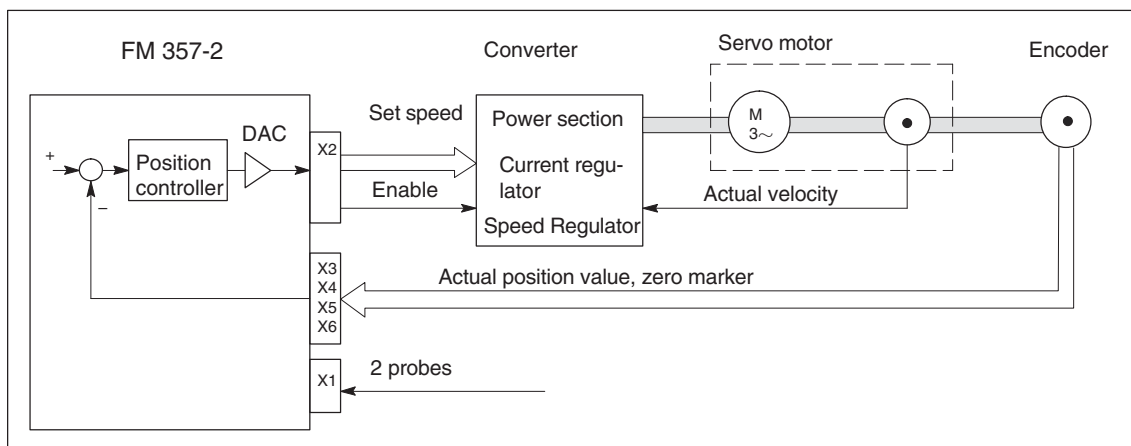


Fig. 2-1 Servo system with converter, e.g. SIMODRIVE 611-A

Incremental encoder

Encoders are generally connected as position sensors. These supply count pulses (number depends on encoder resolution) for the path increments traversed. Rotary encoders or linear-scale encoders can be used.

Absolute encoder (SSI)

Instead of conventional incremental encoders that supply only a dimension for the traversed path, it is possible to connect absolute encoders with a serial interface. A reference point is no longer needed, since these encoders always return the absolute position as the actual value.

Position-controlled motion control for servo axes (digital)

The FM 357-2 allows the motion of a maximum of four axes to be controlled via PROFIBUS DP as a function of position.

The following PROFIBUS protocol is used for this purpose:

PROFIDRIVE drive profile version 3 (cycle-synchronous, equidistant)

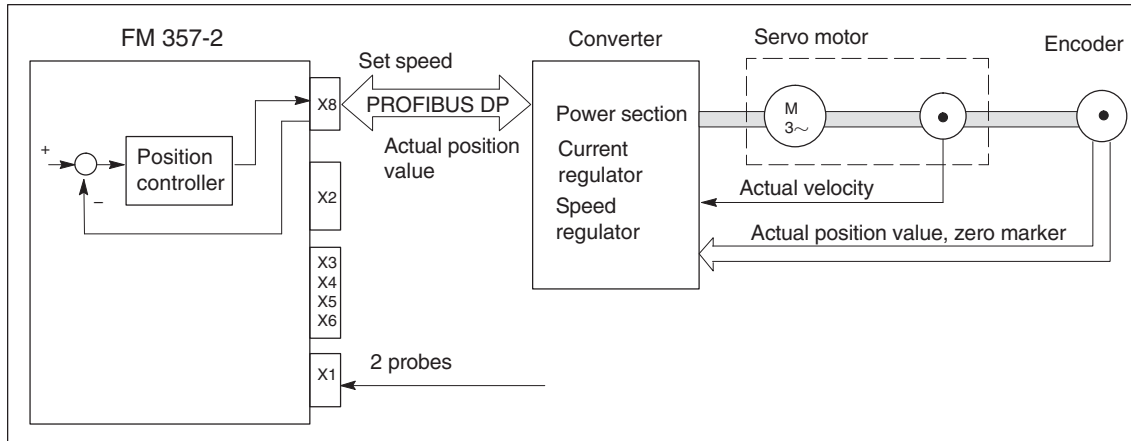


Fig. 2-2 Servo system with SIMODRIVE 611-U converter

Stepper motor control

In addition to its analog setpoint outputs, the FM 357-2 also has pulse outputs for a maximum of 4 stepper motor axes. The stepper motor is controlled by clock pulses. The number and frequency of clock pulses determine the velocity of the axis. The actual position value is not measured in open-loop control mode; the position controller interprets the number of output pulses (distance setpoint) as an actual value. The motor must not lose any increments if positioning is to be accurate.

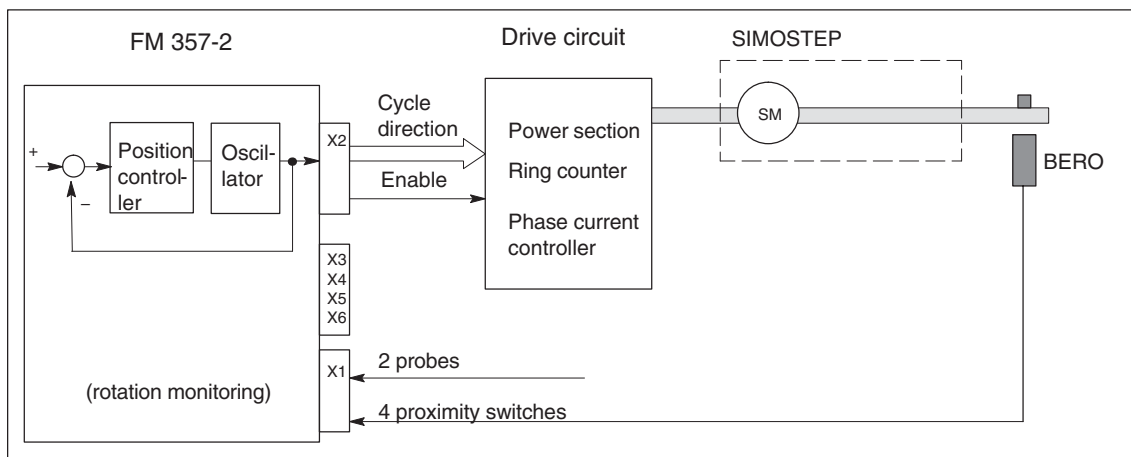


Fig. 2-3 Controlled stepper motor system with drive circuit, e.g. FM STEPDRIVE

Stepper motor control as a function of position

With one encoder input per axis, the FM 357-2 provides the option of controlling stepper motors as a function of position like a servo axis.

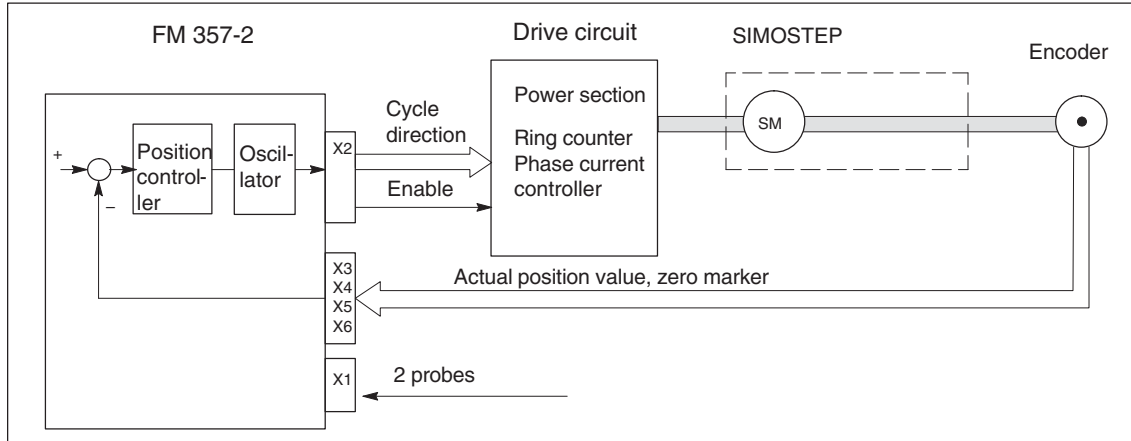


Fig. 2-4 Controlled stepper motor system with drive circuit, e.g. FM STEPDRIVE



Installation and Removal

Section overview

Section	Title	Page
3.1	Installing the FM 357-2	3-3
3.2	Installing firmware/firmware update	3-4
3.3	Removing and replacing the FM 357-2	3-6

Overview

The FM 357-2 multi-axis module is integrated into the SIMATIC S7-300 controller as an I/O module.

Configuring the mechanical installation

A description of the range of possible mechanical installation variations and how they can be configured can be found in the manual *S7-300 Programmable Controller; Hardware and Installation*.

We have given just a few supplementary pointers below.

Mounting position of FM 357-2

The preferred mounting position is horizontal.

In vertical installations, please observe the ambient temperature restrictions (max. 40 °C).

What must be noted in relation to mechanical configuration?

The slot location of the FM 357-2 module will depend on the mechanical configuration of your control. You must follow the rules below:

1. A maximum of eight SMs or FMs (including FM 357-2) may be installed on each tier.
2. The maximum number is restricted by the width of the modules or length of your mounting rail.

The FM 357-2 requires a mounting width of 200 mm.

3. The maximum number is restricted by the total power consumed by all modules to the right of the CPU from the 5V backplane bus supply.

The CPU 314, for example, can supply a maximum of 1.2 A.

The FM 357-2 requires 100 mA of this amount.

Local P bus (centralized installations only)

If you install additional digital inputs/outputs on the local P bus of the FM 357-2, you should insert the corresponding SMs to the right of the FM 357-2.

Installation of FM STEPDRIVE

FM STEPDRIVE modules can be installed additionally to the eight SMs or FMs. They are not connected to the SIMATIC bus, and must therefore only be matched to the mounting width.

Important safety rules

The integration of an S7-300 with an FM 357-2 module in a plant or system is subject to important safety rules which you must observe.

These rules and specifications are explained in the manuals *S7-300 Programmable Controller, Hardware and Installation*, and *S7-300, M7-300 Programmable Controllers Module Specifications*.

3.1 Installing the FM 357-2

Rules

No particular protective measures (ESD Guidelines) are necessary for the installation of the FM 357-2.



Warning

Install the FM 357-2 only after all power to the S7-300 has been switched OFF.

Tool required

4.5 mm screwdriver

Procedure

To install the FM 357-2, proceed as follows:

1. The FM 357-2 is supplied with a bus connector. Plug this into the bus plug of the module to the left of the FM 357-2. (The bus plug is on the back; you may have to loosen the module already in place.)

If further modules are to be mounted to the right, plug the bus connector of the next module into the right backplane bus connector on the FM 357-2.

If the FM 357-2 is the last module in the rack, do not connect a bus connector.
2. Hook the FM 357-2 onto the mounting rail and swing it down into place.
3. Tighten the screws on the FM 357-2 (tightening torque approximately 80...110 Ncm).
4. Once you have installed the modules, you can allocate a slot number to each one. Slot labels for this purpose are enclosed with the CPU.

For information on appropriate numbering schemes and instructions for attaching slot labels, please refer to manual *S7-300 Programmable Controller, Hardware and Installation*.

Notice

The slot determines the initial address of each module. To find out how to allocate the module start address, please refer to the manual *S7-300 Programmable Controller, Hardware and Installation*.

3.2 Installing firmware/firmware update

Programming the memory card

Use the program "Firmware Update for FM 357-2" to copy a new firmware to the memory card. Older software versions are to be found on the CD-ROM under "Add_on_Firmware_BFct". Select the appropriate "fm2_upd.abb" file in accordance with the desired version and the memory card license.

Proceed as follows:

1. Insert the memory card into the "Mem-Card" slot on the PG or into the external prommer.
2. Call "Update-FM357-2" by one of the following means:
 - via the Start menu (SIMATIC → Step7 → Update-FM357-2)
 - via the desktop icon "Update-FM357-2"
 - by executing the file "fmupdate.exe" in the target directory of the installation, e.g. Run... C:\Siemens\Step7\s7fw357\fmupdate.exe
3. After successful data preparation, the dialog box for data transfer is displayed. Use the "..." button to select the file to be transferred. Selecting the "Start data transfer to memory card" button starts the deletion and programming sequence.
4. If no further memory cards are to be written, you can close the dialog box once the message "Successfully created memory card" is displayed.

Notice

The FM 357-2-coded memory card is deleted and reprogrammed. This deletes all data currently on the memory card. Only the license is left unaffected.

Installing the firmware from memory card

Please proceed as follows:

1. Switch the control off and plug the memory card containing the firmware into the control.

Note

The following firmware versions are available for the FM 357-2:

- FM 357-2 with license L
- FM 357-2 with license LX
- FM 357-2 with license H

The appropriate CPU must be in the RUN status.

2. Set start-up switch to position A.
3. Switch control “ON” → The system software and data are transferred from the memory card to the control.

The “SF” LED lights up and the “DIAG” LED flashes in cycles of four during the data transfer.

An error is indicated (see also Table 12-2):

- if the “SF” LED flashes (2 Hz). The firmware cannot be transferred to the internal memory of the FM (e.g. because part of the system software in the internal memory has been deleted).
 - If the “SF” LED lights up and the “DIAG” flashes rapidly three times and then flashes periodically three times. The firmware cannot be transferred to the internal memory of the FM (e.g. because the system software on the memory card is defective).
4. When the “DIAG” LED flashes five times cyclically and the “SF” LED goes out, the data transfer is complete → Control “OFF”.
 5. Leave the memory card in the FM 357-2.
 6. Set the start-up switch on the FM 357-2 to position 1
 7. Control “ON” → Wait for control to power up with defaults. The “DIAG” LED flashes cyclically (3 Hz) after approximately one minute.
 8. Switch control “OFF”
 9. Set the start-up switch on the FM 357-2 to position 0
 10. Control “ON” → The control powers up with the new firmware

The license on the memory card is checked while the FM 357-2 is powering up (see Section 7.2).

Notice

Before you install a new software version, you must back up **all** FM 357-2 data (e.g. parameter data), which do not belong to the firmware, on a data storage medium using the parameterization tool.

After you have updated the firmware, please proceed as follows:

1. Switch control "OFF"
 2. Set the start-up switch on the FM 357-2 to position 1
 3. Control "ON" → Wait for control to power up with defaults (approx. 40 sec.)
 4. Switch control "OFF"
 5. Set the start-up switch on the FM 357-2 to position 0
 6. Control "ON" → The control powers up with the new firmware
-

3.3 Removing and replacing the FM 357-2

Overview

You can only replace the complete FM 357-2.



Warning

It is only possible to replace the FM 357-2 with the load power supply switched off. Switch the power supply off, e.g. by operating the on/off switch on the PS power supply module.

Tool required

4.5 mm screwdriver

Notice

It is easier to start up the new module after a module replacement if you follow the recommendations for data backup on initial start-up.

Data backup on memory card

All parameter data and the firmware are saved. All data already stored on the memory card are overwritten (except for the firmware license).

Procedure for backing up data:

1. Switch control "OFF"
2. Insert the memory card in the FM 357-2 module.
3. Set the start-up switch on the FM 357-2 to position 0
4. Switch control "ON" and wait for it to power up. After approx. one minute the LED "DIAG" flashes cyclically (3 Hz).
5. Set the start-up switch on the FM 357-2 to position 3

An NC Restart is initiated automatically after about 10 seconds and data backup commences (LEF "SF" lights up and LED "DIAG" flashes twice cyclically).

An error is indicated if the "SF" lights up and the "DIAG" flashes rapidly three times and then once. Check whether the wrong memory card is inserted (see also Table 12-2).

6. The data backup is complete when the "DIAG" LED flashes three times and the "SF" LED goes out.

The control does **not** power up automatically after a data backup.

7. Switch control "OFF"

Note:

A battery error must not be active.

Notice

You should always perform this data backup procedure after starting up your system for the first time.

Remove a defective module

Please proceed as follows to remove the FM 357-2:

1. Open the front doors. If necessary, remove the labeling strips.
2. Separate the power supply connections on the terminal block.
3. Disconnect the sub-D connector to the encoder and drive unit.
4. Unlock and remove the front connector.

Slacken the fastening screws in the center of the front connector. Hold the grips and pull the front connector out.

5. Undo the fastening screws and lift the module out upwards.

Install the new module

Proceed as follows:

1. Remove the top part of the front connector coding from the new module.
2. Engage the module (of the same type) on the mounting rail, lower it into position and tighten the mounting screws.
3. Insert the front connector and set it to the operating position (tighten the fastening screw). The coding element is adjusted so that the front connector only fits this module.
4. Insert the sub-D connector.
5. Connect up the load power supply on the terminal block.
6. Close the front doors and replace the labeling strips.

The control is now ready again and can be started up. You can now read in the firmware and the backup parameter data from the memory card.

Reading in backup data from the memory card

Proceed as follows:

1. With the control switched off, insert the memory card.
2. Set start-up switch to position A
3. Switch the control "ON" → The firmware and parameter data are transferred from the memory card to the control.
LED "DIAG" flashes cyclically 4 times while the data are being transferred.
4. When LED "DIAG" flashes 5 times cyclically, the data transfer is finished → Control "OFF".
5. Leave the memory card in the FM 357-2.
6. Set start-up switch to position 0
7. Control "ON" → The control powers up with the firmware and the backup data from the memory card.



Wiring

4

Section overview

Section	Title	Page
4.1	Wiring plan of a FM 357-2	4-3
4.2	Connecting the power supply	4-7
4.4	Description of the drive port	4-12
4.5	Connecting the drive unit	4-18
4.6	Description of the measurement system port	4-22
4.7	Connecting the encoders	4-27
4.8	Description of the I/O port	4-29
4.9	Wiring up the front connector	4-33
4.10	Installing and changing the backup battery	4-37

Safety rules

To ensure the safe operation of your system, you must take the following measures and adapt them to suit the conditions in your plant:

- An EMERGENCY OFF concept meeting appropriate safety regulations (e.g. European standards EN 60204, EN 418 and associated standards).
- Additional measures for limiting axis end positions (e.g. hardware limit switches).
- Equipment and measures for protecting the motors and power electronics in accordance with the installation guidelines for SIMODRIVE and FM STEP-DRIVE/SIMOSTEP.

We also recommend you carry out a risk analysis in accordance with basic safety requirements / Appendix 1 of the EC machine directive 89/392/EEC, in order to identify sources of danger affecting the complete system.

Further references

Please also note the following sections in the manual *S7-300 Programmable Controller, Installation*:

- Flash protection and overvoltage protection: Section 4.2
- Guidelines for handling of electrostatic sensitive devices (ESDs)
- Configuring the electrical installation
- Wiring

For further information about EMC guidelines, we recommend the description in: *Equipment for Machine Tools, EMC Guidelines for WS/WF Systems*, Order number: 6ZB5 440-0QX01-0BA1.

Standards and regulations

The FM 357-2 must be wired up in compliance with the relevant VDE Guidelines.

4.1 Wiring an FM 357-2

FM 357-2 with servo drive (analog)

Figure 4-1 shows the interconnections between the individual components of the multi-axis control with FM 357-2 and the servo drive (analog).

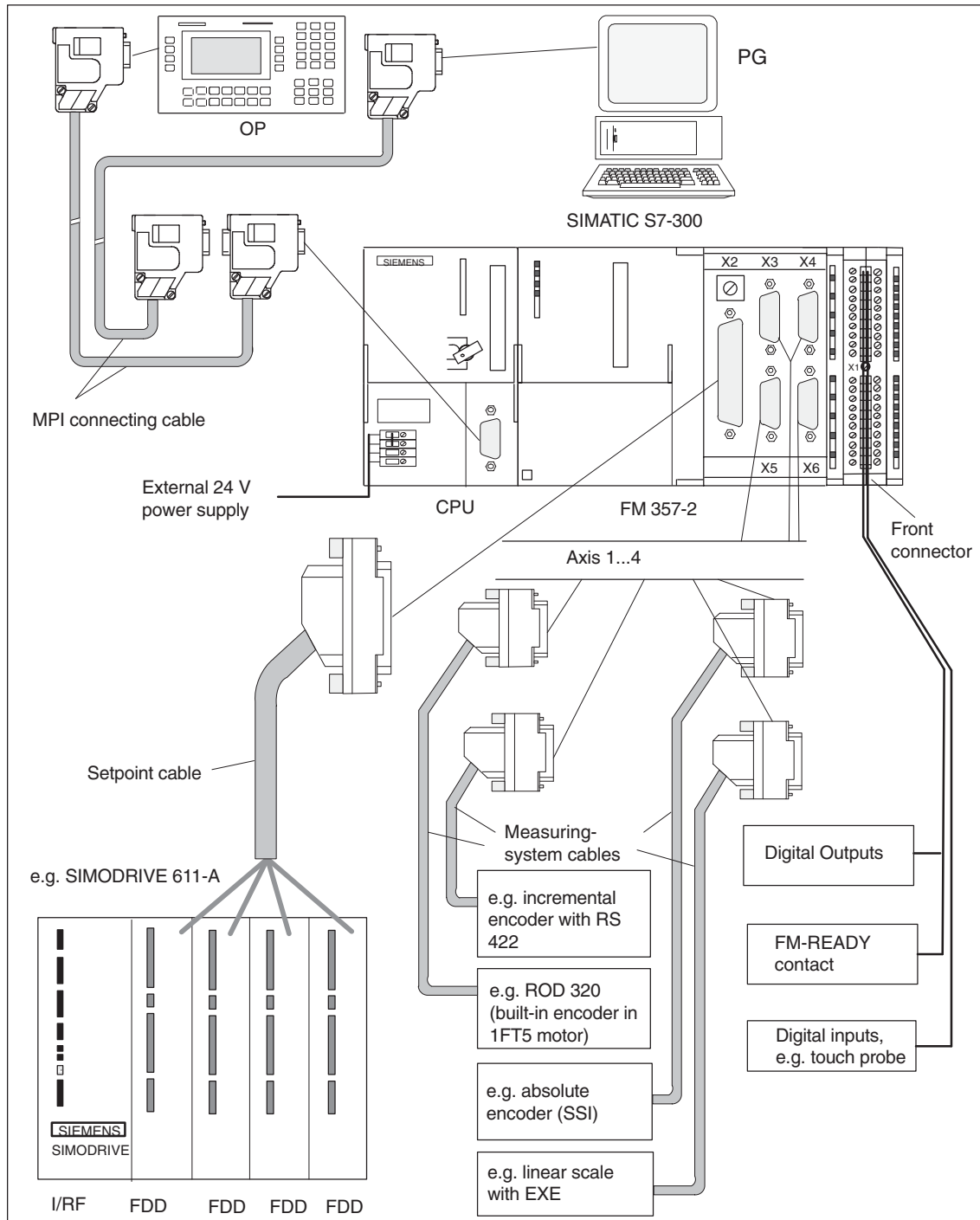


Fig. 4-1 Overview of connecting cable between FM 357-2 and analog servo drive (example)

FM 357-2 with servo drive (digital) via PROFIBUS-DP

Figure 4-2 shows the interconnections between the individual components of the multi-axis control with FM 357-2 and the servo drive (digital).

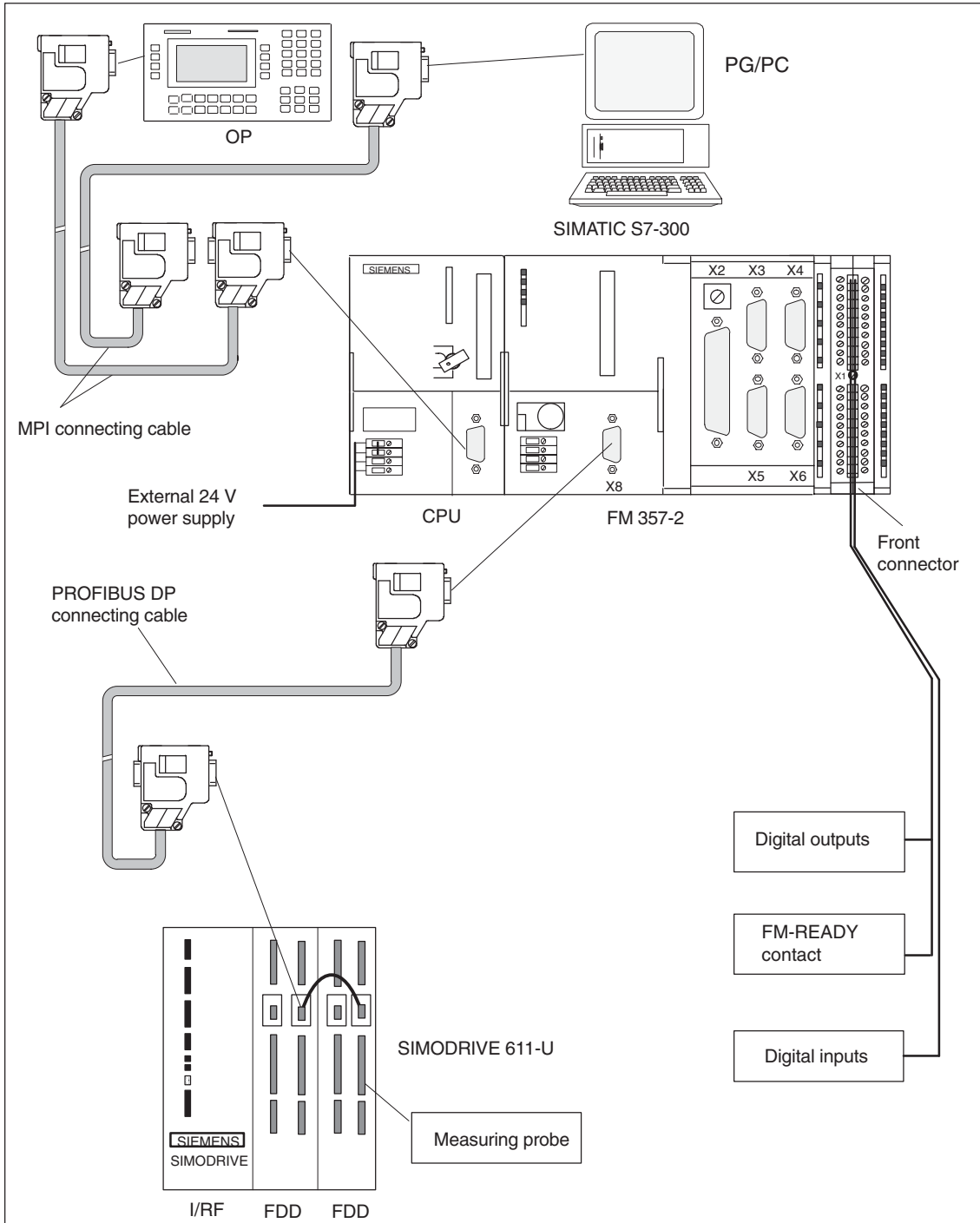


Fig. 4-2 Overview of connecting cable between FM 357-2 and digital servo drive (example)

FM 357-2 with stepper drive

Figure 4-3 shows the interconnections between the individual components of the multi-axis control with FM 357-2 and the stepper drive.

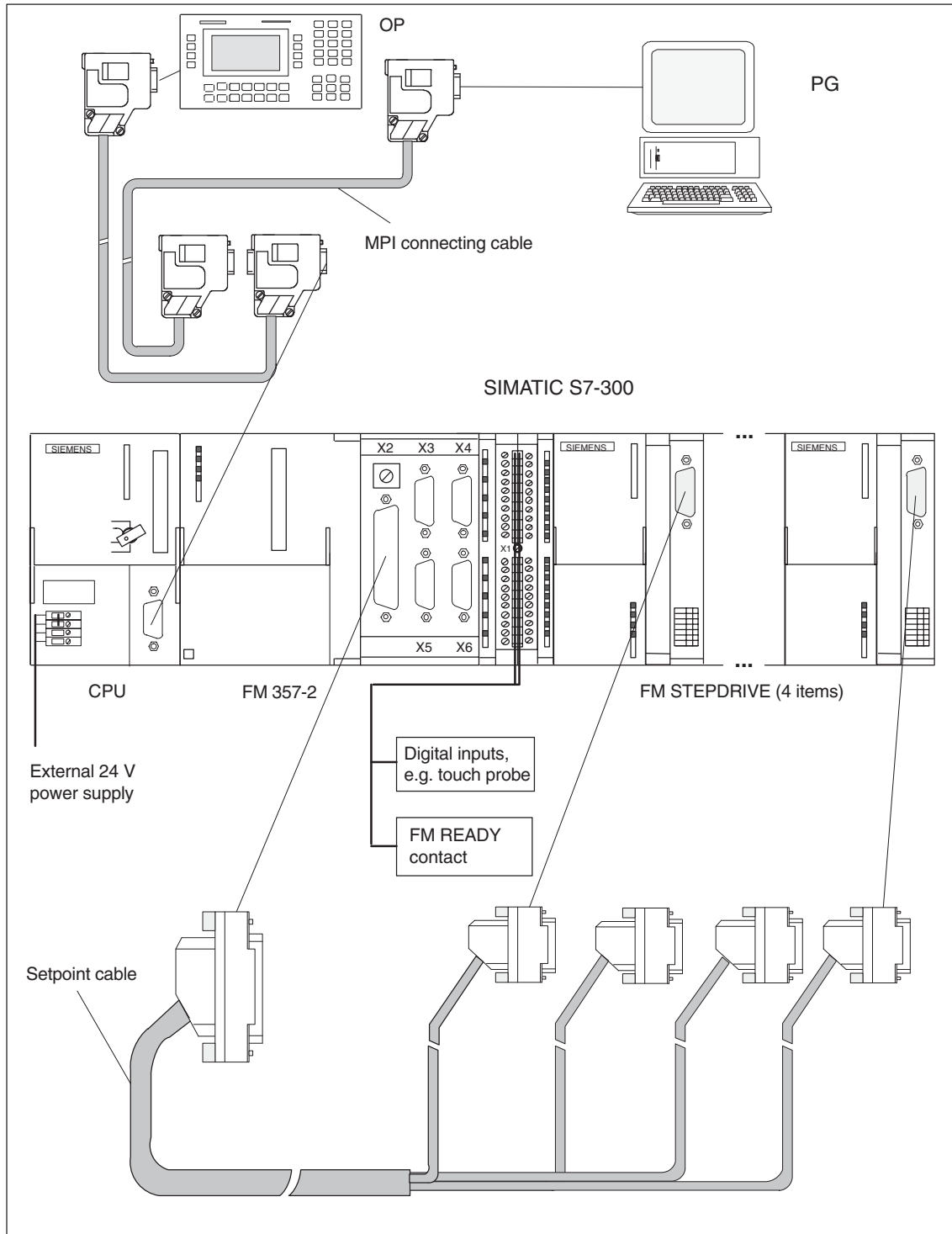


Fig. 4-3 Overview of connecting cable between FM 357-2 and stepper drive (example)

Connecting cable

Table 4-1 lists the connecting cables for a multi-axis control with FM 357-2 .

Table 4-1 Connecting cable for a multi-axis control with FM 357-2

Type	Order No.	Description
MPI connecting cable	See manual <i>S7-300 Programmable Controller, Hardware and Installation</i>	Connection between OP, PG/PC and S7-300 CPU
PROFIBUS DP connecting cable	See manual <i>S7-300 Programmable Controller, Hardware and Installation</i>	Connection between FM 357-2 and SIMODRIVE 611-U (universal)
Setpoint cable	6FX2 002-3AD01-1□□0	Connection between FM 357-2 and SIMODRIVE 611-A ± 10 V
Setpoint cable	6FX5 002-3AE00-1□□0	Connection between FM 357-2 and four stepper drives
Measuring system cable	6FX2 002-2CD01-1□□0	Incremental encoder with RS 422 and FM 357-2 (EXE with linear scale)
Measuring system cables	6FX2 002-2CE01-□□□0	ROD 320 encoder with 1FT5 motor and FM 357-2
Measuring system cables	6FX2 002-2CC11-□□□0	Connection of absolute encoder (SSI) and FM 357-2
Measuring system cables	6FX2 002-2CJ00-1□□0	Connection of SIMODRIVE 611-A control module with 1FK6 motors with resolver and FM 357-2

The setpoint and measuring system cables are available in a variety of lengths.

see *Catalog NC Z* Order No.: E86060-K4490-A001-A□-7600

and/or

Catalog NC 60, Order No.: E86060-K4460-A101-A□-7600

and/or

Catalog ST 70, Order No.: E86060-K4670-A101-A□-7600

Front connector

To wire up the digital inputs/outputs, you will require a 40-way, screw-type front connector. It must be ordered separately.

Order No.: 6ES7 392-1AM00-0AA0

see *Catalog ST 70*, Order No.: E86060-K4670-A101-A□-7600

and/or

Catalog NC 60, Order No.: E86060-K4460-A101-A□-7600

4.2 Connecting the power supply

Screw-type terminal block

The 24 V DC load power supply is wired up to the screw-type terminal block.

Properties of the load power supply

Notice

Please observe the design guidelines for SIMATIC. Make sure, in particular, that the M connection (reference potential) is connected to the device ground of the PLC (M connection on the terminal block of the S7-300-CPU).

see Installation Manual *S7-300 Automation System, Creating*.



Danger

The 24 V d.c. voltage must be designed as a function low–extra voltage with safe electrical isolation.

Notice

Make sure that the interconnecting cable between power supply and load power connection L+ and the relevant reference potential M **does not exceed** a maximum permissible length of 10 m.

The 24 V DC voltage must be generated as a functional extra-low voltage with protective separation.

Table 4-2 Electrical parameters of load power supply

Parameters	min	max	Unit	Conditions
Average voltage range	20.4	28.8	V	
Ripple		3.6	V _{pp}	
Non-periodic overvoltage		35	V	500 ms period 50 s recovery time
Rated power consumption		1	R	
Startup current		2.6	R	

Connector pin assignment

The following table show the connector pin assignments on the screw-type terminal block.

Table 4-3 Assignment of the screw-type terminal

Terminal		
1	L+	24 V DC
2	M	Ground
3	L+	24 V DC
4	M	Ground

Contacts 1/3 and 2/4 are connected internally in the unit.

Notice

The FM 357-2 and the S7-300 CPU should be connected to a common load power supply.

Suitable units include the S7-300 power supply module PS 307 and other SIEMENS load power supplies (e.g. series 6EP1).

In other cases, it is necessary to equalize the potential between the power supplies.

Mains buffering

PS 307 load power supplies for S7-300 guarantee power failure bridging for 20 ms.

Notice

If you use a different load power supply than the PS 307, this must guarantee the required power failure bridging time of 20 ms.

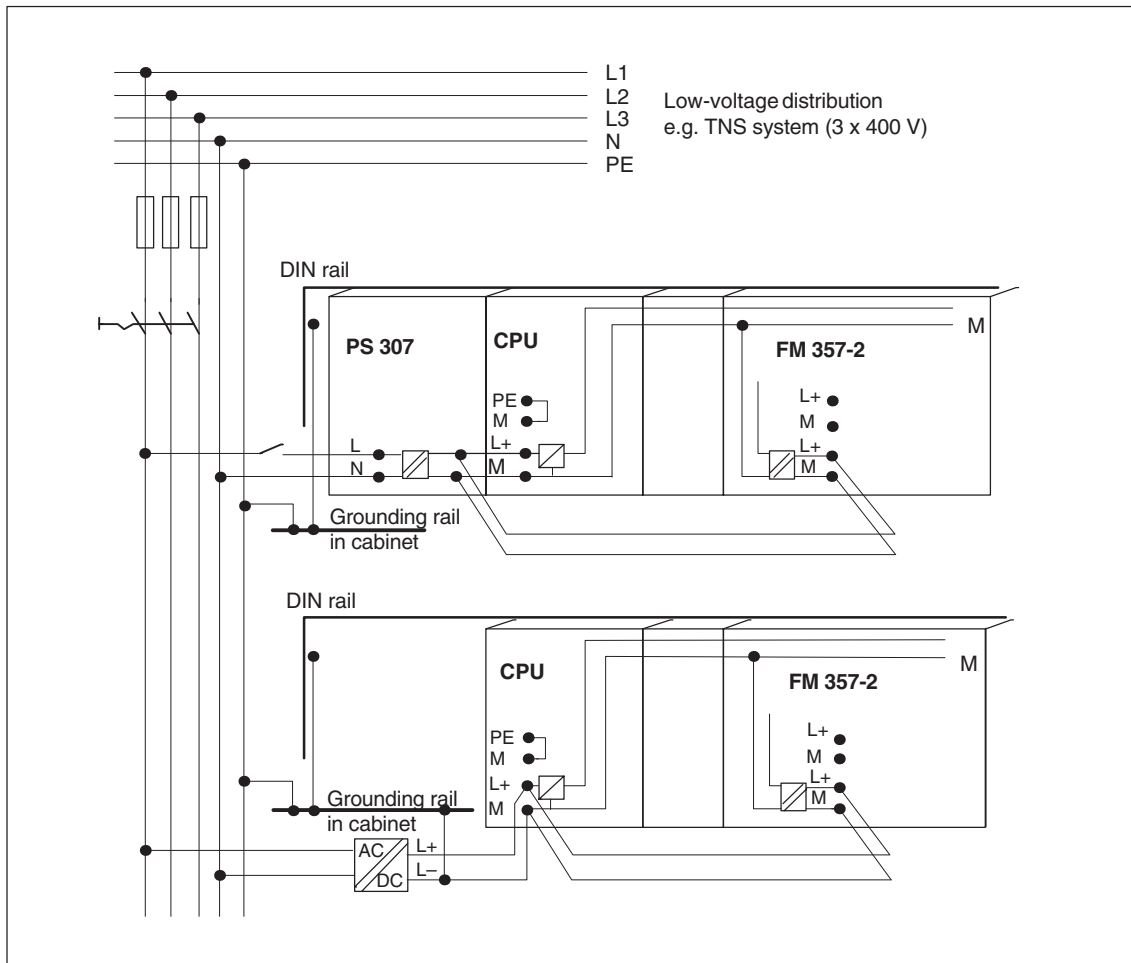


Fig. 4-4 Options for module power supply



Warning

Only wire up the S7-300 when the power is switched off!

Cables

Use flexible cables with a cross-section of between 1.0 and 2.5 mm² (or AWG 18...AWG 14).

Insulation stripping length 12 mm

Ferrules are not necessary.

You can use ferrules without insulating collars per DIN 46228, Shape A, long configuration.

Connecting the power supply

Please proceed as follows:

1. Open the left-hand front cover on the FM 357-2.
2. Connect the flexible cable to the terminals on the screw-type terminal block. **Check** that the polarity is correct.
3. Tighten the cables with a 3.5 mm screwdriver, applying a torque of about 50...80 Ncm.
4. Make the connection to the power supply unit (e.g. PS 307).

Polarity reversal protection

The LED "DC 5V" lights up green when you have made the connection correctly and switched on the power supply.

Notice

Your module will not run if the polarity is reversed. An integrated system protects the electronic circuitry against damage from polarity reversal.

Fuse

The integrated fuse will blow only if the module develops a fault. In this case, it will be necessary to replace the whole module.

4.3 PROFIBUS DP drive interface

Female connector for connecting the FM 357-2 to PROFIBUS DP (for SIMODRIVE 611-U)

You can operate power sections with digital interfaces on the 9-pin sub D socket connector (via PROFIBUS DP).

Position of the socket connector

Figure 4-5 shows the mounting position and designation of the socket connector on the module.

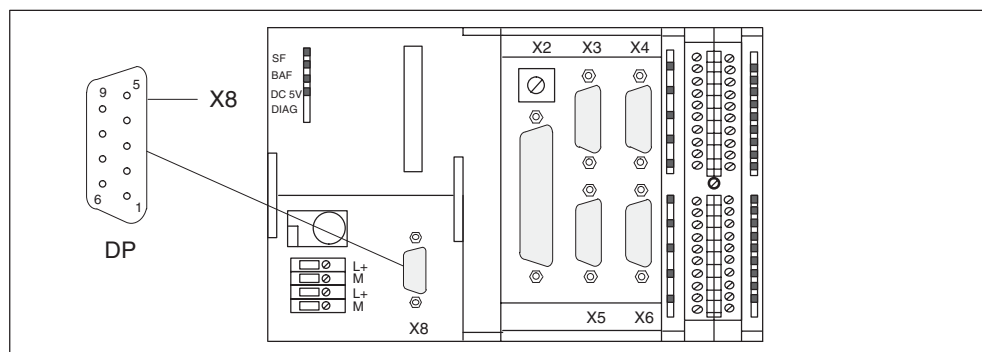


Fig. 4-5 Position of connector X8

Assignment of connector pins

Designation: **X8** DP
 Type: 9-pin female sub-D socket connector

Table 4-4 Pin assignment of socket connector X8

Pin	Name	Type	Pin	Name	Type
1			6	P5	VO
2			7		
3	B	I/O	8	R	I/O
4	RTS	I	9		
5	M	VO			

Signal names

A, B Data I/O (RS 485)
 RTS Request to send
 P5 5 V power supply
 M Ground

Signal type

I	Signal input
I/O	Signal I/O
VO	Voltage outlet

Cable length

up to 100 m (Baud rate 12 Mbaud)

4.4 Description of the drive interface for analog and stepper drives

Connector to drive unit

Power sections with an analog interface (± 10 V) or stepper motor power sections, which have at least a clock pulse and direction input, can be connected to the 50-way sub-D connector X2 on the FM 357-2. Any hybrid configurations are supported for a maximum of four drives.

The FM 357-2 also supplies an enable signal for each axis.

Position of connector

Figure 4-6 shows the mounting position and designation of the connector on the module.

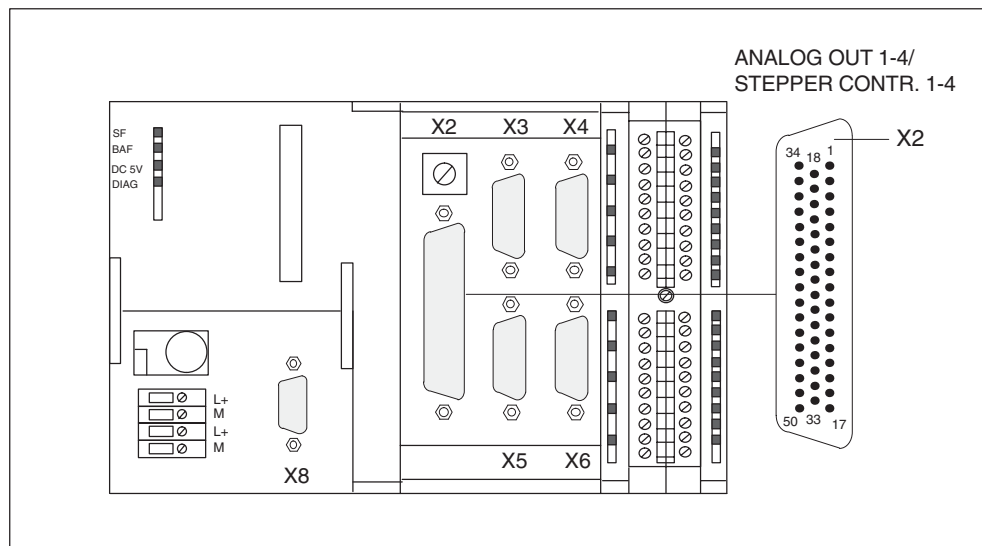


Fig. 4-6 Position of X2 connector

Assignment of connector pins

Drive interface (servo interface for 4 axes)

Connector name: **X2**
ANALOG OUT 1-4/STEPPER CONTR. 1-4
Connector type: 50-pin sub-D plug connector

Table 4-5 Pin assignment of the X2 connector

Pin	Name	Type	Pin	Name	Type	Pin	Name	Type
1	SW1	VO	18	ENABLE1	O	34	BS1	VO
2	BS2	VO	19	ENABLE1_N	O	35	SW2	VO
3	SW3	VO	20	ENABLE2	O	36	BS3	VO
4	BS4	VO	21	ENABLE2_N	O	37	SW4	VO
5	PULSE1	O	22	GND		38	PULSE1_N	O
6	DIR1	O	23	GND		39	DIR1_N	O
7	PULSE2_N	O	24	GND		40	PULSE2	O
8	DIR2_N	O	25	GND		41	DIR2	O
9	PULSE3	O	26	ENABLE3	O	42	PULSE_N	O
10	DIR3	O	27	ENABLE3_N	O	43	DIR3_N	O
11	PULSE4_N	O	28	ENABLE4	O	44	PULSE4	O
12	DIR4_N	O	29	ENABLE4_N	O	45	DIR4	O
13	not assigned		30	not assigned		46	not assigned	
14	RF1.1	K	31	not assigned		47	RF1.2	K
15	RF2.1	K	32	not assigned		48	RF2.2	K
16	RF3.1	K	33	not assigned		49	RF3.2	K
17	RF4.1	K				50	RF4.2	K

Signal names

for stepper drives:

PULSE[1...4], PULSE[1...4]_N Clock pulse true and negated
DIR[1...4], DIR[1...4]_N Direction signal true and negated
ENABLE[1...4], ENABLE[1...4]_N Servo enable true and negated
GND Signal ground

for analog drives:

SW[1...4] Setpoint
BS[1...4] Reference potential for setpoint (analog ground)
RF[1.1...4.1], RF[1.2...4.2] Servo enable contact

Signal type

O	Signal output
VO	Voltage outlet
K	Switching contact

Analog drives

Signals:

One voltage and one enable signal is made available for each axis.

- **SETPOINT (SW)**

An analog voltage signal in the range ± 10 V for output of an rpm setpoint.

- **REFERENCE SIGNAL (BS)**

Reference potential (analog ground) for the setpoint signal; connected internally to logic ground.

- **SERVO ENABLE (RF)**

A relay contact pair is used to switch the axis-specific enables of the power section, for example, of a SIMODRIVE drive unit. After the FM 357-2 has powered up, the RF signal to the drive is set as soon as controller enable RFG (user DB "AXy" DBX2.1+m) is signalled by the user program.

Signal parameters

The setpoint is output as an analog differential signal.

Table 4-6 Electrical parameters of the setpoint signal

Parameters	min	max	Unit
Voltage range	-10.5	10.5	V
Output current	-3	3	mA

Relay contacts

The controller enable signals are switched by relay outputs (NO contacts).

Table 4-7 Electrical parameters of the relay contacts

Parameters	max	Unit
Switching voltage	50	V
Switching current	1	R
Switching capacity	30	VA

Cable length: up to 35 m

Stepper drives

Signals:

A clock pulse, direction and enable signal is made available for each axis as a true and negated signal.

- **PULSE (CLOCK)**

The clock pulses control the motor. The motor executes one increment in response to each rising pulse edge.

This means that the number of pulses which are output determines the angle of rotation, i.e. the distance to be traversed.

The pulse frequency determines the rotational velocity, i.e. the traversing velocity.



Caution

If your drive unit responds to falling clock signal edges, you must swap the true and negated clock signals, otherwise there may be a discrepancy between the position calculated by the controller and the actual position.

- **DIRECTION**

The signal levels which are output determine the direction of rotation of the motor.

Signal ON:	“Rotation to left”
Signal OFF:	“Rotation to right”

Notice

If the direction of rotation of your motor is different, you can use machine data to reverse the direction. Check the technical documentation for your drive device regarding assignment of signal levels to direction of rotation.

- **ENABLE**

The FM 357-2 activates this signal anytime the cyclical control operating mode is detected.

Signal ON:	Power activation is enabled
Signal OFF:	Depending on power section, one or more of the responses mentioned may occur: <ul style="list-style-type: none"> – Disable pulse input – Switch off power to motor – Reset ring counter – Erase error messages

Notice

The ENABLE signal is output at the same time as the servo enable contact (RF). You can therefore use the relay contacts as an alternative.

Signal parameters

All signals for stepper drives are output by way of differential-signal line drivers in compliance with Standard RS 422. To ensure optimum noise immunity, the power section should feature differential signal receivers or optical coupler inputs to permit balanced signal transfer. Unbalanced transfer is also possible, however cable length in such cases is limited to a maximum of 10 m.

Notice

In the case of asymmetrical transmission satisfactory functioning cannot be guaranteed because of the various non-standardized input circuits of the drive units. Especially the lead length and the limit frequency depend on the properties of the input circuit and the lead used. Furthermore, the reference potential GND must be floating in order to prevent electrical interference.

All outputs are electronically protected against shorting and thermal overload.

Figure 4-7 shows various options for connecting signals. Table 4-8 gives a summary of the electrical data of the interface output signals.

Table 4-8 Electrical parameters of the stepper drive signal outputs

Parameters	min	max	Unit	when
Differential output voltage V_{OD}	2		V	$R_L = 100 \Omega$
Output voltage "High" V_{OH}	3.7		V	$I_O = -20 \text{ mA}$
	4.5		V	$I_O = -100 \mu\text{A}$
Output voltage "Low" V_{OL}		1	V	$I_O = 20 \text{ mA}$
Load resistance R_L	55		Ω	
Output current I_O		± 60	mA	
Pulse frequency f_P		750	kHz	

Cable length: up to 50 m
for hybrid systems with analog axes: 35 m
for asymmetrical transfer: 10 m

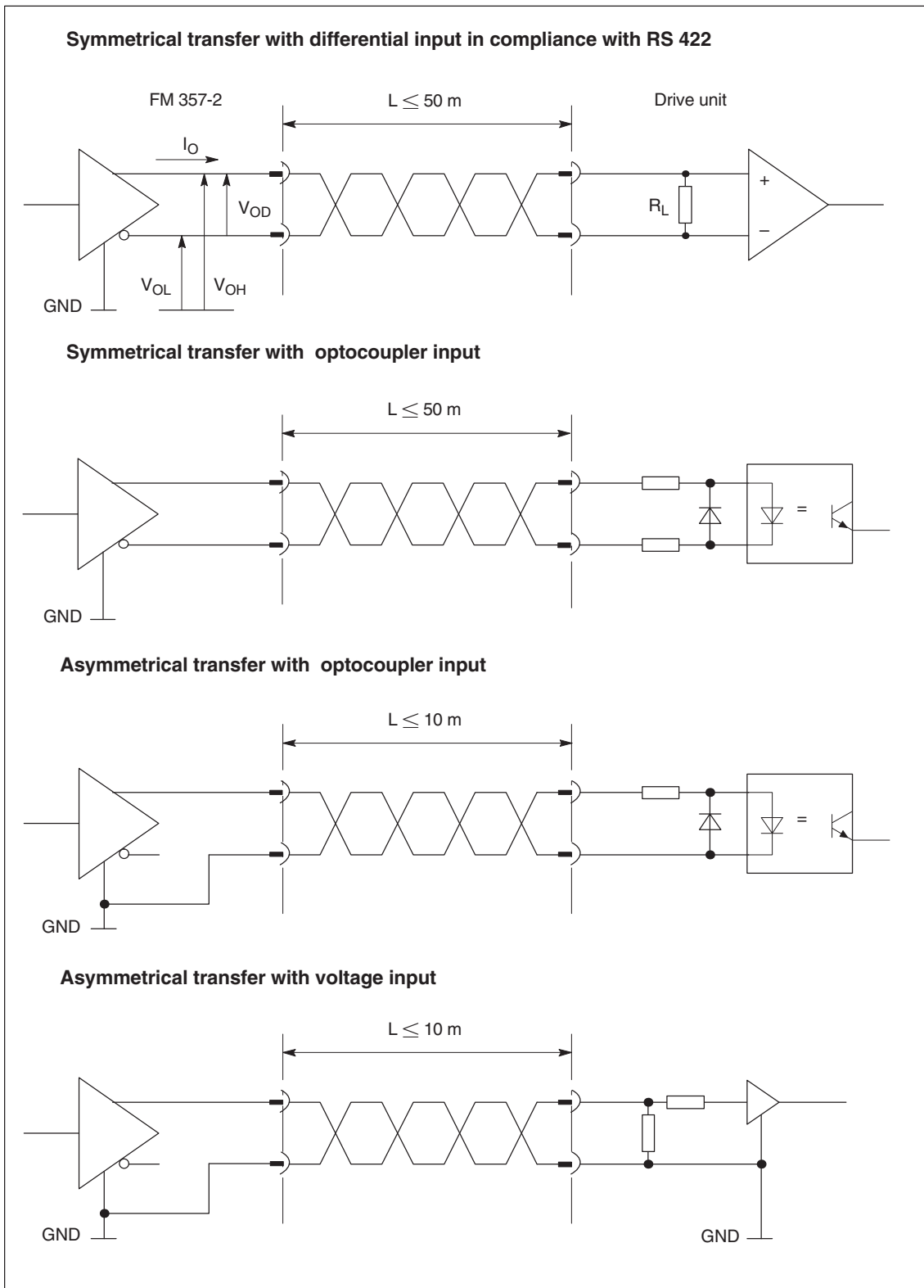


Fig. 4-7 Options of signal circuits for the stepper motor interface

4.5 Connecting the drive units

Connection of the connecting cable

Please note the following:

Notice

Use only twisted pairs for lines. The shielding must be connected to the metallic or metallized connector jacket on the controller side. To protect the analog setpoint signal against low-frequency interference, we recommend that you not ground the shielding on the drive-unit side.

The cable set supplied as an accessory offers excellent immunity against interference.

Connection of analog drives (e.g. SIMODRIVE 611-A)

The following diagram shows you how to connect the FM 357-2 to a SIMODRIVE 611-A drive unit.

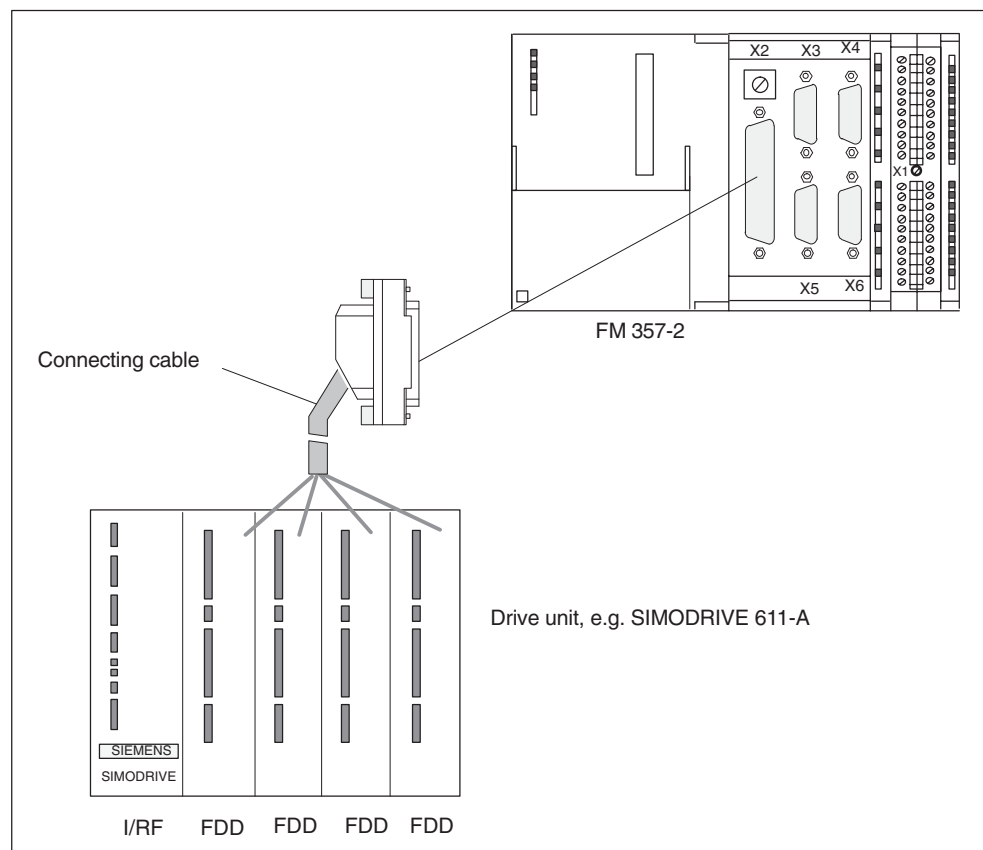


Fig. 4-8 Connecting a SIMODRIVE 611-A drive unit

Please proceed as follows:

1. Connect the free end of the cable to the terminals on the drive unit. (The terminal identifiers on the cable ends indicate the proper terminals for SIMODRIVE units.)
2. Open the front door of the FM 357-2 and connect the sub-D socket connector (50-way) to connector X2.
3. Use the thumbwheel screws to lock the connector in place. Close the front door.

Connecting cable

The connecting cable is a prefabricated cable for four axes with an analog interface, terminal designation for SIMODRIVE drive units.

The connecting cable is available in a variety of lengths.

see *Catalog NC Z* and/or *Catalog NC 60* and/or *Catalog ST 70*

Connection of stepper drives (e.g. FM STEPDRIVE)

The following figure shows you how to connect the FM 357-2 to FM STEPDRIVE drive units.

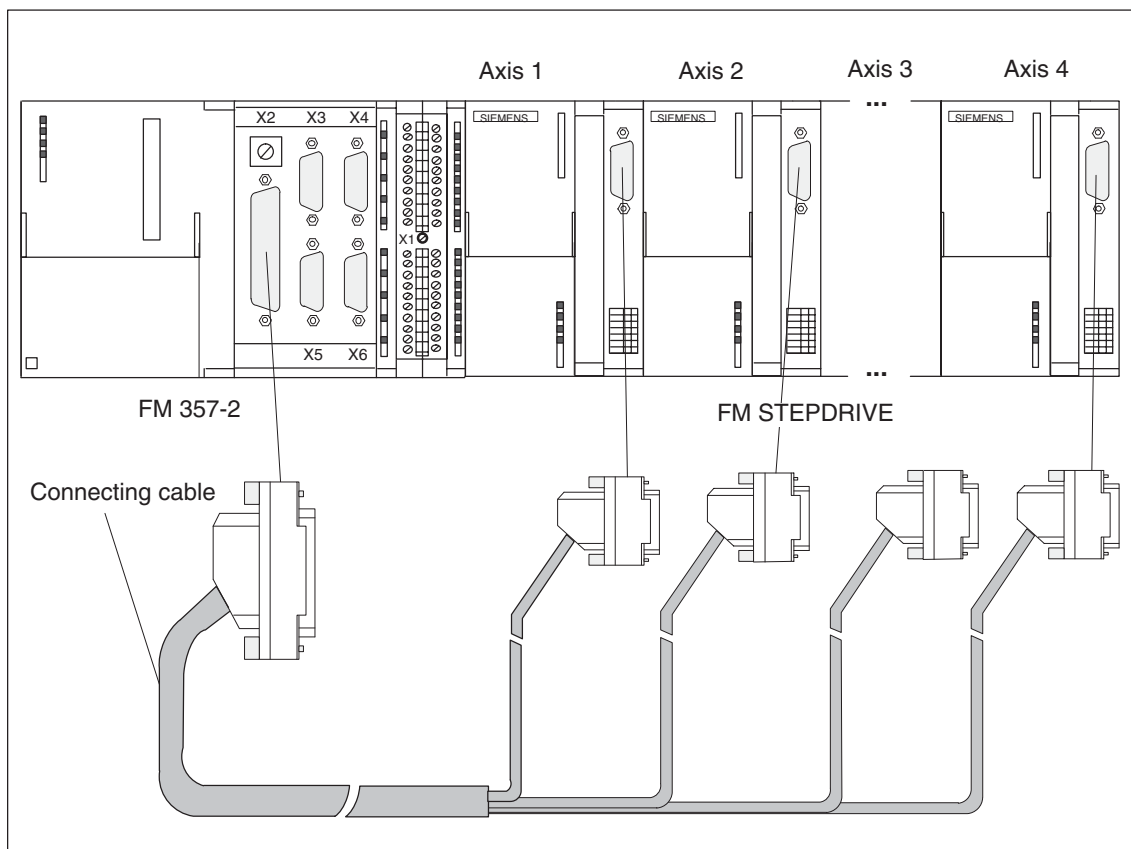


Fig. 4-9 Connection of FM STEPDRIVE drive units

Please proceed as follows:

1. Connect the sub-D connector (15-way) to the FM STEPDRIVE module.
2. Open the front door of the FM 357-2 and connect the sub-D socket connector (50-way) to connector X2.
3. Use the thumbwheel screws to lock the connector in place. Close the front door.

Connecting cable

The connecting cable is a prefabricated cable for four FM STEPDRIVE stepper motor drive units.

see *Catalog NC Z* and/or *Catalog NC 60* and/or *Catalog ST 70*

Connection of digital drives (SIMODRIVE 611-U)

The following diagram shows you how to connect the FM 357-2 to a SIMODRIVE 611-U drive unit.

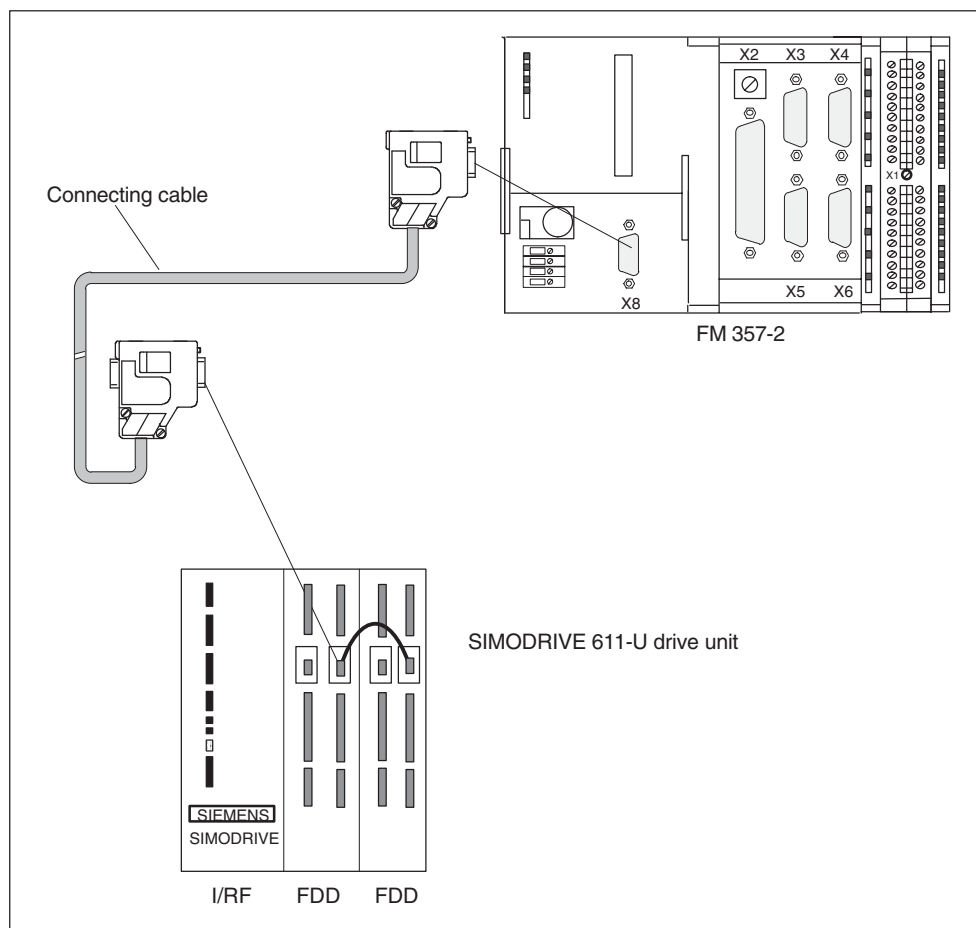


Fig. 4-10 Connecting a SIMODRIVE 611-U drive unit

Please proceed as follows:

1. Plug the male sub-D connector (9-way) into the power section.
2. Open the front door of the FM 357-2 and connect the sub-D socket connector (9-way) to connector X8.
3. Use the thumbwheel screws to lock the connector in place. Close the front door.

Connecting cable

For details on bus cable, bus connector and assembly please refer to the Manual *S7-300 Installation and Hardware, Chapter "Networking"*.

Notice

The maximum cable length is 100 m.

At the beginning and end of a bus segment, you must connect the terminating resistor (switch position "On").

Mixed operation of analog and stepper drives

Details of connecting cables for other configurations are available on request.

Proceed as described for the connection of analog or stepper drives. Whether you install a terminal distributor or wire up the equipment by terminating the connecting cables depends on the design properties of your system.

Notice

Make sure that the signal polarity is correct. Check the connections you have made against the specifications in the technical documentation for your drive unit (e.g. Manual *FM STEPDRIVE, Description of Functions*) and Section 4.4 of the FM 357-2 Manual.

Mixed operation of analog, digital and stepper drives

Analog, digital and stepper drives can be operated in a hybrid configuration.

Proceed as described for the connection of analog, digital or stepper drives.

Setpoint assignment

There is a predefined assignment of setpoints for axes 1 to 4.

Setpoint output signals (X2) for **analog drive**:

- SW1, BS1, RF1.1, RF1.2 for axis 1
- SW2, BS2, RF2.1, RF2.2 for axis 2
- SW3, BS3, RF3.1, RF3.2 for axis 3
- SW4, BS4, RF4.1, RF4.2 for axis 4

Setpoint output signals (X2) for **stepper drive**:

- PULSE1, PULSE1_N, DIR1, DIR1_N, ENABLE1, ENABLE1_N for axis 1
- PULSE2, PULSE2_N, DIR2, DIR2_N, ENABLE2, ENABLE2_N for axis 2
- PULSE3, PULSE3_N, DIR3, DIR3_N, ENABLE3, ENABLE3_N for axis 3
- PULSE4, PULSE4_N, DIR4, DIR4_N, ENABLE4, ENABLE4_N for axis 4

4.6 Description of the measuring system interface

Socket connectors for encoders

A 15-way, sub-D socket connector is provided on each axis for the connection of an incremental or absolute encoder (SSI).

Position of socket connectors

Figure 4-11 shows the mounting position and designation of the socket connector on the module.

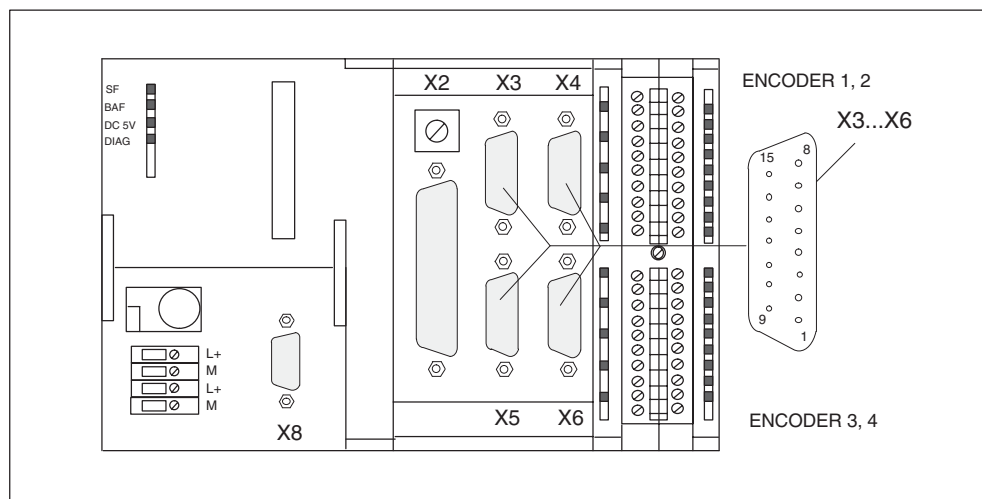


Fig. 4-11 Assignment of connectors X3 to X6

Pin assignment of socket connectors

Designation: **X3, X4, X5, X6** ENCODER 1...4
 X3 Axis 1
 X4 Axis 2
 X5 Axis 3
 X6 Axis 4

Type: 15-pin female sub-D plug connector

Table 4-9 Assignment of connectors X3 to X6

Pin	Encoder		Type	Pin	Encoder		Type
	Incremental	Absolute			Incremental	Absolute	
1	not assigned			9	MEXT		VO
2		CLS	O	10	N		I
3		CLS_N	O	11	N_N		I
4	P5EXT		VO	12	B_N		I
5	P24EXT		VO	13	B		I
6	P5EXT		VO	14	A_N	DATA_N	I
7	MEXT		VO	15	A	DATA	I
8	not assigned						

Signal names

A, A_N Track A true and negated (incremental encoder)
 B, B_N Track B true and negated (incremental encoder)
 N, N_N Zero mark true and negated (incremental encoder)
 CLS, CLS_N SSI sliding pulse true and negated (absolute encoder)
 DATA, DATA_N SSI data true and negated (absolute encoder)
 P5EXT +5 V power supply
 P24EXT +24 V power supply
 MEXT Ground power supply

Signal type

VO Voltage outlet (power supply)
 O Output (5 V signal)
 I Input (5 V signal)

Suitable encoder types

Incremental and absolute encoders

Incremental encoders with 5 V square-wave signals or absolute encoders with a synchronous serial interface (SSI) can be connected directly (e.g. digital rotary encoders); they are then selected via machine data.

Encoders with SINE/COSINE signals (e.g. length scales) may be connected by way of an external electronic pulse shaper (EXE) that converts the signals to 5 V levels.

Resolver

It is not possible to connect a resolver directly. However, you can use motors with resolvers if you use a SIMODRIVE control module for resolvers. This converts the resolver signals into incremental encoder signals.

Encoder properties

Encoders for direct connection (or EXEs) must fulfil the following conditions:

Incremental Encoders

Transfer procedure:	Differential transfer with 5 V square-wave signals (as with RS422 standard)
Output signals:	Track A as true and negated signal (U_{a1} , $\overline{U_{a1}}$) Track B as true and negated signal (U_{a2} , $\overline{U_{a2}}$) Zero signal N as true and negated signal (U_{a0} , $\overline{U_{a0}}$) When connecting an incremental encoder, please note that, at the time of the zero pulse (true signals), the signals of tracks A and B must also be "true". If necessary, the negated signal must be wired and the direction modified ("Actual value direction reversal" parameter).
Maximum output frequency:	1 MHz
Phase shift, track A to B:	$90^\circ \pm 30^\circ$
Power consumption:	Not more than 300 mA

Absolute Encoders (SSI)

Transfer procedure:	Synchronous serial interface (SSI) with 5 V differential signal transfer (as with RS422 standard)
Output signals:	Data as true and negated signal
Input signals:	Sliding pulse as true and negated signal
Resolution:	Not more than 25 bits
Maximum transfer frequency:	1.5 Mbits/s
Power consumption:	Not more than 300 mA

5 V encoder power supply

The 5 V supply voltage for encoders is generated in the module and is available at the sub-D socket connector. It can therefore supply the encoders via the connecting cable without any additional wiring. The available voltage is electronically protected against shorting and thermal overload, and is monitored.

Notice

Please note that the maximum current which can be drawn from the 5 V supply (P5EXT ports) must not exceed 1.35 A for all encoders connected!

24 V encoder supply

For encoders with a 24 V operating voltage, the 24 V DC power supply is distributed among the sub-D socket connectors so that the encoders can be supplied via the connecting cable without any additional wiring. The available voltage is electronically protected against shorting and thermal overload, and is monitored.

Notice

Please note that the maximum current which can be drawn from the 24 V supply must not exceed 1 A for all encoders connected!

Table 4-10 Electrical parameters of encoder power supply

Parameters	min	max	Unit
5 V power supply			
Voltage	5.1	5.3	V
Ripple		50	mV _{pp}
Current carrying capacity per encoder		0.3	R
Max. current carrying capacity		1.35	R
24 V power supply			
Voltage	20.4	28.8	V
Ripple		3.6	V _{pp}
Current carrying capacity per encoder		0.3	R
Max. current carrying capacity		1	R

Connecting cable to encoder

The maximum permissible cable length depends on the encoder supply specification and the transmission frequency. In order to ensure reliable operation, the values below must not be exceeded when using terminated connecting cables from SIEMENS, see *Catalogs NC Z/NC 60/ST 70*):

Table 4-11 Maximum cable length as a function of encoder power supply

Supply voltage	Tolerance	Power consumption	Max. cable length
5 V DC	4.75 V...5.25 V	≤ 300 mA	25 m
5 V DC	4.75 V...5.25 V	≤ 220 mA	35 m
24 V DC	20.4 V...28.8 V	≤ 300 mA	100 m
24 V DC	11 V...30 V	≤ 300 mA	300 m

Notice

If you want to use incremental encoders with cable lengths longer than 25 or 35 m, select a type that uses a 24 V power supply.

Table 4-12 Maximum cable length as a function of transfer frequency

Encoder type	Frequency	Max. cable length
Incremental encoder	1 MHz	10 m
	500 kHz	35 m
Absolute encoder (SSI)	1.5 Mbps	10 m
	187.5 kbps	250 m

4.7 Connecting the encoders

Connection of the connecting cable

Please note the following:

Notice

Use only shielded cables. The shielding must be connected to the metallic or metallized connector jacket.

The cable sets supplied as an accessory offer excellent immunity from interference, as well as cross-sections large enough for the power supply to the encoders.

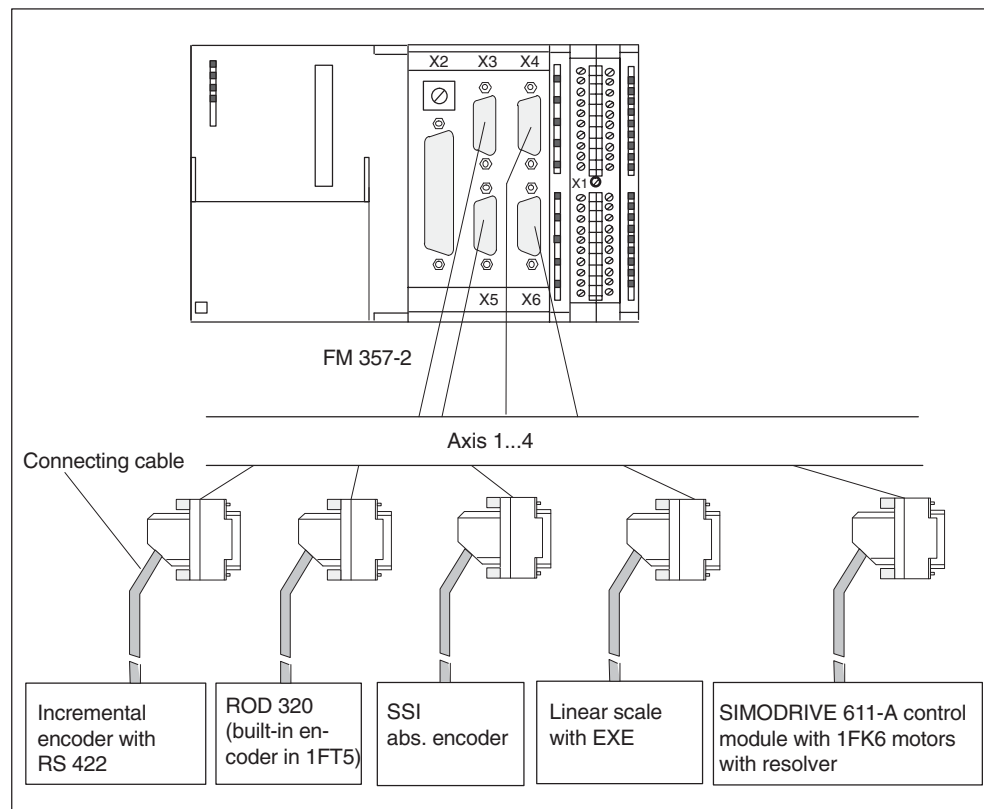


Fig. 4-12 Connection of encoders

Procedure for connecting encoders

Please proceed as follows to connect the encoders:

1. Attach the connecting cable to the encoders.
2. Open the front door of the FM 357-2 and connect the sub-D plug connectors (15-way) to socket connectors X3...X6.
3. Use the thumbwheel screws to lock the connector in place. Close the front door.

Available connecting cables for encoders

The following connecting cables are available:

- Terminated cable for add-on encoders or EXEs (for connection of linear scales)
- Cable set for built-in encoders with 17-pin round plugs.
- Cable set for absolute encoders (SSI)
- Cable set for SIMODRIVE 611-A control module with 1FK6 motors with resolver

Connecting cables are available in a variety of lengths.

see *Catalog NC Z* and/or *Catalog NC 60* and/or *Catalog ST 70*

Actual value assignment

There is a predefined assignment of actual values for axes 1 to 4.

- The encoder for axis 1 must be connected to actual value input X3
- The encoder for axis 2 must be connected to actual value input X4
- The encoder for axis 3 must be connected to actual value input X5
- The encoder for axis 4 must be connected to actual value input X6

4.8 Description of the I/O interface

Front connector

Probes, BEROs or other signal encoders can be connected to the 40-way front connector X1 with discrete connection.

A ready signal is also provided. This must be integrated in the emergency stop equipment.

Position of connector

Figure 4-13 shows the position of the front connector.

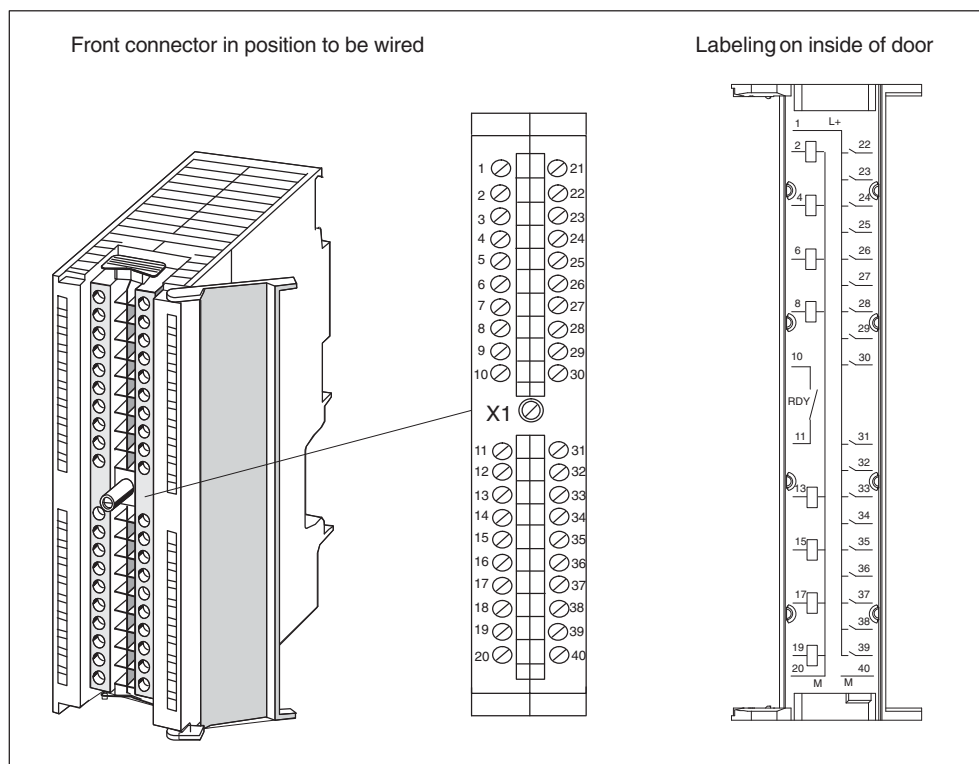


Fig. 4-13 Location of X1 connector

Assignment of connector pins

Connector name: **X1**
 Connector type: 40-pin S7 front connector for single-wire terminal

Table 4-13 Pin assignment of the X1 connector

Pin	Name	Type	Pin	Name	Type
1	L+	VI	21	not assigned	VI
2	Q0	DO	22	BERO1	DI
3	not assigned		23	BERO2	DI
4	Q1	DO	24	BERO3	DI
5	not assigned		25	BERO4	DI
6	Q2	DO	26	MEPU1	DI
7	not assigned		27	MEPU2	DI
8	Q3	DO	28	I0	DI
9	not assigned		29	I1	DI
10	RDY.1	K	30	I2	DI
11	RDY.2	K	31	I3	DI
12	not assigned		32	I4	DI
13	Q4	DO	33	I5	DI
14	not assigned		34	I6	DI
15	Q5	DO	35	I7	DI
16	not assigned		36	I8	DI
17	Q6	DO	37	I9	DI
18	not assigned		38	I10	DI
19	Q7	DO	39	I11	DI
20	M _{output}	VI	40	M _{input}	VI

Signal names

RDY.1...2	Ready (FM-READY contact 1...2)
BERO1...BERO4	BERO input for axis 1...4
MEPU1, MEPU2	Measurement pulse input 1 and 2
I0...I11	Digital inputs 0...11
Q0...Q7	Digital outputs 0...7
L+	Power supply for digital outputs
M _{output}	Power supply for digital inputs
M _{input}	Power supply for digital outputs

Signal type

DI	Digital input (24 V signal)
DO	Digital output (24 V signal)
K	Switch contact
VI	Voltage input

18 digital inputs including 2 probes and 4 BERO proximity switches

These fast inputs (on-board) are PLC-compatible (24 V current-sourcing). Switches or contactless sensors (2-wire or 3-wire sensors) can be connected.

They can be used:

- as switches for reference point approach (BERO1...BERO4), the inputs are permanently assigned to axes 1 to 4 (applies only to stepper motor, no RPS).
- as sensing probes (MEPU1, 2); the axis assignment is programmed
- as free inputs (I0...I11); system variables are used for access in the NC program

Table 4-14 Electrical parameters of digital inputs

Parameters	Value	Unit	Comment
1 signal, voltage range	11 to 30	V	
1 signal, power consumption	6 to 30	mA	
0 signal, voltage range	-3 to 5	V	or input open
Signal delay 0 → 1	15	μs	
Signal delay 1 → 0	150	μs	

8 digital outputs (Q0...Q7)

These high-speed outputs (on-board) are PLC-compatible (24 V current-sourcing). System variables are used for access in the NC program.

Table 4-15 Electrical parameters of digital outputs

Parameters	Value/Unit
Supply voltage (L+)	DC 24 V (permissible range: 20.4...28.8 V)
1 signal, voltage range	$V_L^{1)}$ – 3... $V_L^{1)}$ V
1 signal, max. output current	
• at 40 °C ambient temperature	
– rated value	0.5 A
– permitted range	5 mA...0.6 A (via supply voltage)
– lamp load	max. 5 W
• at 55 °C ambient temperature	
– rated value	0.25 A
– permitted range	5 mA...0.3 A (via supply voltage)
– lamp load	max. 2.5 W
0 signal, max. leakage current	2 mA
Signal delay 0 → 1	500 ms
Signal delay 1 → 0	500 ms

1) V_L – Supply voltage of outputs



Danger

The 24 V supply voltage must be designed as a functional low–extra voltage with safe electrical isolation to EN60204-1, Section 6.4, PELV (with M grounding).

Notice

Make sure that the interconnecting cable between power supply and load power connection L+ and the relevant reference potential M **does not exceed** a maximum permissible length of 10 m.

FM READY output (RDY)

Ready signal as a floating relay contact (normally-open); must be connected to the emergency stop circuit.

Table 4-16 Electrical parameters of relay contact RDY

Parameters	max	Unit
DC switching voltage	50	V
Switching current	1	R
Switching capacity	30	VA

4.9 Wiring up the front connector

Wiring up the front connector

Figure 4-14 shows you how cables are routed to the front connector and the interference suppression provided by the shield connecting element.

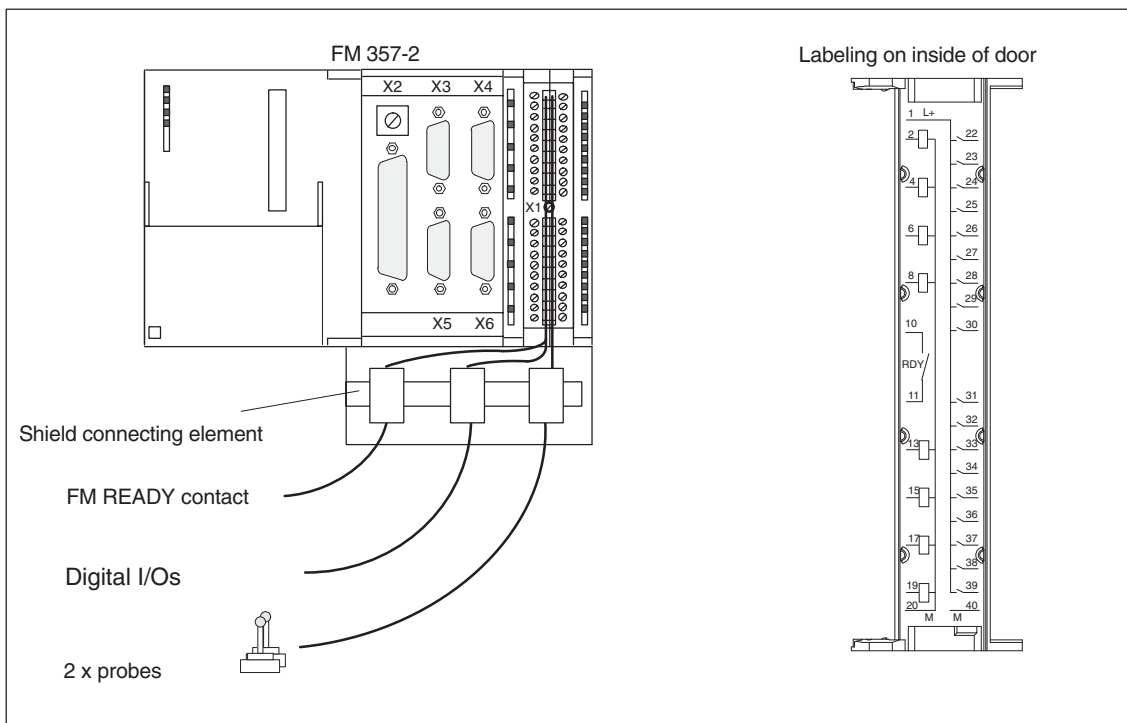


Fig. 4-14 Wiring up the front connector

Connecting cables

Flexible cable, cross-section 0.25...1.5 mm²

Ferrules are not necessary.

You can use ferrules without insulating collars per DIN 46228, Shape A, long configuration.

You can connect two lines measuring 0.25...0.75 mm² in a single ferrule.

Notice

To provide optimum immunity to interference, shielded cables should be used to connect touch probes or sensors.

Tool required

3.5 mm screwdriver or drill

Procedure for wiring front connector

Please proceed as follows to wire up the terminal strip:

1. Remove insulation from 6 mm of cable, press on ferrule if necessary.
2. Open front door, place front connector in wiring position.
Push the front connector into the module until it clicks into place. In this position, the front connector still protrudes from the module.
Lock the connector in place without any electrical contact to the module.
3. If you intend to bring the cables out underneath, start wiring underneath or otherwise at the top. Screw down unused terminals as well.
The tightening torque should be 40...70 Ncm.
4. Wrap the cable strain relief around the cable and the front connector.
5. Tighten the cable strain relief for the cable strand. Press the strain relief lock in to the left to make better use of the cable stowing area.
6. Bring the front connector into the operating position by tightening the fastening screw.
Note: When the front connector is brought into the operating position, a front connector coding key slots into the front connector. The front connector then only fits this type of module.
7. Close the front door.
8. You can fill out the enclosed labelling strip and insert it in the front door.

You can find a detailed description of how to wire up a front connector in the Installation Manual *S7-300 Programmable Controller, Hardware and Installation*.

Shielded cables

If you are using a shielded cable, please take the following additional measures:

1. At the point of cable entry into the cubicle, the cable shield must be attached to an earthed shielding bus (cable insulation must be removed first).

For this you can use the shield connecting element mounted on the DIN rail; it will accept up to eight shielding terminals.

Please refer to the manual *S7-300 Programmable Controller, Hardware and Installation*.

2. Route shielded cable up to the module, but do not make a shield connection at this point.

Shield connecting element

This element can be inserted in the mounting rail for terminating the shields on shielded cables. It can accept up to eight shielding terminals (KLBÜ line from Weidmüller).

see *S7-300 Programmable Controller, Hardware and Installation*, (Wiring section)

Connection of probes or proximity sensors (BEROs)

Please proceed as follows:

1. Wire up the power supply for your sensors. This must meet the same conditions as the load power supply of the FM 357-2. You can use the load power supply terminals of the FM 357-2.
2. Connect the shielded signal cable to the sensors.
3. Remove enough of the cable sheath at the control end to allow you to attach the shield to the shield connecting element and wire up the free cable ends to the front connector.
4. Wire up the signal cable to the front connector.

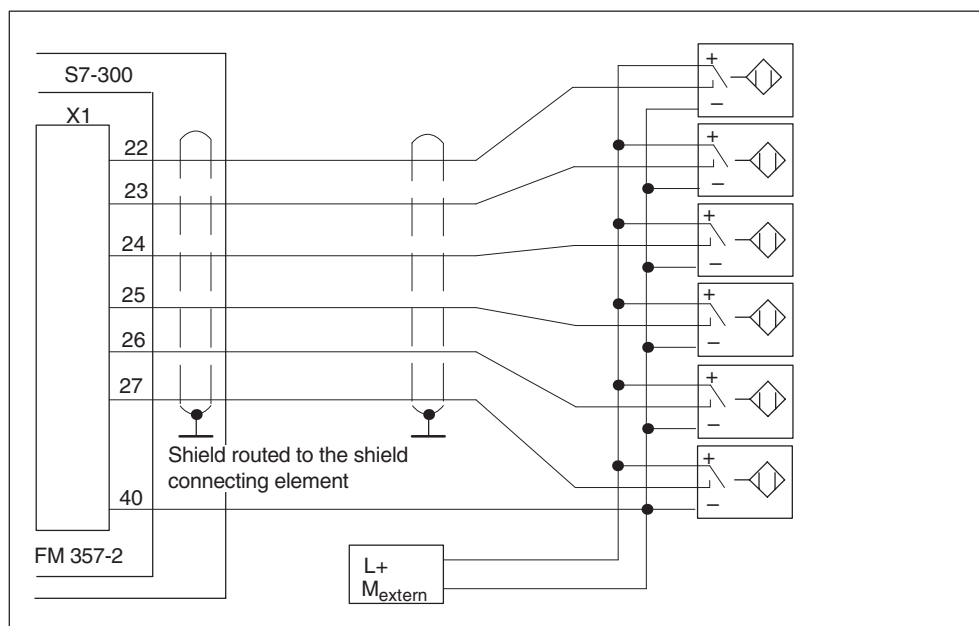


Fig. 4-15 Connection overview for probes and proximity switches

Connection of FM READY contact

When the FM READY contact is opened, the EMERGENCY STOP device is actuated.

Connection of other actuators/sensors

If you wish to connect other actuators/sensors to the SMs on the local bus, please follow the instructions for connecting digital inputs/outputs to the SIMATIC S7-300, which apply analogously.

See *S7-300 Programmable Controller, Hardware and Installation*, and *S7-300, M7-300 Module Specifications*.

4.10 Inserting and replacing the backup battery

General

The FM 357-2 is provided with a backup battery for the supply of emergency power to the RAM.

Before the controller is started up, the supplied Li battery must be inserted in the battery compartment of the FM 357-2.

Inserting the battery

Please proceed as follows:

1. Open the left-hand front door on the FM 357-2.
2. Insert the battery connector into the socket in the battery compartment.

Please make sure that the battery is connected correctly (the notch on the connector must point to the right).

3. Place the battery in the compartment and close the front door.

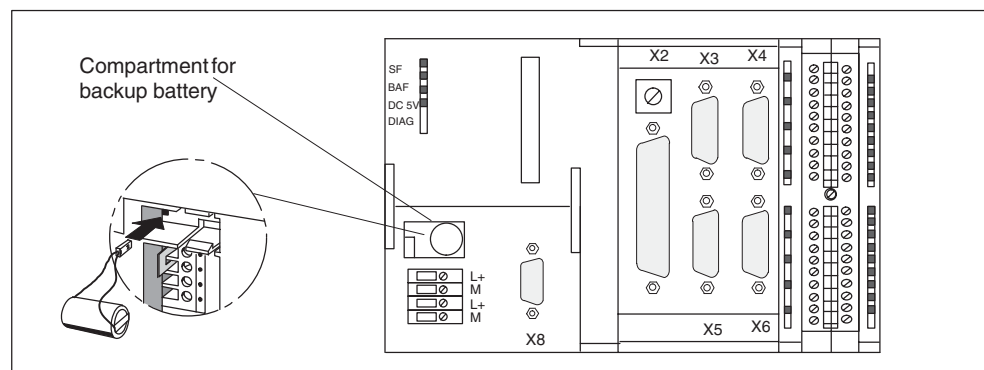


Fig. 4-16 Installing the backup battery

A battery fault is indicated if the battery has been connected up incorrectly.

Notice

A battery which is incorrectly connected can discharge and become unusable.

Replacing the battery

The battery must be replaced when the corresponding error message appears. The “BAF” and “DIAG” LEDs also indicate the status of the battery power and the backed up memory.

The battery can be operated for at least two years without maintenance. Depending on the operating status, you may need to change it after five years or more.

Since the battery properties deteriorate with increasing age, we recommend you change it after five years at the most.

LED “DIAG” flashes (0.5 Hz)

The buffered data are still stored, but the battery is beginning to discharge. It is necessary to replace the battery.

LED “BAF” lights up steadily

The buffered data have been lost. A new start-up will need to be performed after replacement of the battery. The FM 357-2 forces this status.

Notice

You must always leave the load power supply switched on when replacing the battery, otherwise the backup data will be lost!

Battery type

Prefabricated batteries with connectors must be used.

Order No.: 6ES7-971-1AA00-0AA0

Storage of backup batteries

Store the backup batteries in a cool and dry place.

Backup batteries can be stored for 5 years.

Rules for handling backup batteries

Please note the following:



Caution

Improper handling of backup batteries can cause ignition, explosion and burns. You must therefore follow the rules below:

Backup batteries must not

- be recharged
- be heated or burnt
- be pierced or crushed
- be manipulated mechanically or electrically in any other way

Disposal

Please observe your national specifications/guidelines when disposing of backup batteries.



Parameterization

5

Section overview

Section	Title	Page
5.1	Installation of "Parameterize FM 357-2"	5-3
5.2	Configuring	5-5
5.3	Introduction to "Parameterize FM 357-2"	5-19
5.4	Adaptation to firmware	5-20
5.5	Parameterization data	5-21
5.6	Settings of parameterization interface	5-39

General

This chapter gives you an overview of how to define the parameters of the FM 357-2 with the “Parameterize FM 357-2” tool.

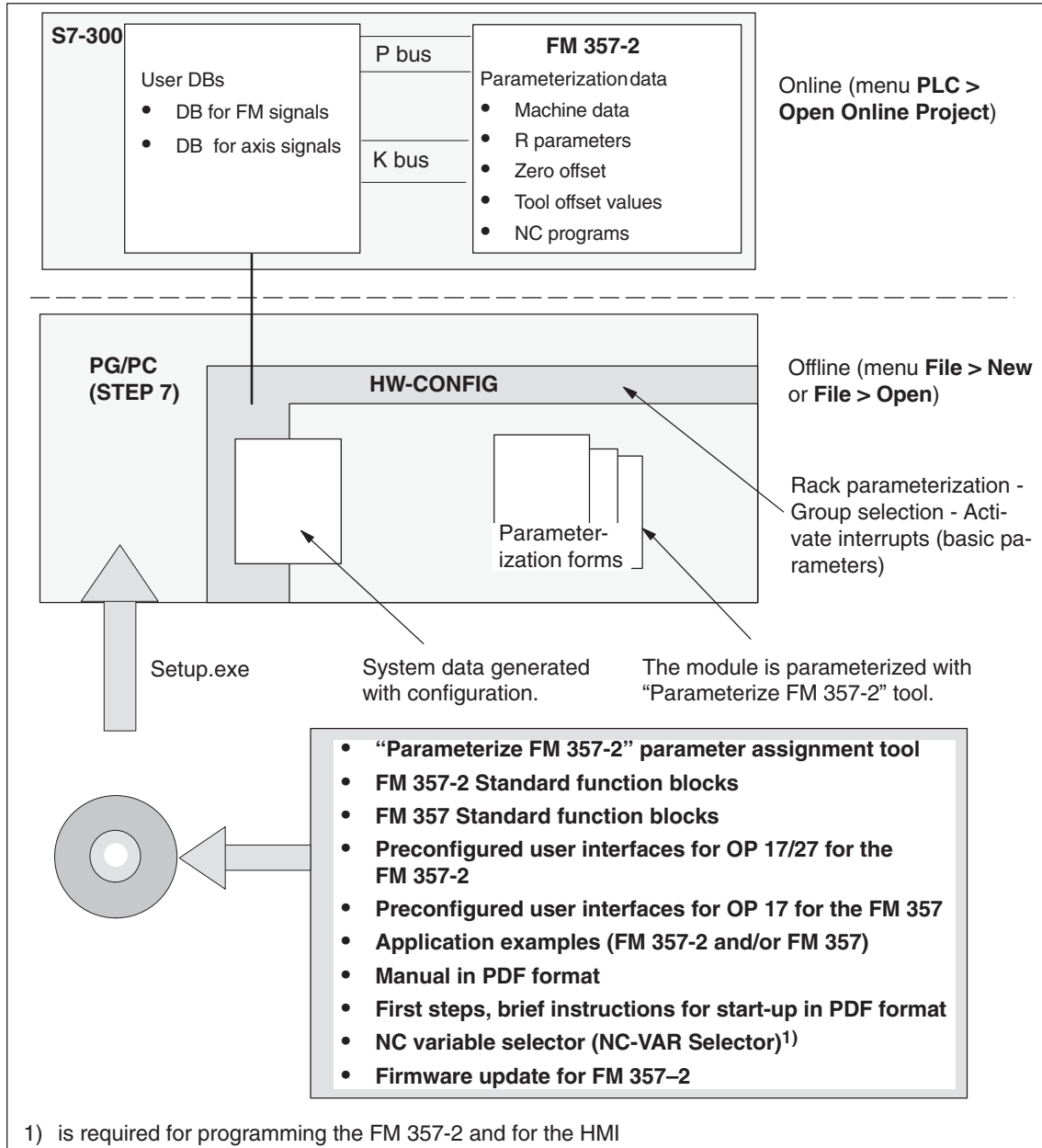


Fig. 5-1 Overview of parameterization

5.1 Installation of “Parameterize FM 357-2”

Precondition

One of the following operating systems must be installed on the programming device (PG/PC):

- “MS Windows XP Professional SP2 or SP3“
- “MS Windows Server 2003 R2 SP2 standard edition as workstation computer“
- ”MS Windows 7 32 Bit Ultimate, Professional or Enterprise“
- ”MS Windows 7 64 Bit Ultimate, Professional or Enterprise“

You need also the corresponding STEP 7 program (from V5.5 + HF4)

For online operation, the link between the PG/PC and the S7-300 CPU must already be set up (see Figures 4-1 and 4-3).

Installation

The entire software (parameterization tool, function blocks, preconfigured interface for OPs and NC-VAR Selector) is stored on CD ROM.

Install the software as follows:

1. Insert the CD ROM in the CD ROM drive of your PG/PC.
2. Run file **Setup.exe** on the CD ROM.
3. Follow the instructions displayed by the installation routine step for step.

In the “FM 357-2 Toolset: Components” dialog, you can select the components you need for your application (see the compatibility list in the “Preface” chapter).

Result: The software is installed in the following directories by default:

- “Parameterize FM 357-2” parameter assignment tool: **[STEP7 directory]\S7FM357**
- FM 357-2 function blocks: **[STEP7 directory]\S7LIBS\FM357_2L**
- FM 357 function blocks: **[STEP7 directory]\S7LIBS\FM357_LI**
- FM 357-2 application examples: **[STEP7 directory]\EXAMPLES\zEn16_01**
- FM 357 application examples: **[STEP7 directory]\EXAMPLES\zEn15_01**
- Sample user interfaces for FM 357-2 (OP 17, OP 27, TP 170 B, MP 270 B): **[STEP7 directory]\EXAMPLES\FM357-2\zdt16_01_FM357-2_OP_EX**
- Sample user interface for FM 357 (OP 17): **[STEP7 directory]\EXAMPLES\FM357\zdt15_01_FM357_OP_EX**
- NC-VAR Selector: **[nc_var-directory]**
- Firmware update for FM 357-2: **[STEP7 directory]\S7fw357**

Notice

If you already have an earlier version of “Parameterize FM 357-2” installed, you must **deinstall** it. Deinstallation is absolutely essential before you install a new version.

5.2 Configuring

General

The term 'configuring a system/installation' comprises the arrangement of sub-racks, modules and distributed I/O modules in a station window in the *SIMATIC Manager HW Config*.

STEP 7 will assign each module an address in a configuration table automatically.

For further information on configuring modules, please refer to your User's Guide *Basic Software for S7 and M7, STEP 7*. In the following, only the most important steps are explained.

Sequence (without PROFIBUS-DP drive)

1. Start the *SIMATIC Manager* and create a new project.
2. Insert a **SIMATIC 300 station** using the **Paste > Station** menu.
3. Select the **SIMATIC 300 station**. The **Edit > Open object** menu will grant you access to the S7 hardware configuration.
4. Select either **Insert > Hardware components** or **View > Catalog** to open the hardware catalog. Select **SIMATIC 300 > RACK-300 > Mounting channel** from this catalog. Draw the mounting channel into your hardware project.
5. Select the CPU (analogously to the selection of the mounting channel) and the FM 357-2 multi-axis module with the appropriate order numbers ("MLFBs") from the hardware catalog and draw them to the desired slot.

The following modules are offered for the FM357-2:

Table 5-1 FM 357-2 modules

Module	MLFB / Note	Remark
FM 357 4AxisControl	6ES7 357-4AH00-0AE0 4-axis motion control	up to software release 02.04
FM 357-2 4AxisControl	6ES7 357-4AH01-0AE0 4-axis motion control	up to software release 03.04
FM 357-2 4AxisControl	6ES7 357-4AH01-0AE0 4-axis motion control, also for distributed use (consistent data)	with software release 05.01 and higher Use: centralized and distributed (in the distributed use: consistent data transfer) No reloadable SDBs and no routing to the PROFIBUS-DP drive
FM 357-2 4AxisControl DP	6ES7 357-4AH01-0AE0 Only for centralized use with routing to the PROFIBUS-DP drive	with software release 05.02 and higher Use: only centralized Reloadable SDBs and routing to the PROFIBUS-DP drive

The old modules (prior to software release 05.01) will remain in the Catalog; thus, existing projects can be processed. When creating a new project, use the modules with software release 05.01 or 05.02 and higher.

6. Save and compile the created hardware project using the menu command **Station > Save and compile.**

Now, use the SIMATIC Manager to enter the configured CPU and the FM 357-2 in your project.

Notice

A PROFIBUS-DP drive in conjunction with fixed (internal) SDBs need not to be configured in SIMATIC S7 HW Config; it merely needs to be configured in the FM (Section 9.1.2).

5.2.1 Sample project with PROFIBUS-DP drive

General

A project for a PROFIBUS-DP drive SIMODRIVE 611 universal (double-axis module) is to be created. Using SIMATIC S7 HW Config, SDBs are created for the routing and for the PROFIBUS-DP drive and be loaded into the FM357-2.

Prerequisites

STEP 7 V5.2 and DRIVE ES V5.1 SP2 is installed.

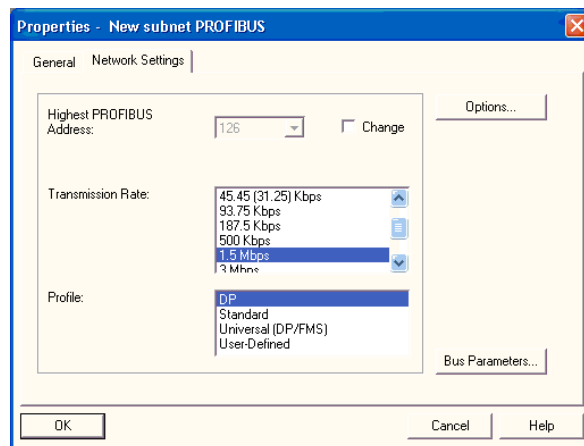
Knowledge of the PROFIBUS-DP communication SIMODRIVE 611 universal (Documentation: Description of Functions SIMODRIVE 611 universal, order no.: 6SN1197-0AB20-0AP[edition]).

Sequence

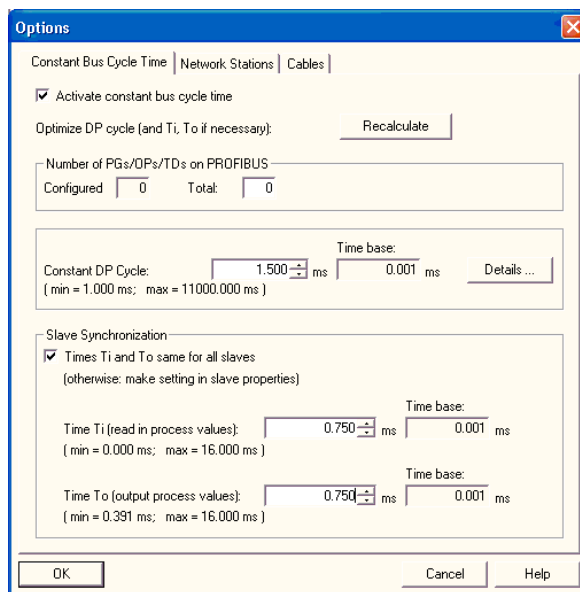
1. Create a project as described in Section 5.2.
2. Select the **FM357-2 4AxisControl DP**. In the dialog box **Properties – PROFIBUS DP interface**, use the **New...** button to open the dialog box **Properties – New PROFIBUS subnet**.

Set the following on the **Network settings** tab:

- Data transfer rate: 12 Mbit/s
- Profile: DP



3. Click on the **Options** button and select the following from the **Options** dialog box:
 - Activate equidistant bus cycle: yes
 - Equidistant DP cycle: 1.5 ms
 - Times T_i and T_o for all slaves the same: yes

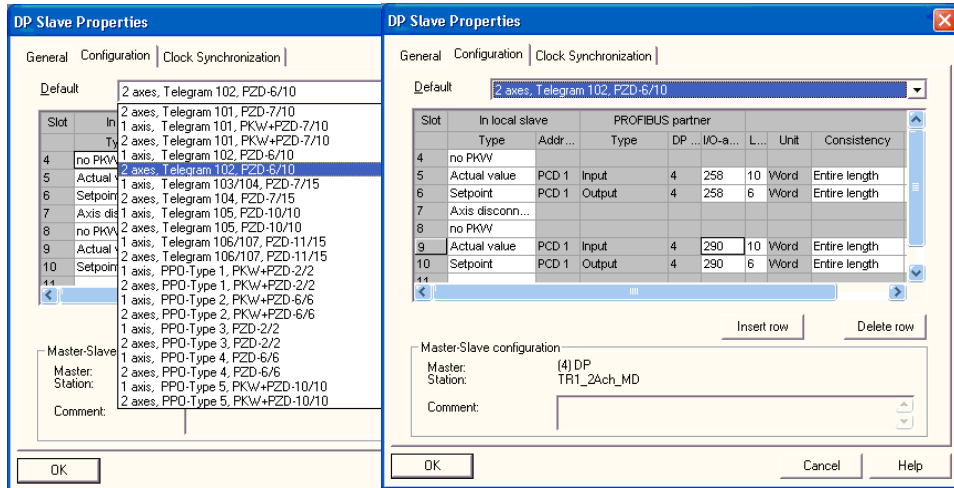


4. Select **PROFIBUS-DP > SIMODRIVE > SIMODRIVE 61 1 universal** (6SN1114-0NB01-0AA0 DP slave, Drive ES interface) from the hardware catalog.

Note:
This drive is only available in the hardware catalog after installing DRIVE ES V5.1 SP 2.
5. Parameterize the **PROFIBUS DP interface properties** as described in Point 2.

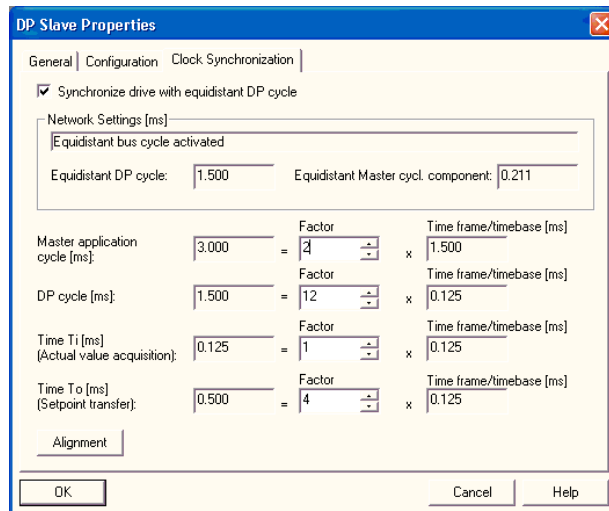
6. Parameterize the DP slave properties. Open the **DP slave properties** dialog and set the following on the **Configuration** tab:

- Default: 2 axes, message frame 102 PZD-6/10
- I/O addresses: 1st axis 258
2nd axis 290
(parameter "Logic addresses (external SDB)" of the FM357-2)



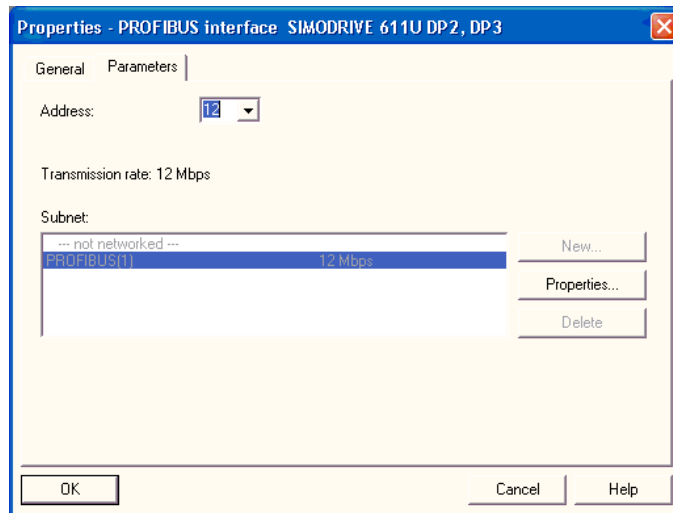
Select the following from the **Cycle synchronization** tab:

- Synchronize drive on equidistant DP cycle: yes
- Master application cycle [ms]: 3,000 (is fixed with FM357-2 and 611-U)

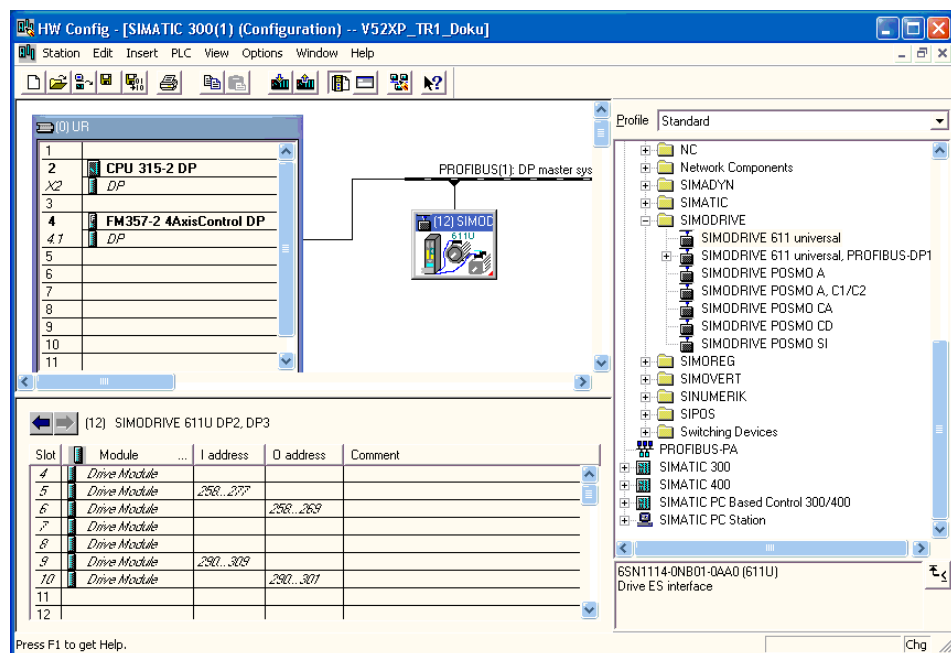


Select the following from the **General** tab:

Address: 12 (must also be parameterized accordingly in the drive)

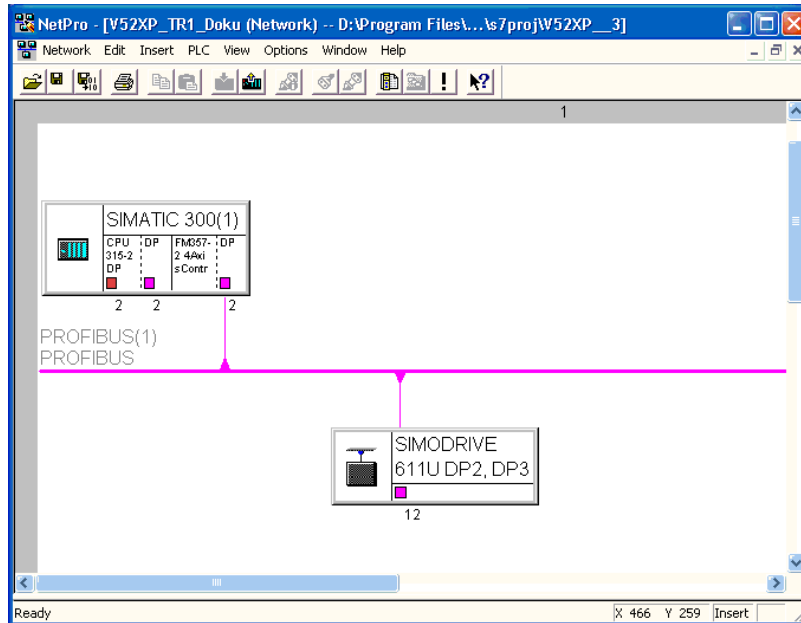


7. Your entire project will now look as follows:



8. Save and compile your project with NetPro.

The screenform below shows your project in the NetPro view (*SIMATIC Manager* > **Tools** > **Network configuration**):



9. Load your project into the FM357-2.

Note:

Answer the interrogation "Do you want to restart the module now?" with **No**.

This function is currently not yet possible using the *SIMATIC Manager*.

Trigger restart either via the parameterization tool of the FM357-2 (as described in Point 10.) or via **STOP > RUN** on the CPU.

10. Set the following parameters in the FM357-2:

	1st axis	2nd axis
Drive:	Profibus drive	Profibus drive
Logical address:	258	290
Frame message type:	611U type 102	611U type 102
Profibus drives:	External SDB	

The screenshot shows the 'Axis Configuration' dialog box with the following settings:

Parameter	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5
Active axis number:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Machine axis name:	X1	Y1	Z1	A1	B1
Channel and special axis names:	X	Y	Z	A	
Geometry axis name:	X	Y	Z		
Axis type:	Linear axis	Linear axis	Linear axis	Linear axis	Linear axis
Drive:	Profibus drive	Profibus drive	Simulation	Simulation	Simulation
Logic address:	258	290	322	354	
Message frame type:	611U type 102	611U type 102	Standard type E	Standard type E	
External master value	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
VDI axis output when simulating	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Controlling	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Global measuring	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DSC (Dynamic Servo Control)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
CPU axes:	Independent CPU axes: 0				
PROFIBUS configuration:	PROFIBUS drives: External SDB				

Buttons: OK, Cancel

5.2.2 Sample project PROFIBUS-DP drive and PG/PC connected to the MPI interface of the CPU

General

Integrate a PG/PC into the PROFIBUS-DP drive sample project.

The drive will be parameterized using the PG/PC connected to the MPI interface of the CPU. This means that the routing functionality of the FM357-2 will be used.

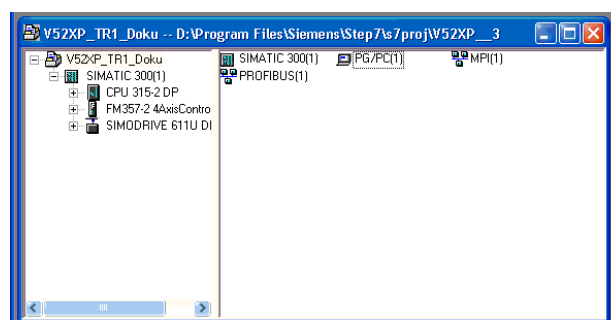
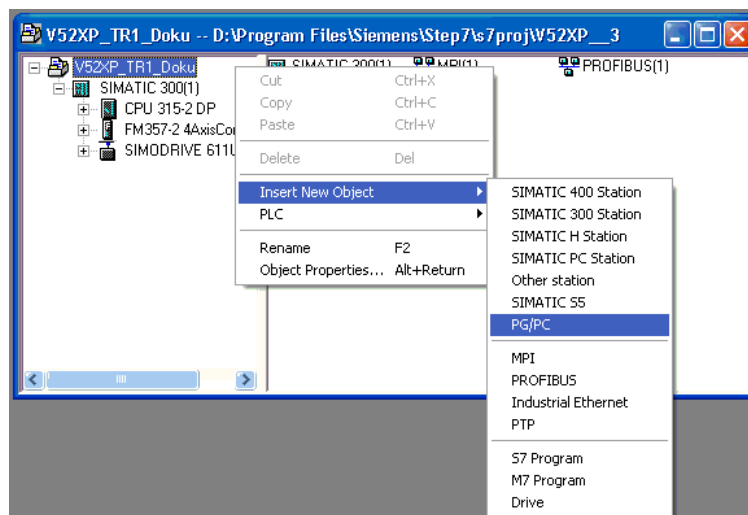
Prerequisites

STEP 7 V5.2 and DRIVE ES V5.1 SP2 is installed.

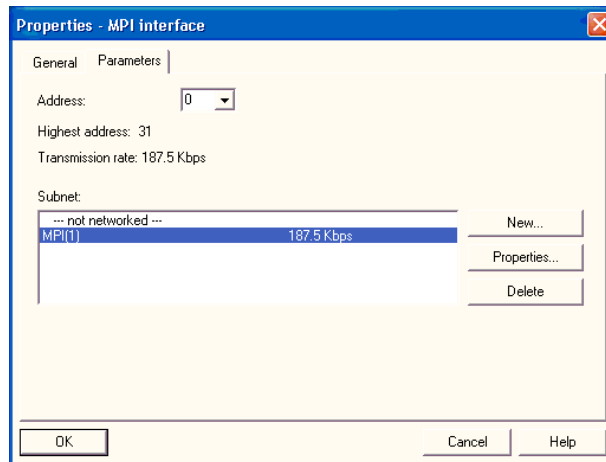
Knowledge of drive parameterization using the SimoComU tool.

Sequence

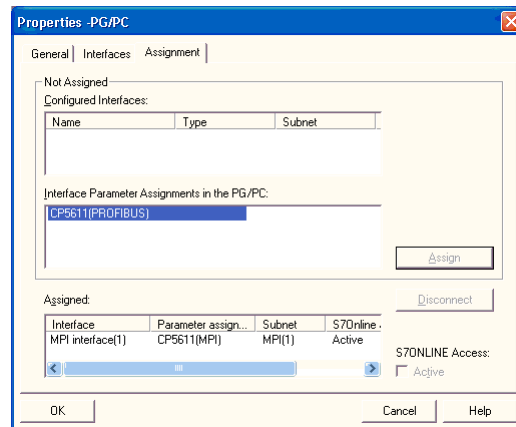
1. Create a project as described in Section 5.2.1.
2. Insert a PG/PC into your project.



3. Parameterize the PG/PC interface.
 Select **Interfaces > New > Type > MPI** from the **Properties – PG/PC** dialog box and open there the **Properties – MPI interface** dialog. Select the following from the **Parameters** tab:
 - Address: 0
 - MPI(1): 187.5 kbit/s
 (the interface on the PG/PC side must be configured accordingly)

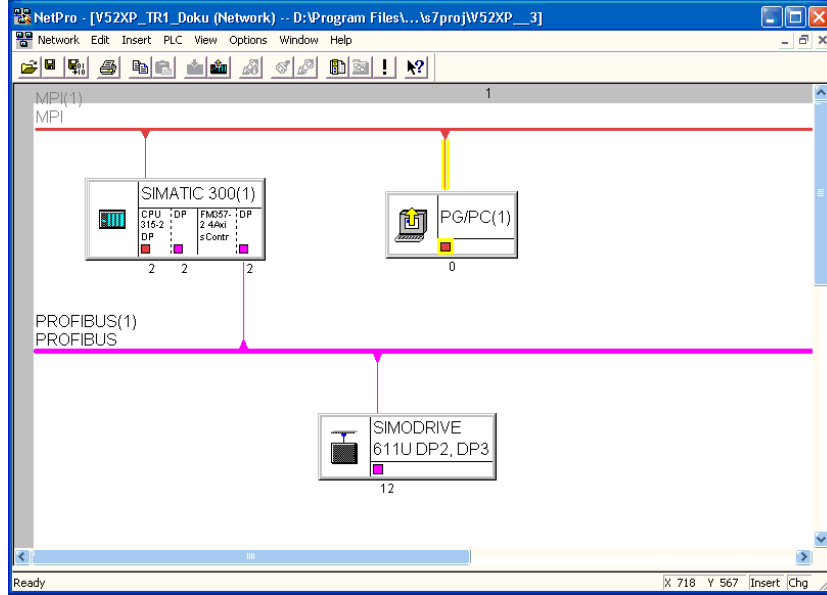


4. Assign the PG/PC to the MPI interface of the CPU.
 In the dialog box **Properties – PG/PC**, select the **Assignment** tab and click on the "Assign" button.



5. Save and compile your project with NetPro.

The screenform below shows your project in the NetPro view (*SIMATIC Manager* > **Tools** > **Network configuration**):



6. Load your project into the FM357-2.

Note:

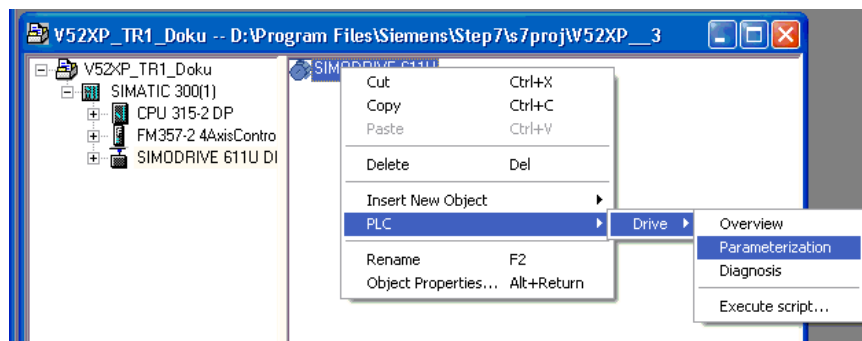
Answer the interrogation "Do you want to restart the module now?" with **No**.

This function is currently not yet possible using the *SIMATIC Manager*.

Trigger restart either via the parameterization tool of the FM357-2 or via **STOP > RUN** on the CPU.

7. Call the drive parameterization tool:

SIMODRIVE 611U > Target system > Drive > Parameterization



The SimoComU tool will be started and connected to the selected drive automatically.

Prerequisite:

The drive is turned on and the DP address is parameterized correctly.

5.2.3 Sample project PROFIBUS-DP drive and PG/PC connected to the DP interface of the CPU

General

The parameterization of the FM357-2 and the drive parameterization is to be performed via the PG/PC connected to the PROFIBUS-DP interface of the CPU.

Note:

The communication from the DP interface to the K bus of the FM357-2 is carried out via routing in the CPU. This requires a routing-capable CPU, such as the CPU 315-2DP (315-2AF03-0AB0 V1). The communication times are correspondingly longer, compared with the MPI interface (direct access to the K bus).

Prerequisites

STEP 7 V5.2 and DRIVE ES V5.1 SP2 is installed.

Knowledge of drive parameterization using the SimoComU tool.

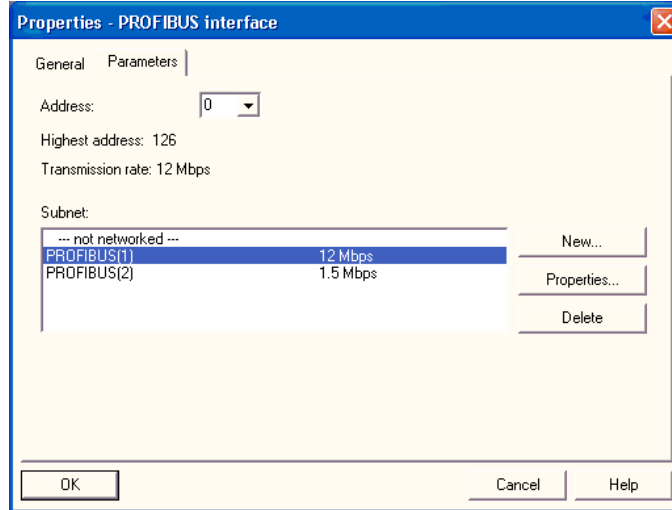
Routing-capable CPU. The PG/PC is connected to the MPI of the CPU and does not exist or is not assigned in the S7 project. (The PROFIBUS-DP interface of the CPU becomes active after loading the project into the CPU and restarting the CPU).

Sequence

1. Create a project as described in Section 5.2.1.
2. Activate the DP interface of the CPU.
In the dialog box **Properties – PROFIBUS DP interface**, use the **New...** button to open the dialog box **Properties – New PROFIBUS subnet**.
Set the following on the **Network settings** tab:
 - Data transfer rate: 1.5 Mbit/s
 - Profile: DP
 - Address: 0
3. Compile and load the project into the CPU and into the FM357-2 **separately**.
Note:
Answer the interrogation "Do you want to restart the module now?" with **No**.
This function is currently not yet possible using the *SIMATIC Manager*.
Trigger restart either via the parameterization tool of the FM357-2 or via **STOP > RUN** on the CPU.
4. Disconnect the PG/PC hardware from the CPU. Reconfigure the PG/PC interface to PROFIBUS DP and connect the PG/PC to the PROFIBUS-DP interface of the CPU.
5. Insert a PG/PC into your project.
6. Assign the PG/PC to the DP interface of the CPU via NetPro.

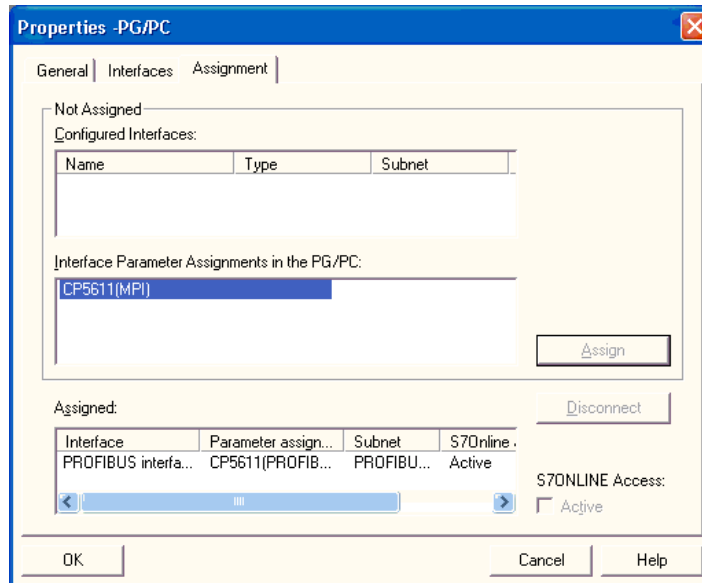
Configure network > Double-click on PG/PC > Interface > New > Profibus type

PROFIBUS interface properties > PROFIBUS(1)



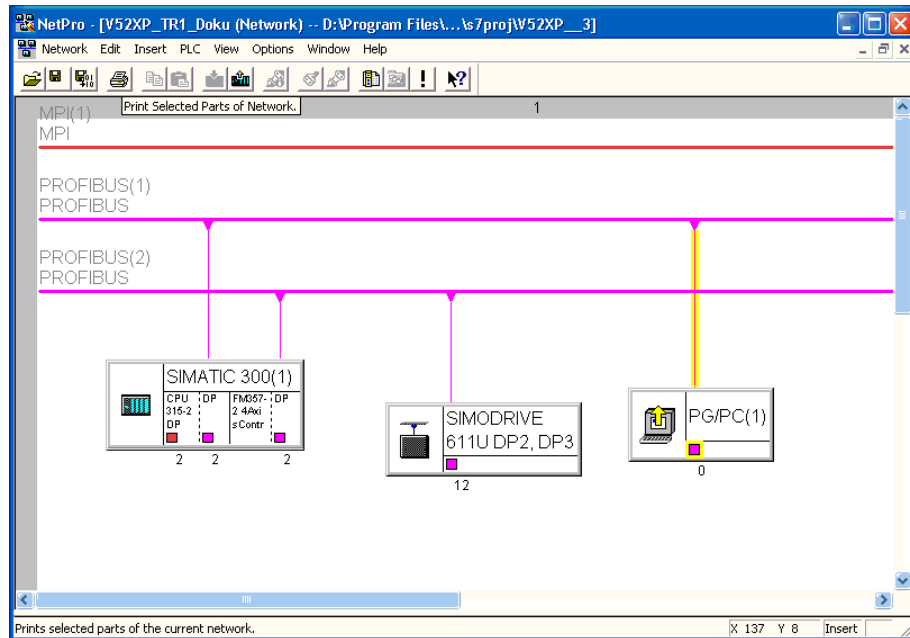
7. Assign the PC/PC to the DP interface.

PG/PC properties > Assignment > Assign > CP561 1 (PROFIBUS)



8. Save and compile your project with NetPro.

The screenform below shows your project in the NetPro view (*SIMATIC Manager* > **Tools** > **Network configuration**):



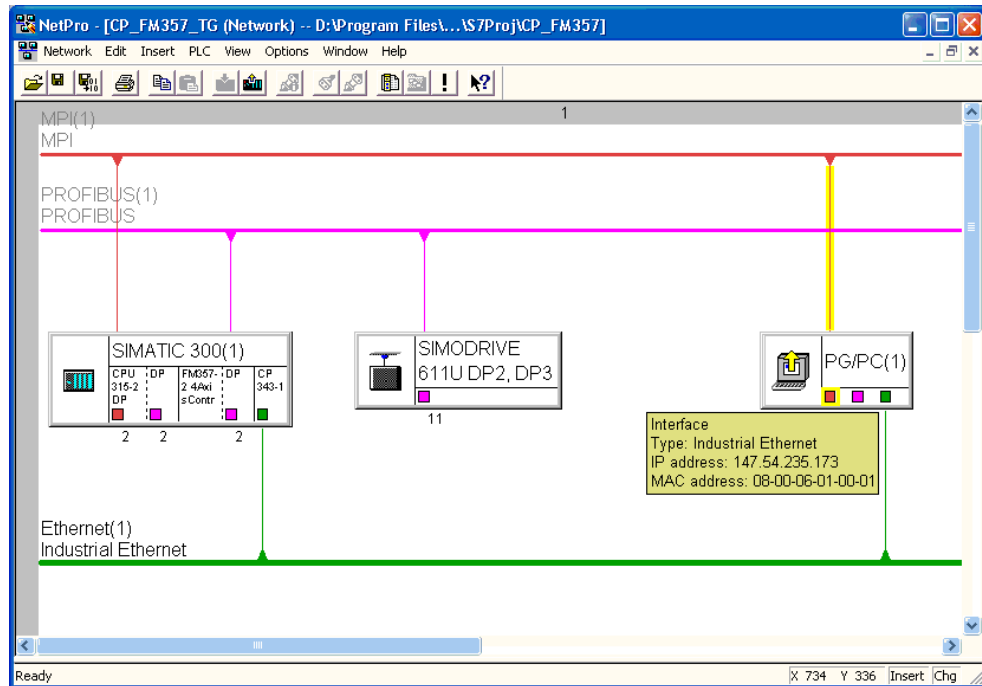
9. Reloading into the modules is not necessary.

Now you can parameterize the FM357-2 and the PROFIBUS-DP drive via the PROFIBUS-DP interface of the CPU.

5.2.4 Sample project "Coupling the PG/PC via Ethernet"

In the sample, the PG/PC is coupled via Ethernet and CP 343-1.

Thus, it is possible to parameterize and start-up the FM 357-2 and the PROFIBUS-DP drive via Ethernet.



5.3 Entry to "Parameterizing the FM 357-2"

Prerequisites

You have installed the software on your PG/PC, as described in Section 5.1 .

You have created a project, as described in Section 5.2.

To parameterize the FM 357-2 online, communication with the CPU is required.

Make sure that the basic program runs in the CPU (see Section 6).

The CPU must be in the RUN condition.

1. Double-click on the module you wish to parameterize.

The **Properties** dialog box appears:

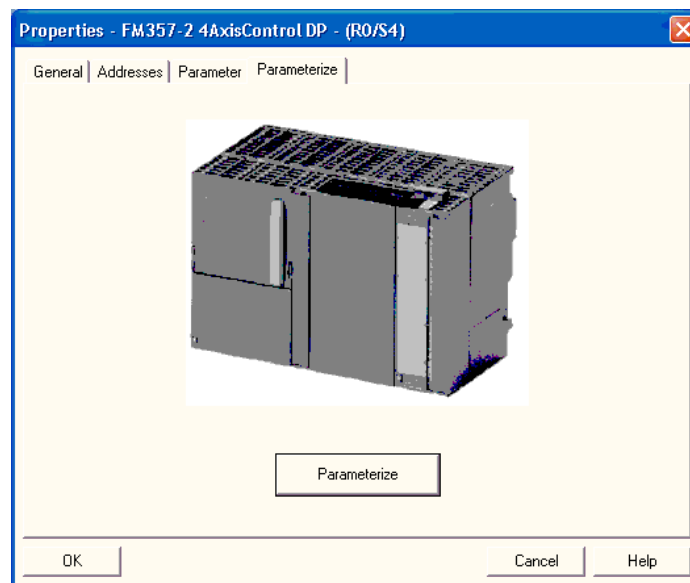


Fig. 5-2 Entry to "Parameterizing the FM 357-2"

2. In the screen above, you can assign the FM 357-2
 - a designation,
 - change the address for the FM,
 - change basic parameters and
 - start the parameterization of the FM357-2 using tabs (General, Addresses and Basic Parameters).

Click on the **Parameterize** button to call the parameterization user interface.

Now you can parameterize your module. The Section 5.5 provides an overview of the data which can be parameterized.

If you have configured your project, you can also call the dialog box **Properties** by selecting the module and using the menu command **Edit > Object properties**.

5.4 Adaptation to firmware

General

You can use the parameterization tool to process offline machine data generated with earlier firmware versions and then load them to FMs which have a different firmware version.

Before you can do this, you must create an image of the standard MDs for each firmware version in a *.BIN file. Only then can the parameterization tool regenerate and process machine data of any firmware version in offline mode.

You are prompted to select the firmware version when you insert a new block.

Update procedure

When the online link to the FM 357-2 is set up, the parameterization tool checks the firmware version of the FM 357-2.

If an unknown (new) version is installed, the description of all machine data can be read out and stored offline in an offline database (*.BIN) file.

A dialog box informs you of the activation of the default values and you can select the following:

- OK - (Load default values, Start firmware alignment)

The parameterization tool must activate the default values (MD) of the firmware as active machine data on the module. During this process, the memory on the module is reorganized and existing NC programs, zero offsets and tool offsets are deleted.

The offline database is created.

- Cancel

If you want to save data (e.g. NC programs, etc.) before the firmware alignment, select **Cancel**.

Machine data cannot be read, edited or saved online.

You can create the offline database from the menu **Options > Firmware Alignment**.

5.5 Parameter data

What can be parameterized?

It is possible to parameterize the following data areas:

- Machine data (parameters)
- R parameters
- Zero offset
- Tool offset values
- GUD-Program (see section 10.22, User variables)
- NC programs

You can edit parameter data either online or offline (on a programming device/PC) and save the data in a project.

Online editing

For online operation, the link between the PG/PC and the S7-300 CPU must already be set up (see Figures 4-1 and 4-3).

You can call up the online project with menu **PLC > Open Online Project**. All the parameter data (data storage areas) are displayed. The blocks to be parameterized (the blocks cannot be used in STEP 7) can be opened by double-clicking.

The following blocks are stored in the working memory of the FM 357-2:

- Machine data
- R parameters
- Zero offset
- Tool offset values
- GUD-Program (editor)

The following blocks are stored in the program memory of the FM 357-2:

- NC programs
 - Main programs
 - Subprograms
 - Special program

Offline editing

To create parameter data without an FM 357-2 on the PG/PC, please proceed as follows:

1. You can create a new offline project with menu **File > New**.

This project shows the data areas to be parameterized but does not yet contain any blocks.

2. Select a data area. Press the right mouse button or **Edit** and select the menu **Insert New Block**. A new block is displayed.
3. You can open the block by double-clicking.

You can save the blocks as a project (*.mcp) on the hard disk of your programming device/PC by selecting **File > Save As...**


Existing projects can be edited by selecting **File > Open**.

You can import files from earlier versions into your open project by selecting **Edit > Import File**.

You can also save a block from a project in a file of an earlier version (e.g. machine data *.pda) by selecting **Edit > Export To File**. You can only do this when the block is open.

Integrated help

The parameterization interface features an integrated help function that will support you in parameterizing the positioning module. To call up the integrated help:

- Select menu command **? > Help Index...** or
- press the **F1** key or
- click on the symbol.  Then click on the element or window about which you need to know more and press the left-hand mouse key.

5.5.1 Machine data (parameters)

General

Machine data are used to adapt the FM 357-2 to the user's application. Parameterization with machine data is essential in order for the functions of the FM 357-2 to be activated.

Parameter definition

There are two ways in which the user can parameterize the FM 357-2:

- Parameterization wizard (normal mode)
- List-based parameterization (expert mode)

By selecting **View > Table View** you can switch between the Parameterization wizard and list-based parameterization.

Information about parameterization in an online project

When you change machine data (wizard or list-based parameterization), most of the data are not activated immediately. One of the following actions is necessary in order to activate the machine data, depending on the type of machine data:

- Send "New config" signal
- Reset
- FM restart

In the parameterization wizard, all modified data are not transferred to the FM until you activate the button **Finish** or the menu item **PLC > Transfer/Activate Data**. The necessary action (e.g. Reset) is selected and executed simultaneously.

In list-based parameterization, all machine data are transferred to the FM when you press the **Enter** key or when the **Cursor** leaves the input field. No action is performed to activate the data. You can see the action required to activate the individual data by selecting and reading the menu **View > Contents of 5th column > Activation**. You can activate the machine data by selecting **PLC > Transfer/Activate Data**.

When you switch from the parameterization wizard to list-based parameterization, a window appears prompting you to select whether to transfer the modified data to the FM and activate the data. It is **not** possible to transfer and activate the data modified in the machine data wizard afterwards.

All machine data are also activated by an FM restart. You can initiate an FM restart by selecting **PLC > Restart > Normal Startup**.

Parameterization wizard

You can call up the parameterization wizard as follows:

1. Open an online or offline project (see Section 5.5).
2. Select the “machine data” area in the project window.
3. Double-click the block to open the machine data wizard.

The Parameterization Wizard contains the main parameters that you will need for initial start-up. With the support of the Wizard, the user is requested to enter data in interactive mode. Parameterization using the Wizard represents the basic parameterization tool and, as such, is opened first (the user is working in normal mode). Expert mode provides an alternative method of parameterization using lists.

The following screenshot shows a parameterization dialog in the Parameterization Wizard.

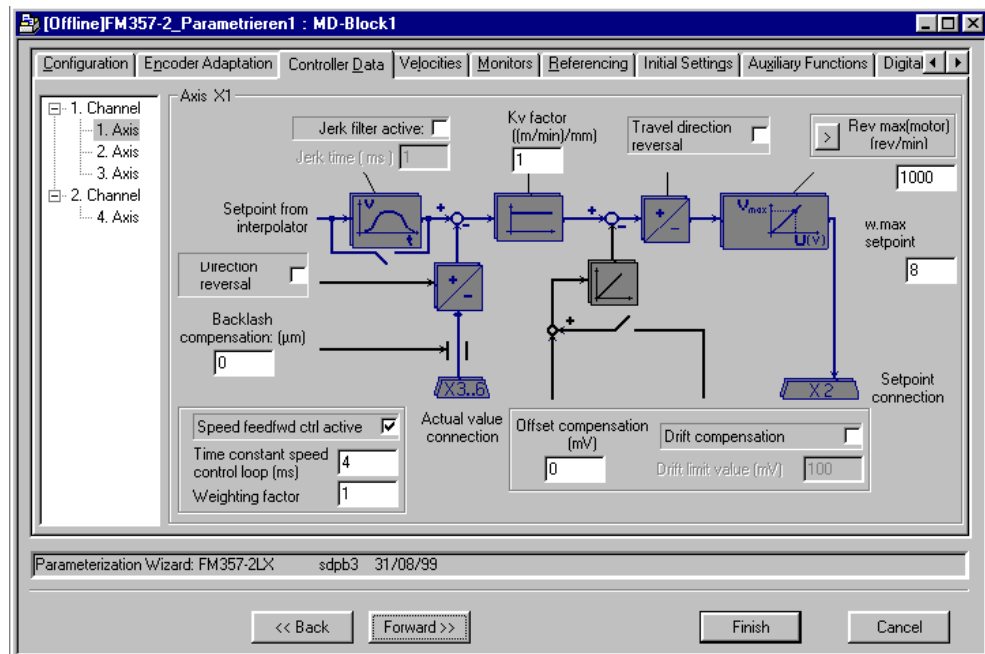


Fig. 5-3 Machine data, e.g. controller data

Using the “User display” index card, you can transfer machine data from list-based parameterization to the Parameterization Wizard.

Machine data list

The following table describes the machine data (parameters) which you can enter in the parameterization tool (Parameterization Wizard).

Table 5-2 Machine data (parameters)

Parameters	Default values	Value range/meaning	Unit	Sec.
Configuration				
Internal unit system	metric	Metric = 10^{-3} Inches = 10^{-4}	[mm] [inches]	9.1
Override coding	Gray	Gray (default value) binary	–	9.1
Max. cycle time of user program	40	10 to 200	[ms]	9.1.1
Servo cycle	3	3 (permanent with PROFIBUS-DP drive) 0,5 to 32	[ms]	9.1.1
Ratio between servo cycle and IPO cycle	3	1 to 100	–	9.1.1
Override coding	Gray	Gray (default setting) Binary	–	9.1
Memory configuration (R parameters, FIFO range, protection zones, curve tables)				
Number of R parameters	100	0 to 10 000	–	10.19
Number of FIFO ranges	0	0...10	–	10.21
Start of FIFO range	0	0...max. value ¹⁾ 1) dependent on "Number of R parameters" and "Number of FIFO elements"	–	10.21
Number of FIFO elements	0	0...max. value ¹⁾ 1) dependent on "Number of R parameters" and "Start of FIFO range"	–	10.21
FIFO range sum generation	no	no yes	–	10.21
Number of protection zones	0	0: No protection zones can be defined 1, 2, 3, 4: This number of protection zones can be defined.	–	9.15
Number of curve tables	0	0 to 100	–	9.16.3
Number of curve segments	0	0 to 1 800	–	9.16.3
Number of curve table polynomials	0	0 to 3 600	–	9.16.3
Number of GUD variables globally FMs	10	0 to ¹⁾ 1) dependent on the free SRAM available	–	10.22
Number of GUD variables globally channels	40	0 to ¹⁾ 1) dependent on the free SRAM available	–	10.22

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Reserved memory area (SRAM) for GUD values	12 kB	0 to ¹⁾ 1) dependent on the free SRAM available	–	10.22
Channel and axis configuration				
Active axes	1	1...5 (5th axis only in simulation)	–	9.1.2
Axis name	X1, Y1, Z1, A1, B1 X, Y, Z A, B X, Y, Z A, B B1, B	Machine axis Channel axis Geometry axis Special axis 5th axis only possible in simulation Note The following designations may not be used: <ul style="list-style-type: none"> • D, E, F, G, H, I, J, K, L, M, N, P, R, S, T (max. 8 characters) • Instructions which are used for programming purposes 	–	9.1.2
Axis type	Linear axis	Linear axis = (10 ⁻³ mm or 10 ⁻⁴ inches) Rotary axis = (10 ⁻³ degrees) Modulo rotary axis = (10 ⁻³ degrees)	–	9.1.2
Drive	Simulation	Simulation Servo drive Stepper motor (SM) without encoder Stepper motor (SM) with encoder PROFIBUS-DP: SIMODRIVE 611-U PROFIBUS-DP: MASTERDRIVES PROFIBUS-DP: External SDB	–	9.1.2
Axis module	none	no 1st single-axis module 2nd single-axis module 3rd single-axis module 4th single-axis module 1st two-axis module 2nd two-axis module	–	9.1.2
Logic address (external SDB)	258 290 322 354	258 290 322 354	–	9.1.2
Message frame type (external SDB)	Type 2	Standard types 1 to 5 611U types 101 to 105	–	9.1.2
External master value	none	no yes	–	9.1 9.16.3

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
VDI output (with simulation)	no	no yes	–	9.1
Controlling	no	no yes	–	9.20
Global measuring	no	no yes	–	9.17
DSC (Dynamic Servo Control)	no	no yes	–	9.4
Independent CPU axes	0	0 to 4	–	6.5.1
Active channels	1	1 to 4	–	9.1.3
Channel assignment	1	1 to 4	–	9.1.3
Geometry axes	1. X, 2. Y, 3. Z	1. X, 2. Y, 3. Z	–	9.1.3
Encoder matching				
Encoder version	Rotary	Linear: Linear scale Rotary: Rotary encoder	–	9.2
Encoder mounting	Motor	Motor: Indirect path encoding Machine: Direct path encoding	–	9.2
Encoder type	Incremental	Incremental: Incremental encoder Absolute: Absolute encoder (SSI)	–	9.2
Distance per spindle revolution	10	0,001 to 100 000	[mm/rev]	9.2
Load gearbox (LG)	1/1	No. of motor revolutions $\frac{1 \text{ to } 10\,000}{1 \text{ to } 10\,000}$ No. of spindle revolutions $\frac{1 \text{ to } 10\,000}{1 \text{ to } 10\,000}$	–	9.2
Resolver gearbox	1/1	No. of motor revolutions $\frac{1 \text{ to } 10\,000}{1 \text{ to } 10\,000}$ No. of encoder revolutions $\frac{1 \text{ to } 10\,000}{1 \text{ to } 10\,000}$	–	9.2
2nd encoder (only relevant for PROFIBUS drives)	No	No Yes	–	9.2
Increments per encoder revolution	2048	2 to 16 384	–	9.2.1
Indexing period	0.01	0.001 to 100	[mm]	9.2
Baud rate	250	250 kHz 400 kHz 500 kHz 1 MHz	[kHz] [MHz]	9.2.2
Coding	x	Output code of encoder: Gray code Binary code	–	9.2.2
Measuring	x	Not provided Provided	–	9.2.2

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Sensing probe connection	Input 4	Input 4 Input 5	–	9.2.2
Traversing range extension	On	ON OFF	–	9.2.2
Encoder revolutions in the absolute range	4,096	4,096 2 ⁰ to 2 ¹³	–	9.2.2
Message frame length	x	25-bit multi-turn 13-bit single-turn 21-bit multi-turn	–	9.2.2
Steps per encoder revolution	–	8,192 only with 25-bit multi-turn and 13-bit single-turn 4,096 2,048 ... 2 ¹	–	9.2.2
Steps per motor revolution	1 000	2 to 1 000 000	–	9.2.3
Controller data				
Jerk filter active	No	No Jerk filter not active Yes Jerk filter active	–	9.3
Jerk time	1	0 to 100	[ms]	9.3
Direction reversal/ actual value	No	No No reversal Yes Reversal	–	9.3
Backlash compensation	0	–10 000 to +10 000 Positive value: on positive backlash Negative value: on negative backlash	[mm], [10 ^{–3} deg]	9.3
Position loop gain (K _v factor)	1	0,1 to 100	[(10 ³ mm/ min)/mm], [(10 ³ deg/ min)/deg]	9.3
Traversing direction reversal	No	No No reversal Yes Reversal	–	9.3
Max. motor speed U _{max} [Motor] (servo drive and stepper drive)	1 000	1 to 999 999	[rev/min]	9.3
Maximum velocity V _{max} [axis] (servo driver and stepper drive)	10 000	1 to 999 999	[mm/min], [rev/min]	9.3
Voltage setpoint max (servo drive)	8	0,1 to 10	[V]	9.3
Offset compensation	0	–2 000 to +2 000	[mV]	9.3
Drift compensation	No	No Drift compensation off Yes Drift compensation on	–	9.3
Drift limit value	100	0 to 500	[mV]	9.3

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Speed feedforward control active	Yes	No Yes	–	9.3
Time constant for speed control loop	0.5	0 to 10	[ms]	9.3
Weighting factor	1	0 to 10	–	9.3
Transfer dead time	0	–0.02 to 0.02	[s]	9.4
Velocities				
Positioning velocity	10 000	0 to 999 999	[mm/min], [rev/min]	9.5
Axis velocity	2 000	0 to 999 999	[mm/min], [rev/min]	9.5
Rapid traverse override	10 000	0 to 999 999	[mm/min], [rev/min]	9.5
Acceleration pattern Initial setting	Brisk accel- eration	Brisk acceleration Soft acceleration Drive acceleration Ö Brisk acceleration	–	9.5
Acceleration	1	0 to 10 000	[m/s ²], [rev/s ²]	9.5
Jerk	1 000	0 to 100 000	[m/s ³], [rev./s ³]	9.5
Creep velocity	10 000	0 to 999 999	[mm/min], [rev/min]	9.5
Creep acceleration	1	0 to 10 000	[m/s ²], [rev/s ²]	9.5
Path acceleration	10	0 to 1 000	[m/s ²]	9.5
Path jerk	100	0 to 100 000	[m/s ³]	9.5
Braking time EMER- GENCY STOP	0,05	0.02 to 1 000	[s]	9.19
Cutout delay controller enable EMERGENCY STOP	0,1	0.02 to 1 000	[s]	9.19
Monitoring				
Monitoring time (mov- ing into position)	1	0 to 100	[s]	9.6.1
Target range coarse	0,04	0 to 1 000	[mm], [deg]	9.6.1
Target range fine	0,01	0 to 1 000	[mm], [deg]	9.6.1
Following error monitoring system (movement of axis)	1	0 to 1 000	[mm], [deg]	9.6.1
Zero speed control delay	0,4	0 to 100	[s]	9.6.1

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Zero speed range	0,2	0 to 1 000	[mm], [deg]	9.6.1
Threshold velocity for stationary axis	5	0 to 10 000	[mm/min], [rev/min]	9.6.1
Clamping tolerance	0,5	0 to 1 000	[mm], [deg]	9.6.1
Set speed	100	0 to 100	[%]	9.6.1
Monitoring time (set speed)	0	0 to 100	[s]	9.6.1
Actual speed	11 500	0 to 9 999 999	[mm/min], [rev/min]	9.6.1
Tolerance Setpoint/actual position	5.0	0 to 100 000 000	[mm], [deg]	9.6.1
Encoder limit frequency	300 000	0 to 1 500 000 0 to $9 * 10^{299}$ (as from software version 5)	[Hz]	9.6.2
Zero marker monitoring	x	Off: HW encoder monitoring on Off: HW encoder monitoring off On: 1 to 99 or 101 to 10 000 Number of detected zero marker errors	–	9.6.2
Number of increments (rotation monitoring)	2 000	10 to 1 000 000	–	9.6.2
Increment tolerance (rotation monitoring)	50	10...number of increments	–	9.6.2
1st software limit switch plus	10^8	–100 000 000 to +100 000 000	[mm], [deg]	9.6.3
1st software limit switch minus	-10^8	–100 000 000 to +100 000 000	[mm], [deg]	9.6.3
2nd software limit switch plus	10^8	–100 000 000 to +100 000 000	[mm], [deg]	9.6.3
2nd software limit switch minus	-10^8	–100 000 000 to +100 000 000	[mm], [deg]	9.6.3
Working area limitation plus	x	Off: Working area limitation is not effective for this axis On: Working area limitation is effective for this axis	–	9.6.3
Working area limitation minus	x	Off: Working area limitation is not effective for this axis On: Working area limitation is effective for this axis	–	9.6.3
Referencing				
Start without reference point approach	No	No Yes	–	9.7
Reference point approach necessary	Yes	Yes No	–	9.7

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Axis with reference point switch	Yes	Yes No	–	9.7.1
Referencing in follow-up mode	no	no yes	–	9.7.1
Reference point approach direction	Plus	Plus Minus	–	9.7.1
Zero marker/BERO	In front of RPS	In front of reference point switch (RPS) Behind/on RPS	–	9.7.1
Reference-point coordinate	0	–100 000 to +100 000	[mm], [deg]	9.7.1
Reference-point shift	–2	–100 000 to +100 000	[mm], [deg]	9.7.1
Maximum distance to RPS	10 000	0 to 100 000	[mm], [deg]	9.7.1
Max. distance to zero marker/BERO	20	0 to 10 000	[mm], [deg]	9.7.1
Distance from reference point switch to zero mark/BERO	0	0...10 000	[mm], [deg]	9.7.1
Referencing speed	5 000	0 to 999 999	[mm/min], [rev/min]	9.7.1
Creep velocity	300	0 to 999 999	[mm/min], [rev/min]	9.7.1
Creep velocity tolerance	10	0 to 100	[%]	9.7.1
Approach velocity	1 000	0 to 999 999	[mm/min], [rev/min]	9.7.1
BERO edge evaluation	1	1-edge evaluation 2-edge evaluation	–	9.7.2
Traversing direction key	Minus	Minus direction Plus direction	–	9.7.3
Encoder alignment status	Not aligned	Not aligned Enabled Aligned	–	9.7.3
Actual value (alignment value)	0	–100 000 to +100 000	[mm], [degrees]	9.7.3
Adjustment offset	0	–100 000 000 to +100 000 000	[mm], [degrees]	9.7.3
Current encoder position	0	–100 000 000 to +100 000 000	–	9.7.3
Initial setting				
Initial setting on program start				
Movement	x	Linear interpolation with rapid traverse G0 Linear interpolation with feed G1	–	10.5.4 10.5.6

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Exact stop Continuous-path mode	x	Exact stop G60 Continuous-path mode G64 Continuous-path mode with programmed rounding clearance G641	–	10.6.1 10.6.2
Settable zero offset	x	Settable zero offset off G500 1. Settable zero offset 2. Settable zero offset 3. Settable zero offset 4. Settable zero offset	–	10.3.1
Working area limitation	x	Working area limitation on WALIMON Working area limitation off WALIMOF	–	10.13
Path acceleration pattern	x	Brisk acceleration BRISK Soft acceleration SOFT Drive acceleration DRIVE	–	10.6.3
Plane selection	x	Plane selection G17 Plane selection G18 Plane selection G19	–	10.2.7
Workpiece dimensioning	x	Input as metric G71 Input as inches G70	–	10.2.6
Dimension type	x	Absolute dimensioning G90 Incremental dimensioning G91	–	10.2.3
Speed feedforward control	x	Feedforward control off FFWOF Feedforward control on FFWON	–	10.37
Tool number	0	0 to 29	–	10.16
Behaviour after program end and Reset				
Active plane remains valid	No	No Yes	–	10.2.7
Active zero offset remains valid	No	No Yes	–	10.3
Active tool length compensation remains valid	No	No Yes	–	10.16
Event-controlled program calls				
Start	no	no yes	–	9.8
End of program	no	no yes	–	9.8
Reset	no	no yes	–	9.8
FM restart	no	no yes	–	9.8

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Auxiliary functions				
Output options for M functions	x	Output before movement Output during movement Output after movement	–	9.9
Output options for H functions	x	Output before movement Output during movement Output after movement	–	9.9
Digital I/Os				
Slots	None	None Slots 1 to 4	–	9.10
Module size	1 byte	1 byte 2 bytes	–	9.10
Byte 1	–	Inputs Outputs	–	9.10
Byte 2	–	Inputs Outputs	–	9.10
Outputs: 9...16 17...24 Inputs: 17...24 25...32	17 to 24	9 to 16 17 to 24 25 to 32 Assignment of bit numbers	–	9.10
Analog inputs/outputs				
Slots	None	None Slots 1...4	–	9.10
Module size	2 inputs	2 inputs 4 inputs 8 inputs 4 outputs 4 inputs and 2 outputs 4 inputs and 4 outputs	–	9.10
Channel	–	1...8	–	9.10
Evaluation	0...10 V	0 to 10 V +/- 10 V	–	9.10
Software cams				
Cam pair axis number	1 0	1 0 not assigned 1 1 (1st cam pair to 1st axis) 2 1 (2nd cam pair to 1st axis) 3 2 (3rd cam pair to 2nd axis) ...	–	9.11.1
Cam position Minus cam	0	–100 000 000 to +100 000 000	[mm], [degrees]	9.11.1
Cam position Plus cam	0	–100 000 000 to +100 000 000	[mm], [degrees]	9.11.1

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Lead time/delay time minus cam	0	-100 to +100	[s]	9.11.1
Lead time/delay time plus cam	0	-100 to +100	[s]	9.11.1
Signal level for minus cam	0 → 1	0 → 1 1 → 0 (inverted)	-	9.11.1
Signal level for plus cam	0 → 1	0 → 1 1 → 0 (inverted)	-	9.11.1
Assignment to digital outputs, minus cam	None	No assignment Digital outputs (on-board) 1...8 Digital outputs 9...24	-	9.11.1
Assignment to digital outputs, plus cam	XOR minus cam	XOR minus cam Digital outputs 9...24	-	9.11.1
Timer-controlled path switching signals	None	No assignment 1st cam pair to 8th cam pair	-	9.11.1
Signal output, prioritized	On	ON OFF	-	9.11.1
Path/time cam	Off	OFF ON	-	9.11.1 9.11.5
Hot-spot measurement	None	No assignment 1st cam pair to 8th cam pair	-	9.11.1 9.11.5
Adjustment signal	Off	Off On	-	9.11.1
Signal inversion Cam area > 180 degrees	On	On Off	-	9.9.3
Motion coupling				
Overlaid motion with synchronized actions				
Calculation of compensation value	Absolute	Absolute Integral	-	9.16.6
Upper limit of compensation value	10 ⁸	0 to 100 000 000	[mm], [degrees]	9.16.6
Velocity of compensation value	10 ³	0...axis velocity	[mm/min], [rev/min]	9.16.6
Master value link				
Type of master value link	Setpoint value	Actual value Setpoint Simulated master value	-	9.16.3
Threshold value for coarse synchronism	1	0 to 10 000	[mm], [degrees]	9.16.3

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Threshold value for fine synchronism	0.5	0 to 10 000	[mm], [degrees]	9.16.3
Parameterize curve tables				
Offset to master axis position	0	-100 000 000 to +100 000 000	[mm], [degrees]	9.16.3
Scaling of master axis position	1	-1 000 000 to +1 000 000	-	9.16.3
Offset to slave axis position	0	-100 000 000 to +100 000 000	[mm], [degrees]	9.16.3
Scaling of slave axis position	1	-1 000 000 to +1 000 000	-	9.16.3
Behaviour after program end and Reset				
Active coupled axis groupings remain valid	No	No Yes	-	9.16.1
Master value link remains active	No	No Yes	-	9.16.3
Tangential control remains active	No	No Yes	-	9.14.4
Gantry grouping				
Master axis	-	Machine axis name of master axis	-	9.16.2
Synchronized axis	-	Machine axis name of synchronized axis	-	9.16.2
Separate gantry grouping	No	No Yes	-	9.16.2
Limit value for warning	0	0 to 100	[mm], [degrees]	9.16.2
Cutout limit	0	0 to 100	[mm], [degrees]	9.16.2
Cutout limit for referencing	0	0 to 100	[mm], [degrees]	9.16.2
Tangential control				
Limit angle Intermediate block	5,0	5,0 0 ... 360	[deg]	9.16.5
Fixed stop				
Permit travel to fixed stop	No	No Yes	-	9.18
Fixed stop detection	Following error	Following error Sensor Following error or sensor	-	9.18
Following error for fixed stop detection	2	0 to 1 000	[mm], [degrees]	9.18
Monitoring window	1	0 to 1 000	[mm], [degrees]	9.18
Clamping torque	5	0 to 100	[%]	9.18

Table 5-2 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	Sec.
Torque limit on approach to fixed stop	5	0 to 100	[%]	9.18
Error messages: <ul style="list-style-type: none"> • Axis has not reached the fixed stop • Approach to fixed stop aborted 	Yes	Yes No	–	9.18

List-based parameterization

The software displays a list of the entire machine data of the FM 357-2.

The list parameterization (expert mode) is based on the following documentation:

Lists *SINUMERIK 840D, 810D, FM-NC*

Order No.: 6FC5 297-4AB70-0BP0

You can call up list-based parameterization as follows:

1. Open an online or offline project (see Section 5.5).
2. Select the “machine data” area in the project window.
3. Double-click the block to open the machine data wizard.
4. By selecting **View > Table View** you can call up list-based parameterization.

Notice

Functions which are contained in the list-based parameterization, but not documented in this Manual, must not be used. No claim can be made to these functions.

If modifications are made via the list-based parameterization system, problems may occur if the Parameterization Wizard is subsequently used to set parameters. You should therefore only use the list parameterization in exceptional circumstances.

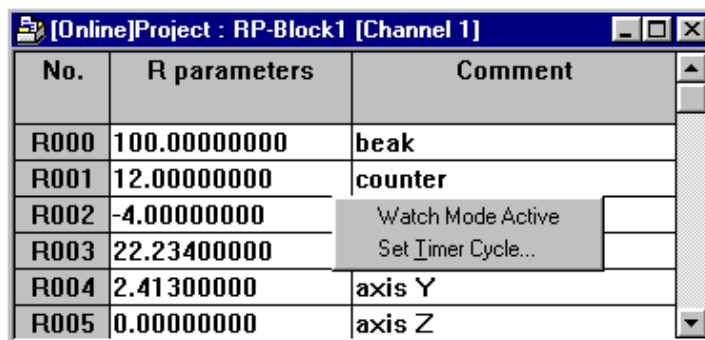
5.5.2 R parameters

You can open the “R parameters” window as follows:

1. Open an online or offline project (see Section 5.5).
2. Select the “R parameters” data area in the project window.
3. Double-click the block to open the window for “R parameters”.

The R parameters for an offline project are stored independent of channels. You can change the names of the inserted blocks. You are not requested to assign the R parameters to a channel until you transfer the parameters to the FM.

R parameters can be updated cyclically in online mode. You can select or deselect this function in menu **View** > **Watch mode active** or in the context menu of the right-hand mouse button.



No.	R parameters	Comment
R000	100.00000000	beak
R001	12.00000000	counter
R002	-4.00000000	
R003	22.23400000	
R004	2.41300000	axis Y
R005	0.00000000	axis Z

Fig. 5-4 Entering values for R parameters

For programming of R parameters, see Section 10.19

5.5.3 Zero offset

You can open the “zero offset” window as follows:

1. Open an online or offline project (see Section 5.5).
2. Select the “zero offset” data area in the project window.
3. Double-click the block to open the window for “zero offsets”.

The zero offsets for an offline project are stored independent of channels. You can change the names of the inserted blocks. You are not requested to assign the zero offsets to a channel until you transfer the parameters to the FM.

For programming of zero offsets, see Section 10.3

5.5.4 Tool offset values

You can open the “tool offset values” window as follows:

1. Open an online or offline project (see Section 5.5).
2. Select the “tool offset values” data area in the project window.
3. Double-click the block to open the window for “tool offset values”.

The tool offsets for an offline project are stored independent of channels. You can change the names of the inserted blocks. You are not requested to assign the zero offsets to a channel until you transfer the tool offset values to the FM.

For programming of tool offset values, see Section 10.16

5.5.5 NC programs

You can open the window for creating NC programs as follows:

1. Open an online or offline project (see Section 5.5).
2. Select the “NC programs” data area in the project window.
3. Double-click the block to open the window for “NC programs”.

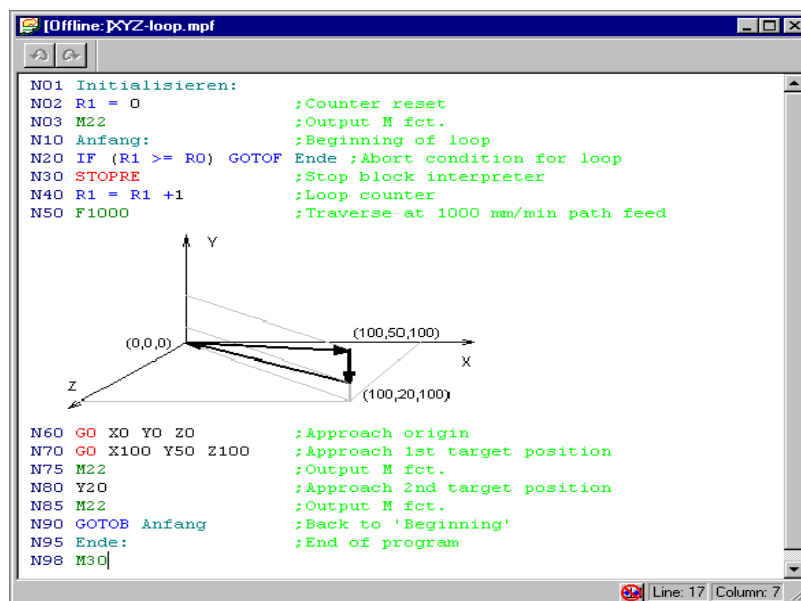


Fig. 5-5 Input of values for NC programs

Graphics integrated in NC programs cannot be stored in online mode.

NC programs are stored on the FM independent of channels and can be started from any channel.

For NC programming, see Chapter 10

For special program, see Chapter 9.8, Event-controlled program calls

5.6 Settings of parameterization interface

You can change the settings for the active window by selecting **Edit > Properties**.

Example

You are in the Startup window and select **Edit > Properties**; you can now change the refresh rate in the dialog box.



Programming of Standard Function Blocks

6

Section overview

Section	Title	Page
6.1	Programming fundamentals	6-3
6.2	Startup with the "Parameterize FM 357-2" tool	6-15
6.3	Description of the standard function blocks	6-16
6.4	Distributed installations	6-47
6.5	User handling, function sequences	6-48
6.6	User data blocks of the interface to the FM 357-2	6-59
6.7	Timing diagrams, communication	6-99
6.8	Application examples	6-100
6.9	Technical specifications	6-110

General

The purpose of this function description of the blocks and the interface is to explain the process of communication between the CPU and FM 357-2 in the SIMATIC S7 programmable controller. The blocks, which you parameterize, and the user data blocks (FMx and AXy = interface to FM 357-2) enable you to develop the user program for your application.

This chapter describes an installation with one FM 357-2. The procedure is similar if several FMs (max. 3) are used.

The FM 357-2 can be configured for single-channel (default) or multi-channel functionality.

Example of a multi-channel application:

Channel assignment	Axis assignment	Blocks	Comments
Channel 1	Axis 1, 2, 3	Blocks are channel-independent, i.e. multi-channel assignments are detected automatically.	e.g. axes 1, 2, 3 are operated in "Automatic" mode with program xxx
Channel 2	Axis 4	FB 4 (channel identification as input parameter)	e.g. axis 4 is operated in "Jog" mode with the "Direction minus or plus" signals (user DB "AXy", DBX304.6/7) or in "Automatic" mode with program yyy.

Notice

In the signal address identifier, "n" stands for the channel offset and "m" for the axis offset (see Section 6.6).

e.g. "Jog" control signal user DB "FMx", DBX100.2+n
 Control signal "Direction plus" user DB "AXy", DBX4.7+m

Preconditions

The following conditions must be met in order to create your user program if you want to control the FM 357-2:

- You have installed the software on your PG/PC, as described in Section 5.1.
 The block library with its basic functions is stored as standard under directory **[STEP7 directory]\S7LIBS\FM357_2L**.
- The link from the programming device/PC to the S7 CPU must be established (see Figures 4-1 to 4-3).
- You have already set up your project for the SIMATIC S7 (see "FM 357-2, first steps").

6.1 Programming fundamentals

Overview

In this section, you can find information about:

- Interface, user data blocks (user DBs), Section 6.1.2, page 6-4
- Standard function blocks, overview, Section 6.1.3, page 6-5
- CPU / FM 357-2 communication, Section 6.1.4, page 6-7
- Information about symbolic programming, Section 6.1.5, page 6-8
- Procedure for writing a user program, Section 6.1.7, page 6-14

6.1.1 FM357_2L library

The supplied FM357_2L library contains several S7 programs (directories):

- **"BF_V05xy_centralized"**: All basic function blocks for CPU modules without PROFIBUS connection or for central use of the FM 357-2.
- **"BF_V05xy_distributed"**: All basic function modules for CPU modules with PROFIBUS connection or for both centralized and distributed applications of the FM357-2.
- **"HPU"**: Blocks which are supplied in addition to the basic functions and are required for operation with the HPU
- **"HT6"**: Blocks which are supplied in addition to the basic functions and are required for operation with the HT6.

Notice

The **application example** "zEn16_01_FM357-2_BF_EX" contains the basic function blocks from the directory "BF_V05xy_distributed".

Please observe the notes provided in the following!

Important notes with regard to the standard function blocks required for the centralized or distributed applications of the FM 357-2

There are CPU modules which are only intended for the centralized applications, and there are CPU modules which can be used both in centralized and distributed applications. According to your particular application, you must load one of the directories listed below into your CPU. The two directories differ by the structure of the FC 22 and the additional block OB 86.

- **Directory "BF_V05xy_centralized" (centralized):**

If you are using a CPU without PROFIBUS connection (e.g. CPU 315-1), it is imperative to copy the basic function blocks from this directory into your project. You can also use these blocks if you are using a CPU with PROFIBUS connection and intend to control the FM 357-2 in centralized operation.

CAUTION: If you load these blocks into a CPU and you wish to control an FM 357-2 in distributed operation, error "0104" is entered in the user data block (user DB "FMx", DBW4), and the start is aborted.
(Remedy: Load blocks from the directory "BF_V05xy_distributed").

- **Directory "BF_V05xy_distributed" (distributed):**

If you are using a CPU with PROFIBUS connection (e.g. CPU 315-2 DP or also a CPU 414-2 DP), you must use these blocks in order to control an FM 357-2 in distributed operation.

You can also use these blocks if you intended to use a CPU with PROFIBUS connection and you wish to control the FM357-2 in distributed operation.

CAUTION: If you load these blocks into a CPU without PROFIBUS connection, STEP 7 will report an error during the loading process, and the block FC 22 will not be loaded into the CPU.
(Remedy: Load blocks from the directory "BF_V05xy_centralized").

6.1.2 Interface, user data blocks (user DB)

The user data blocks (interface) are generated and set up automatically by FC 1 offline or during power-up.

The user can access the signals/data of the interface using absolute or symbolic addresses (integration of UDTs of user DBs "FMx and AXy").

The input parameter of the standard function blocks "FMDB_NO" is used to assign the interface to the FM 357-2. The module address is included in the user DB "FMx" (entered by FC 1).

Creation of the user data blocks by FC 1

The following blocks are created:

- User DB for single and multi-channel FM "FMx" (DB no. "x" acc. to input parameter of FC 1).
- One user DB for all axes "AXy" (DB no. "y" = x + 1).

Two consecutive DB numbers are always required for the interface.

Creation of the user data blocks (offline)

It is only possible to read out user DB “FMx” or “AXy” complete with the symbol structure if you generate the user DBs offline in your project and copy these blocks into your CPU. In this case, there is no automatic generation of the user data blocks (interface) by FC 1.

Two consecutive DB numbers are always required for the interface. Neither of these two DB numbers may be greater than or equal to 1000.

Please proceed as follows:

1. Open your project and select **SIMATIC xxx > CPUxxx > S7 Program > Blocks**.
2. Select menu command **Insert > S7 Block > Data Block** to create the data block in STEP 7 (e.g. DB 21 or DB 22).
3. Double-click the new data block to start the LAD/STL/FBD editor.
4. In “New Data Block” dialog box, select “Data block with assigned user-specific data type”.
5. UDT1 and UDT2 are offered.
 UDT1 contains the structure of user DB “FMx”; UDT2 contains the structure of user DB “AXy”.
6. Select UDT1 (e.g. for DB 21) or UDT2 (e.g. for DB 22) and confirm with **OK**.
7. You have now created the user data blocks.
8. Save the user DB with **File > Save**.
9. Close the editor.

6.1.3 Standard function blocks, overview

The following table shows an overview of the FB/FCs, DBs, OBs and CPU timers needed for communication and control of the FM 357-2.

Table 6-1 Standard function blocks for the FM 357-2 (overview)

Block no.	Block name	Meaning/function	Meaning/function
FC 1 Page 6-17	RUN_UP	Call in OB 100 and OB 86, startup / initialization	required for application, no. can be changed ¹⁾
FC 5 Page 6-19	BF_DIAG	Call in OB 82, FM restart, serious internal FM error	required for application, no. can be changed ¹⁾
FC 22 Page 6-22	BFCT	Call in OB 1, cyclic call (synchronization with FM 357-2) Basic functions and operating modes, interface handling, write requests	required for application, no. can be changed ¹⁾

- 1) - **Block no. is default setting**, it is possible to change the block no. in the SIMATIC Manager
 - It is only necessary to change the symbol table entries if symbolic programming is used

Table 6-1 Standard function blocks for the FM 357-2 (overview), continued

Block no.	Block name	Meaning/function	Meaning/function
FB 2 Page 6-24	FM_GET	Call in OB 1, read FM variable according to FM-VAR list (generated with NC-VAR Selector) A user DB is required as an instance DB or a multi-instance DB (for storing the parameters/values of the variables).	Only to be used if "Read variable" function is needed for the application; no. can be changed ¹⁾
FB 3 Page 6-32	FM_PUT	Call in OB 1, write FM variable according to FM-VAR list. A user DB is required as an instance DB or a multi-instance DB (for storing the parameters/values of the variables).	Only to be used if "Write variable" function is needed for the application; no. can be changed ¹⁾
FB 4 Page 6-38	FM_PI	Call in OB 1, "select program", "acknowledge error", "activate MD"; (services stored in DB 15). A user DB is required as an instance DB or a multi-instance DB (for storing the parameters/program name/path).	Only to be used if the function is needed for the application; no. can be changed ¹⁾
DB 15	PI_DI	Internal block, is supplied	Required for operation of FB 6, no. cannot be changed
DB 16	BF_DB16	Internal block, is supplied	Required for application, no. cannot be changed
DB 121 Page 6-45	VAR_ADDR	DB with selected variables, is supplied (generated with NC-VAR Selector)	Only if use of FB 2, FB 3 necessary, no. can be changed ¹⁾
FB 6	BF_FB6	Internal block, is supplied	Only if use of FB 2, FB 3, FB 4 necessary, no. cannot be changed
FC 23	BF_FC23	Internal block, is supplied	Required for application, no. cannot be changed
OB 1	–	Cyclic level	Required for application
OB 82	–	Diagnostic alarm level	Required for application
OB 100	–	Startup level	Required for application
OB 86	–	Subrack failure	for distributed installations
OB 122	–	I/O access error	for distributed installations
Timer 0	–	Assigned internally, cyclic sign-of-life/interface watchdog of FM	–
Timer 1	–	Assigned internally, power-up time-out, internal = 3 min	–

- 1) - **Block no. is default setting**, it is possible to change the block no. in the SIMATIC Manager
 - It is only necessary to change the symbol table entries if symbolic programming is used

Notice

The description below uses the absolute (default) block name.

6.1.4 CPU/FM 357-2 communication

The following diagram shows you how the FM 357-2 and the standard function blocks (FBs, FCs) communicate via the interface.

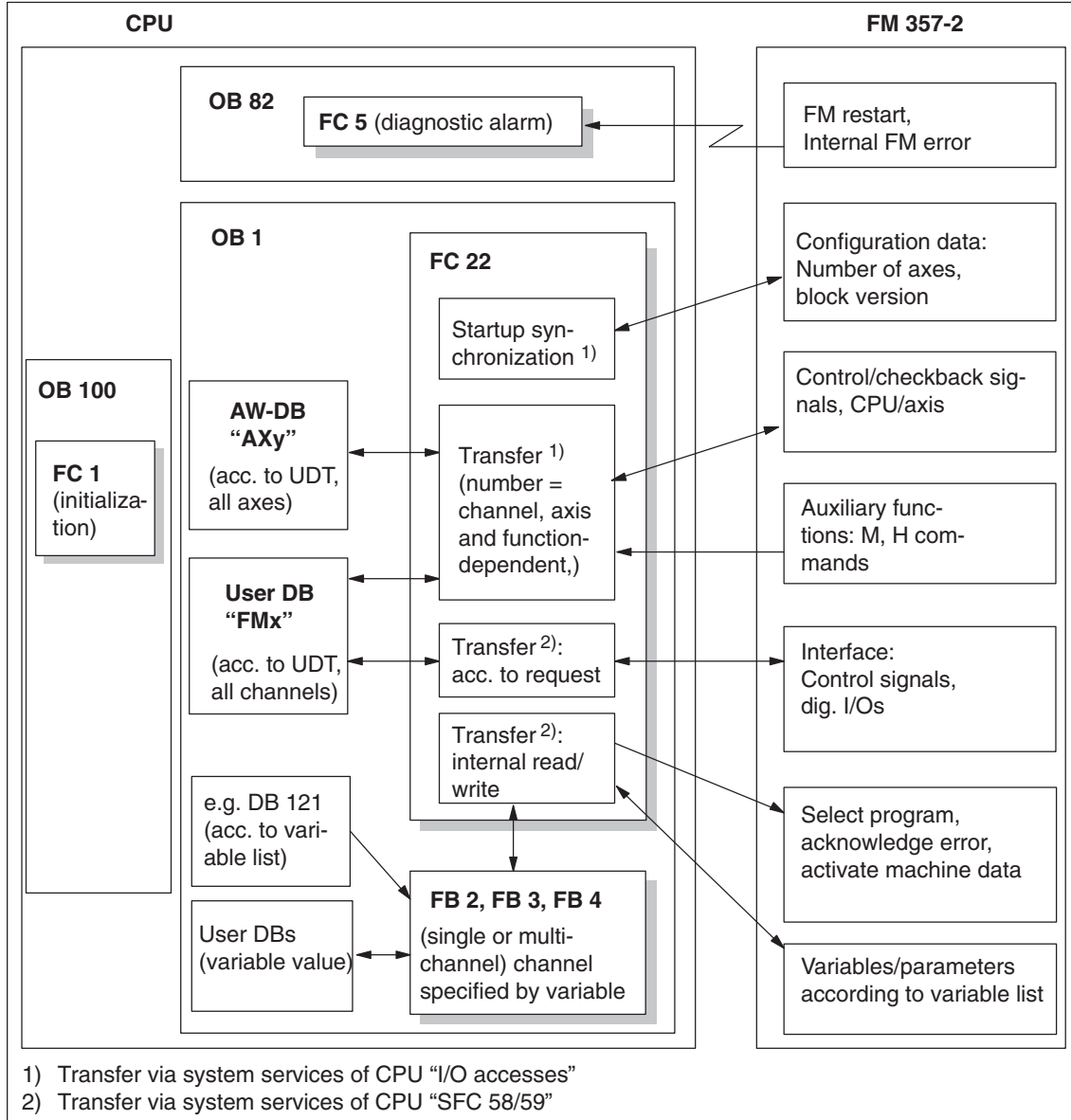


Fig. 6-1 CPU/FM 357-2 communication

6.1.5 Information about symbolic programming

The blocks are entered in the symbol table, by default, with the name of the symbol name, address and data type. (the symbol table is supplied in the project and in the library). If you change the block numbers in the SIMATIC Manager, you must also change the numbers in the symbol table. For example, you can change the numbers of FC 1, 5, 22 and FB 2, 3, 4 (see Table 6-1). The unique assignment of the blocks is established via the symbol table.

Before you can write and compile your user program, you must enter the blocks (user DBs, FCs, FBs) used in your configuration in the symbol table. The symbolic structure of the interface is stored in the two UDT blocks supplied. The symbolic reference is established via your STEP 7 project, the symbol table and the UDT blocks.

Example: extract from the symbol table

You can call up the symbol table as follows:

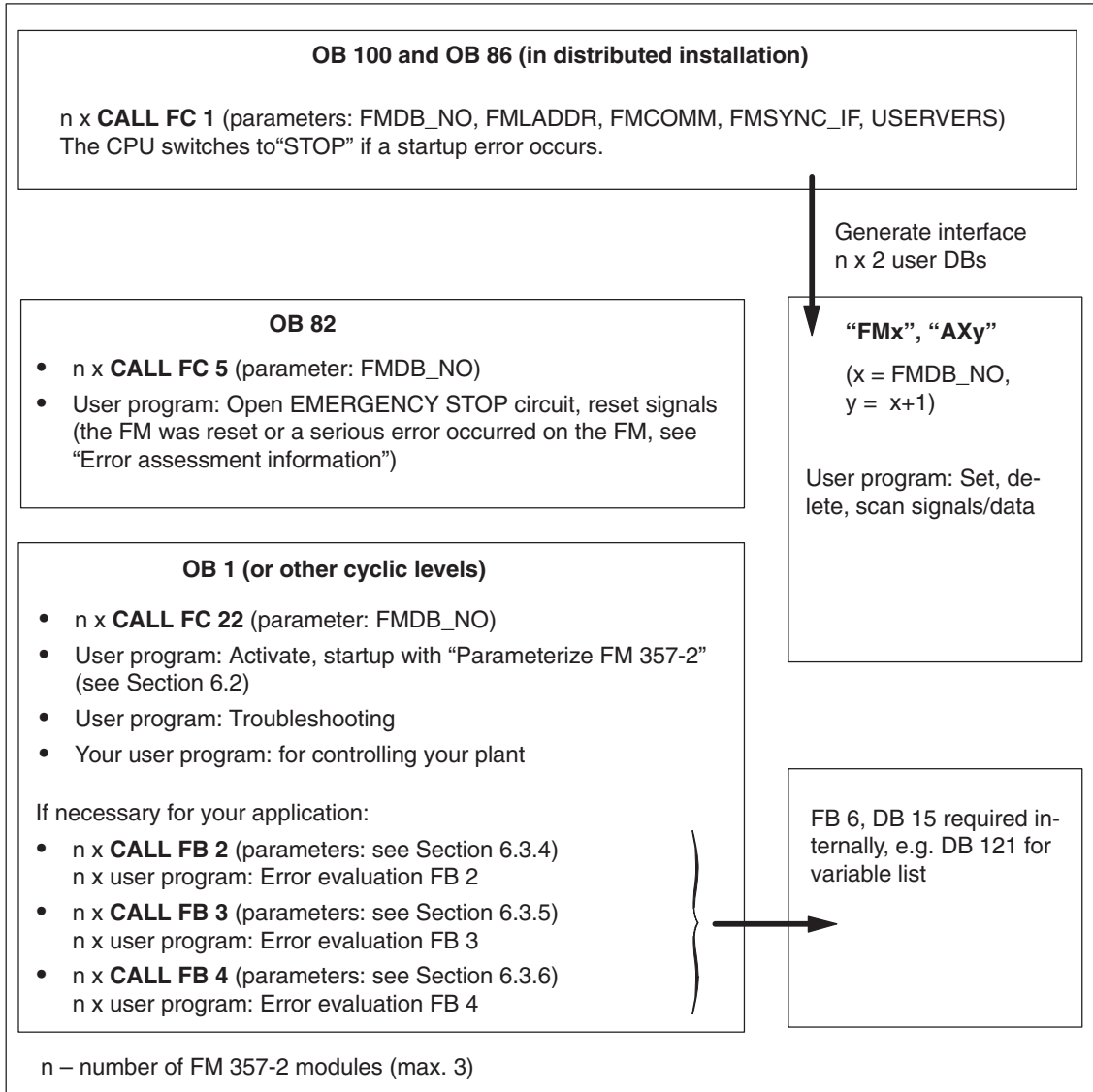
1. Open your project and select **SIMATIC xxx > CPUxxx > S7 Program**.
2. Open **Symbols**.

The following example shows you an extract from the symbol table.

Symbol	Address	Data type	Comments
FM 1	DB 30	UDT1	User DB "FMx" for the FM 357-2
AX1	DB 31	UDT2	User DB "AXy" for the FM 357-2
VAR_ADDR	DB 121	DB 121	Extract from variable list
BFCT	FC 22	FC 22	Basic functions and operating modes

6.1.6 Structure of a user program

The Figure below shows you an overview of the structure of the user program.



FC 1, FC 5, FC 22 and the user DBs "FMx" and "AXy" (generated during power-up or created offline and transferred to the CPU) and the internal blocks FC 23 and DB 16 are required.

User DB "FMx" has a multi-channel structure.

FB 2, 3 and 4 should be included on a function-related basis. This requires user DBs as instance or multi-instance blocks, the supplied internal blocks FB 6, DB 15 and the DB for the variable list (e.g. DB 121, see Section 6.3.7).

CPU ↔ FM 357-2 monitoring

Cyclic bidirectional “sign-of-life monitoring” and “transfer monitoring” (data exchange between CPU and FM) is performed between the FM 357-2 and the CPU (via FC 22).

Monitoring of the CPU by the FM 357-2

Cyclical sign-of-life: default = 100 ms, i.e. FC 22 must be called at least once on the CPU within the time specified above.

Note

If the cycle time of the CPU is greater than 100 ms, the MD 10100 (maximum PLC cycle time) must be altered via the list-based parameterization. You get into the list-based parameterization in the machine data wizard via the menu **View > Table View** (see also section 5.5.1).

Monitoring of the FM 357-2 with FC 22

1. Cyclical sign-of-life: default = 40 ms (9 ms with IPO cycle), via Timer 0

After each call of the FC 22 (start timer 0), the FM must respond within the timer value T0 with a “sign of life”.

The timer value is calculated from four times (default) the set IPO cycle, rounded off to full ms time base (minimum settable time base). The maximum value of the timer is 90 ms.

Caution: Please note the maximum runtime of other processing levels (OBs) in the CPU which can interrupt FC 22 after the timer has been started. It must be less than the value set for Timer 0.

When changing the CPU cycle monitoring time (CPU properties in HW Config), you must ensure that it does not fall below this timer value.

2. Transfer monitoring: Counter default = 500

After the transfer of signals/data to the FM, the FM must send a response before the counter expires.

Information on signal processing:

- The processing cycle of the FM 357-2 (IPO clock cycle) and the application cycle (OB 1) run asynchronously. The processing time of the signals can be 1 to < 2 x IPO cycle, depending on the time of signal transfer to the FM.
- An internal routine ensures that the application cycle (OB 1) is not less than the IPO cycle set on the FM 357-2.
- Please read the description of parameter “FMSYNC_IF” of FC 1 (Section 6.3.1)!
- Please read Section 6.7, Timing charts!
- When a function is aborted (e.g. error, reset), the signal to be triggered must be reset. Afterwards, the function can be restarted.

Notice

Please note that auxiliary functions (including M02 and M30) which have been output must be acknowledged with the signal "Acknowledge auxiliary function" (user DB "FMx", DBX109.0+n)n. Auxiliary functions must also be acknowledged on a "Reset" or program abort.

Notice

The standard function blocks of the "FM357_2L" library are to be called directly in the organization blocks provided for them. It is not permissible to call standard function blocks in a program block of the user program.

For example, for distributed use the standard function block "RUN_UP" needs access to the local data of the Start OB (OB100) and the Subrack failure OB (OB86). For this reason the standard function block "RUN_UP" is to be called directly in the organization blocks OB100 and OB86.

Information for axis start-up:

- You have integrated the appropriate signals (see the following information on error evaluation) in the EMERGENCY STOP circuit (user program).
- Please note that the pulse enable (user DB "AXy", DBX13.7+m) and the servo enable (user DB "AXy", DBX2.1+m) must be set.

Information for testing the user program

When testing the user program with "Set break point", please note that it is not possible to resume the program with the FM 357-2 after the break point is reached (CPU/FM 357-2 monitoring is activated as a result).

The program can only be continued by restarting (CPU: STOP/RUN).

Information for pulse enable

A prerequisite to activate the axes (even for simulation) is the pulse enable. When connecting a drive, this signal is transmitted via PROFIBUS DP to the drive and activated there (SIMODRIVE 611-U).

Information for error evaluation

The FM 357-2 and the FCs use monitoring and diagnostic routines with associated error responses/messages. These must be included in the user program (see Section 12.2).

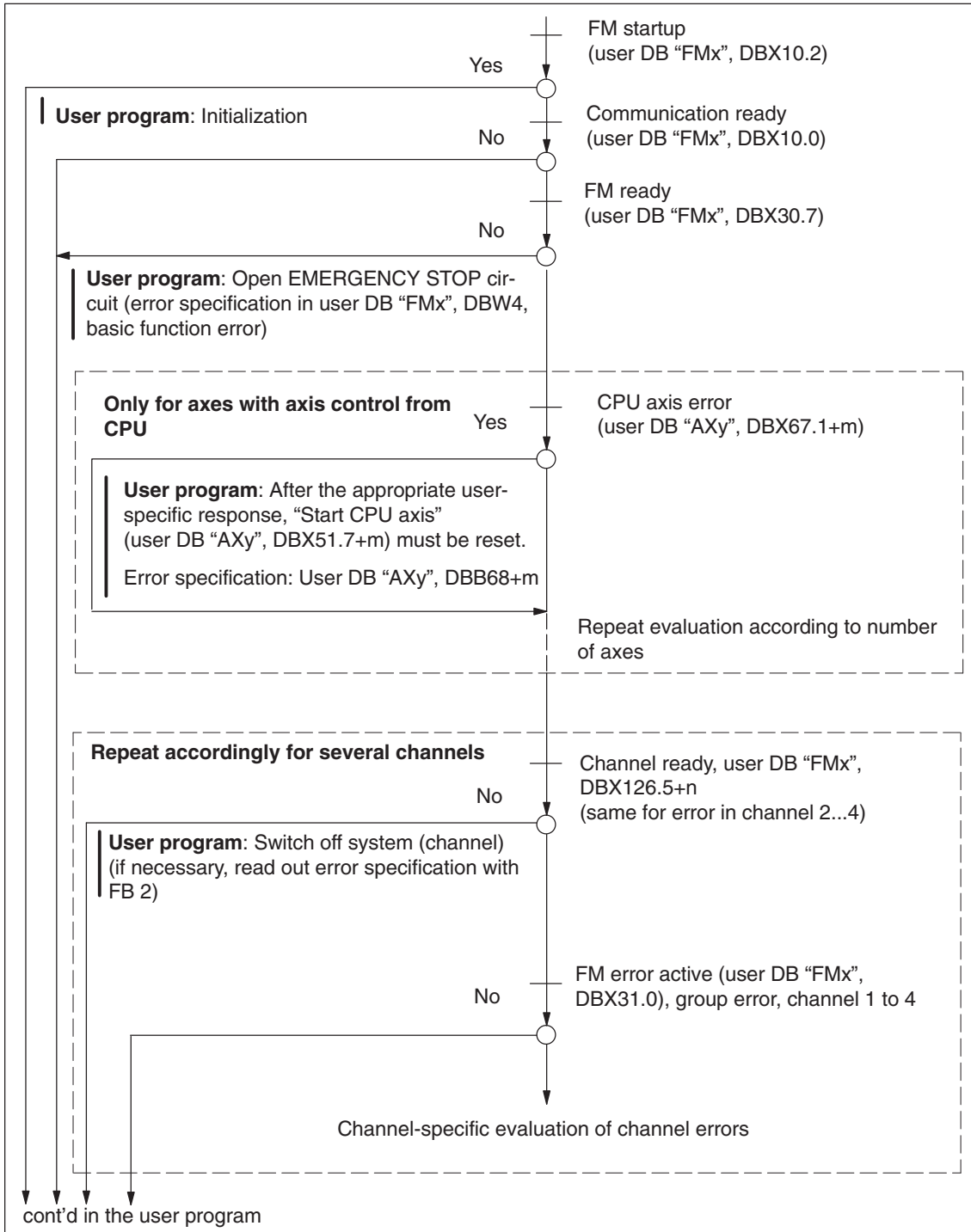
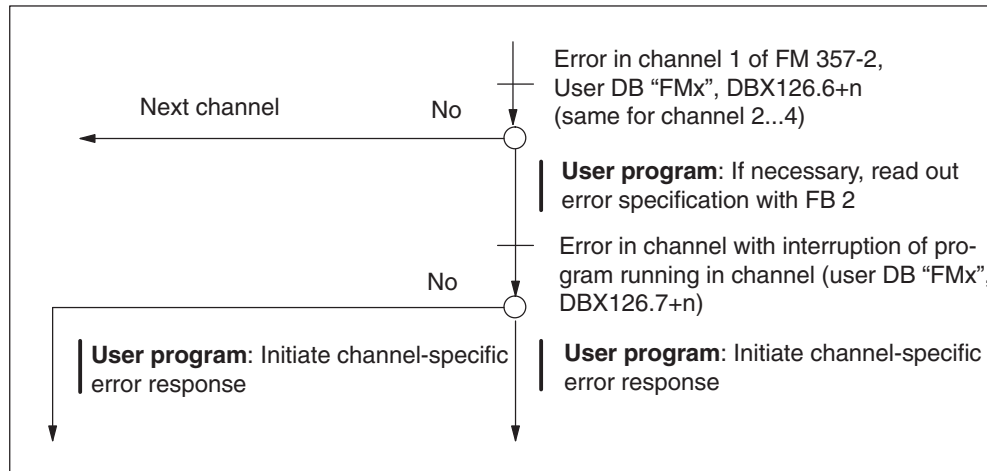


Fig. 6-2 Error evaluation (see also example 7, Section 6.8.7)

Channel-specific evaluation



Note

The errors which result in the resetting of the “communication ready” and/or “FM ready” signals must be acknowledged by “FM restart” or by switching the FM 357-2 off/on.

The other errors must be acknowledged by “Reset” (user DB “FMx”, DBX108.7+n) or “CANCEL” (FB 4).

The error evaluation with FB 2, 3, 4 response should be performed each time the corresponding FB is called.

Error message FB 2, FB 3, FB 4	Further specification of the message	Error response	Error acknowledgement	Meaning
As output parameter of FB 2, 3, 4	Error code as output parameter	(User program: according to error code)	Reset function trigger (input parameter)	See block description

Since FBs 2, 3 and 4 organize variable exchange and services (the actual transfer takes place in FC 22), the errors reported as output parameters are mainly errors associated with the variables and their organization.

6.1.7 Procedure for writing the user program

The library “FM357-2L”, which is supplied as part of the configuration package, can be used as a template for writing a user program. The “USER PROGRAM” section of OB 1 contains a user program for controlling the axes with the aid of the “Parameterize FM 357-2” tool.

Suggested procedure:

1. Open your project in the SIMATIC Manager.
2. Select **SIMATIC xxx > CPUxxx > S7 Program**
3. Open the “FM357-2L” library in the SIMATIC Manager with **File > Open... > Libraries**.
4. Select directory “BF_V3...” in the library.
5. Select the file “Symbols” and copy it into your project under **SIMATIC xxx > CPUxxx > S7 Program** (replace the existing object).
6. Open the “sources” directory and copy the statement list source file “fm357ob_n1” (for one FM) or “fm357ob_n2” (for two FMs) into the “sources” directory of your project. For more than two FMs use the STL source “fm357ob_n2” and increase the basic function calls according to the number of modules.
7. Open the “blocks” directory and copy all the blocks from there into the “blocks” directory of your project (including the UDT blocks).
8. Select the “sources” directory in your project. Double-click the selected “fm357ob_n1” file to start the “LAD/STL/FBD editor”.
9. **Modify the input parameters to suit your application in OB 100 in the FC 1 call and in OB 82 in the FC 5 call (see Section 6.3.1, FC 1 and Section 6.3.2, FC 5).**
10. **Insert your user program in OB 1 (in the USER PROGRAM section). You can insert the functions you need for your application here from the example project “zEn16_01_FM357-2_BF_EX” (examples 1 to 7, see Section 6.8). Activate the functions in your user program by setting/clearing the signals in DB 115 (this user DB is provided for the examples).
Modify the input parameter for FC 22 accordingly .**
11. Select menu commands **File > Save** and **File > Compile** to generate the organization blocks (OB 1, OB 82, OB 100) from the STL source code. (you can ignore the warnings from the compiler)
12. Close the editor.
13. Switch the CPU to “STOP” and switch the CPU on.
14. In the SIMATIC Manager select **SIMATIC xxx > CPUxxx > S7 Program > Blocks**.
15. Load all the S7 blocks contained here onto your CPU (including the system data) with **PLC > Load** (the CPU must be in STOP mode).
16. Switch the CPU to RUN. After approximately one minute, the yellow “DIAG” LED flashes periodically on the FM 357-2, indicating that the start up of the CPU and FM 357-2 has been successfully completed.

6.2 Startup with the parameterization tool “Parameterize FM 357-2”

General

For testing and startup with the “Parameterize FM 357-2” tool (Startup window), the CPU must be switched to RUN with an appropriate user program (standard function blocks must be available).

The supplied STL source file “fm357ob_n1” in library “FM357-2L” already contains an example call in OB 1 in the “USER PROGRAM” section for controlling the axes from the “startup window” (for information on how to handle the creation/installation of the user program, see “FM 357-2 First Steps”, the brief startup guide or Section 6.1).

Startup

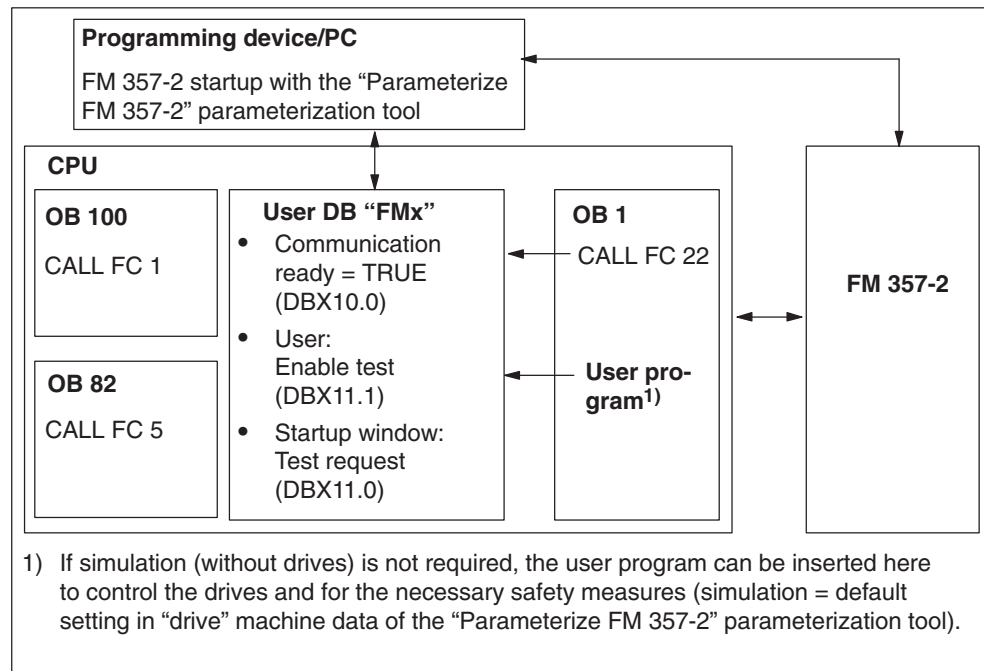


Fig. 6-3 Startup with the “Parameterize FM 357-2” tool (overview)

The user program and the “Startup window” use the same interface to control the FM 357-2. They can only be used as alternatives.

The signals can be read by the user program.

The user program uses the “enable test” as an interlock signal to specify that the interface for controlling the FM 357-2 is allowed to use the “startup window”. When the user program sets signal “Enable test” = TRUE and the startup user interface sets signal “Test request” = TRUE (“Test” button activated in startup window), the FM 357-2 is controlled via the “Startup window” (test mode).

In the supplied library, the bit “ACT_TEST (enable test)” = TRUE is set in OB 1 if the bit of the parameterization user interface “PRES_TEST (test request)” = TRUE.

Interface signal	by user ...	Meaning
Startup window: Test request	Scanned on request	Start test mode with “Parameterize FM 357-2”. The signal is set/cleared in the “Startup” window.
User: Enable test	Set/ cleared	Enable test mode from user program (signal is only required for test mode with parameterization tool)

If startup is deselected (test request), the signals set in the “Startup window” are not cleared.

Exception:

- The “Feed Stop” signal (user DB, “AXy”, DBX4.3+m) is set if desired by the “Parameterize FM 357-2” user (prompt window) on deselection of Test/Startup.
- The signals “Rapid traverse override active”, “Feed override active” (user DB, “FMx”, DBX107.6/7+n) and “Activate override” (user DB, “AXy”, DBX1.7+m) are set to the status valid on selection.

6.3 Description of the standard function blocks

Overview

In this section, you can find information about:

- FC 1: RUN_UP – Startup / initialization, Section 6.3.1, page 6-17
- FC 5: BF_DIAG – Diagnostic alarm and FM restart, Section 6.3.2, page 6-19
- FC 22: BFCT – Basic functions and operating modes, Section 6.3.3, page 6-22
- FB 2: FM_GET – Read FM variable, Section 6.3.4, page 6-24
- FB 3: FM_PUT – Write FM variable, Section 6.3.5, page 6-32
- FB 4: FM_PI – General services, Section 6.3.6, page 6-38
- DB 121: VAR_ADDR – Extract from FM variable list, Section 6.3.7, page 6-45

6.3.1 FC 1: RUN_UP – Startup/initialization

Task

Initialization and generation of the appropriate application interface (if it does not yet exist) and user DBs “FMx” and “AXy”, together with the DB no. according to the input parameters FMDB_NO for user DB “FMx” and FMDB_NO+1 for user DB “AXy” (**N.B.:** 2 consecutive DB numbers are always required for the interface).

If a suitable DB no. already exists, e.g. because the CPU was not reset, the length of the DBs is verified. If the DB length is correct, the contents are deleted (the interface area is reset to the original state).

If the DB length is incorrect (DB with “FMDB_NO” or “FMDB_NO+1” already exists), the CPU switches to “STOP” (see the error messages in this chapter).

Call options

FC 1 must be called once in OB 100 for each FM 357-2 and supplied with the appropriate parameters.

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
	<pre>CALL FC 1 FMDB_NO := FMLADDR := FMCOMM := FMSYNC_IF := USERVERS :=</pre>

Description of parameters

The following table describes the parameters of FC 1.

Table 6-2 Parameters of FC 1

Name	Data type	I/O type	Value range	Significance
FMDB_NO	INT	I	1... see CPU	No. of assigned user DB "FMx"
FMLADDR	INT	I	256...752	I/O address of FM according to entry in S7 hardware configuration
FMMCOMM ¹⁾	BOOL	I	TRUE = Services required FALSE = Services not required	Activate CPU-FM communication services (FB 2, 3, 4) if services required
FMSYNC_IF ²⁾	BOOL	I	TRUE = yes FALSE = no	Interface synchronization with internal IPO cycle (internal processing cycle of FM)
USERVERS ³⁾	INT	I	000...999	Version of user program for display in parameterization tool

Parameter types: I = input parameter

- 1) If these services are not required ("CANCEL" error messages can be cleared with the "Reset" interface signal), blocks FB 2, 3, 4, 6, DB 15, 121 do not have to be included in the user program (saves memory space, see Section 6.9 "Technical Data").
- 2) Interface communication between the CPU/FM357-2 (runtime in the FC 22) can take several milliseconds, depending on the application (e.g. interruption of communication by other execution levels, see Section 6.9 "Technical Data"). This can cause the signals on the FM to be processed in different IPO cycles. When the "FMSYNC_IF" parameter is set, the control signals (channel and axis) in the FM 357-2 are passed to the IPO cycle after completion of communication, and processing is thus simultaneous (e.g. simultaneous starting of axes and channels, see also Section 6.7).
- 3) max. 3 digits, display in "Parameterize FM 357-2" tool = 0.00...9.99

Error messages

For error message, see system error message "module status" CPU.

If the CPU switches to STOP mode again on startup and indicates a group error, an error has occurred during initialization of the interface. Proceed as follows to read out the error code stored in the accumulator of the CPU:

1. In your project, select the menu command **PLC > Module Status** to open the "Module Status" dialog of the CPU.
2. Select the "Diagnostic Buffer" tab in this window. Select the event "STOP by STOP command" (most recent entry). You will find a more detailed description of the error in the event details (e.g. caused by OB: 100, FC number: 1).
3. If these entries exist, select the "Stacks" tab. Select OBxxx and activate the "U-Stack ..." button. You can read out the accumulator contents in the dialog that follows:
 - ACCU 1 = user DB number ("FMx" or "AXy")
 - ACCU 2 = error code

The following table shows the errors and their causes.

Table 6-3 FC 1 error messages

Error code	Significance	FC 1 error response	Error cause
W#16#0201	Serious error in user program during startup	CPU switches to STOP	Error in DB number: <ul style="list-style-type: none"> • Number is 0 • Number is greater than the maximum number of DBs for this CPU
W#16#0202			<ul style="list-style-type: none"> • FMDB_NO or FMDB_NO+1 already allocated
W#16#0203			<ul style="list-style-type: none"> • Cannot generate user DBs, no more memory available

For example call see STEP 7 library “FM357_2L\fm357ob_nx” in OB 100

6.3.2 FC 5: BF_DIAG – Diagnostic alarm and FM restart

Task

FC 5 detects the incoming/outgoing diagnostic alarm from the respective FM 357-2 (address in local variable OB82_MDL_ADDR) and initiates the appropriate responses (see Table 6-5).

Call options

FC 5 must be called once in OB 82 for each FM 357-2 and supplied with the appropriate parameters.

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
	<pre>CALL FC 5 FMDB_NO :=</pre>

Description of parameters

The following table describes the parameter of FC 5.

Table 6-4 Parameters of FC 5

Name	Data type	I/O type	Meaning	by user ...	by block ...
FMDB_NO	INT	I	No. of assigned user DB “FMx”	entered	scanned

Parameter type: I = input parameter

Diagnostic alarm

The error code is stored in the interface (user DB “FMx”, DBW4 “basic function error”).

Diagnostic alarms are also indicated by the “module status” system error message of the CPU.

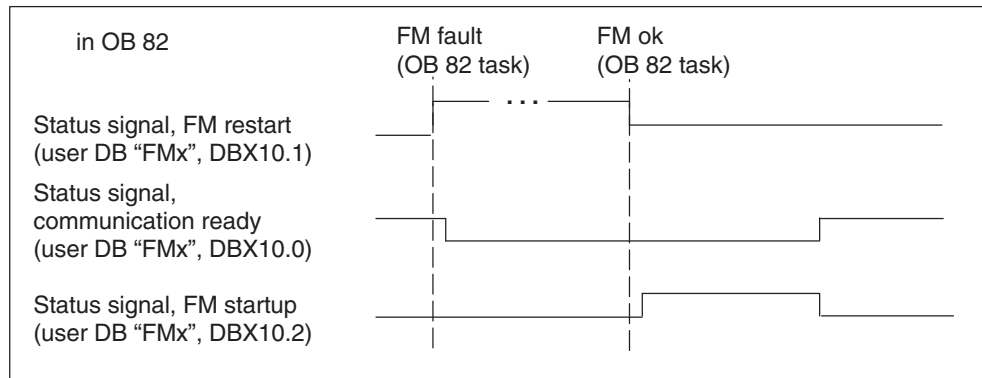
The following table includes diagnostic alarms, FM 357-2 faults or faults in the signal modules on the local P bus.

Table 6-5 Diagnostic alarms

Error code	Significance	FC 5 error response	Error cause
W#16#0010	Diagnostic alarm, FM restart (RESTART, user DB “FMx”, DBX10.1) (OB82_MDL_STOP)	Set FM restart, clear interface in FC 22, see timing chart	Reset FM, (no error)
W#16#0011	Diagnostic alarm, hardware error FM (OB82_INT_FAULT) ¹⁾	Reset communication ready signal	Internal error on FM
W#16#0012	External error “Local P bus segment” (OB82_EXT_FAULT) ¹⁾	None, (FM resets “FM ready”)	Error in local area of FM, (local modules)
W#16#0013	Diagnostic alarm, watchdog timeout (OB82_WTCH_DOG_FLT) ¹⁾	Reset communication ready signal	Internal error on FM
W#16#0014	Diagnostic alarm, internal FM power supply fault (OB82_INT_PS_FLT) ¹⁾	Reset communication ready signal	<ul style="list-style-type: none"> • Internal error on FM • Power supply failure
W#16#0015	Diagnostic alarm, K bus/MPI communication fault	None, (FM resets “FM ready”)	Internal error on FM, K bus/MPI fault

1) Local variable in OB 82

Timing chart for FM restart



If an FM restart is initiated in the FM (via OP or "Parameterize FM 357-2" tool, e.g. after modification of machine data), a corresponding diagnostic alarm is output (OB82_MDL_STOP), detected by FC 5 and evaluated in basic function program FC 22. After the response routine has been started, an outgoing diagnostic alarm signals the reaction on the FM.

The FM is reset on the FM restart (new startup in FC 22) and the interface (user DBs "FMx" and "AXy") is cleared and initialized.

6.3.3 FC 22: BFCT – Basic functions and operating modes

Task

FC 22 is the central block for communication between the user and the FM 357-2.

This function comprises the following:

- Startup synchronization with the FM 357-2
- Basic function operation between CPU and FM 357-2 (including multi-channel operation and for startup/testing with the “Parameterize FM 357-2” tool).

If the communication ready status is active for communication between the FM 357-2 and CPU, the signals/data of the interface are transferred from the FC 22 to the FM 357-2 via I/O (automatic) or with Read/write data (by user request), and read by the FM 357-2.

Other functions are as follows:

- Reset complete interface on FM restart. A new runup is started.
- Transfer of data/parameters that are activated with FB 2, 3, 4.

If the call and the task are initiated simultaneously, please make sure that the data transfer triggered by the read/write request is executed in the following order:

1. In the order of the FB 2, 3, 4 calls
2. Read/write task

- Monitoring of the interface between CPU and FM 357-2

Call options

The block must be called once in the cyclic program (OB 1) for each FM 357-2.

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
	<pre>CALL FC 22 FMDB_NO :=</pre>

Description of parameter

The following table shows the parameter associated with FC 22.

Table 6-6 FC 22 parameters

Name	Data type	I/O type	Meaning	by user ...	by block ...
FMDB_NO	INT	I	No. of assigned user DB “FMx”	entered	scanned

Parameter type: I = input parameter

FC 22 troubleshooting

For error code, see user DB "FMx", DBW4

Table 6-7 FC 22 troubleshooting

Error code	Significance	FC 22 error response	Error cause
W#16#0100	Unknown interface for the FM	Status signals: "FM startup" = FALSE "communication ready" remains = FALSE	e.g. incorrect firmware version on FM, error in FM firmware
W#16#0101	Powerup timeout (3 min)	Status signals: "FM startup" = FALSE "communication ready" remains = FALSE	<ul style="list-style-type: none"> No connection to FM or connection aborted Incorrect FM firmware version Error in FM firmware see system error message (module status)
W#16#0102	Cyclic sign-of-life or transfer timeout	Status signal: "communication ready" is reset	<ul style="list-style-type: none"> Connection to FM aborted Section 6.1.6 "CPU/FM 357-2 monitoring" not observed see system error message (module status CPU)
W#16#0103	Error in the data transfer between FM used in the distributed mode and the CPU	Task not executed Status signal: "Communication ready" is reset	<ul style="list-style-type: none"> Connection to the FM aborted Section 6.1.6 "Monitoring of the CPU/FM 357-2" not observed see system error message (module status of CPU)
W#16#0104	Wrong variant of the basic functions	Job is not executed Status signal: "Readiness for communication" is reset	<ul style="list-style-type: none"> Check your hardware configuration (FM 357-2 with the note "also for distributed applications" must also be configured for distributed application) Check the blocks, which you have chosen from the FM357_2 library, for your particular application (Section 6.1.1)
W#16#0105	Read configuration data not correct (number of channels/axes)	Status signals: "FM start" = FALSE "Readiness for communication" remains = FALSE	Restart your installation by CPU STOP after RUN

Notice

The signals and their assignment are described in Section 6.6.

6.3.4 FB 2: FM_GET – Read FM variable

Task

FB 2 can be used to read variables (e.g. machine data, R parameters, error codes, velocities) from the FM 357-2.

The FB 2 call and control input “REQ” starts a request to read the NC variables referenced by “ADDR1...ADDR8”, and, after the read operation, to copy them into the CPU operand areas referenced by “RD1...RD8” (a user DB is recommended). Variables which are not required (for quantities < 8) are not filled in.

The successful completion of the read operation is indicated by the TRUE condition of the “NDR” status parameter.

The read operation may last for several CPU cycles (generally 2 to 3 cycles for central configuration).

Any errors which occur are indicated by “ERROR” and “STATE”.

Parameters “UNIT1...UNIT8”, “COLUMN1...COLUMN8” and “LINE1...LINE8” are optional parameters which only have to be specified for the corresponding variables (Addr. 1...8).

The read operation is executed in the FC 22 block after any read requests activated there.

A user DB is required as an instance or a multi-instance DB (for storing the parameters/values of the variables).

Addressing and handling of variables

An extract of the possible variables is summarized in an FM-VAR list, which is provided, and is generated and stored in DB 121 (see Section 6.3.7).

Group combination of variables

Only FM variables within a group can be combined in a request.

	Area				
Group 1	C[1]	N	B	O	T
Group 2	C[2]	–	–	–	–
Group 3	C[3]	–	–	–	–
Group 4	C[4]	–	–	–	–

N = FM variable; B/C = Channel variable; A = Axis; T = Tools

The variables of channel 1 can be combined with the variables from the N/B/A/T areas and read in the same request. Variables from channels 2 to 4 may **not** be combined in a request with variables from areas N/B/A/T.

Call options

The block should be called cyclically (OB 1). Before you start a new request, you must call FB 2 at least once with control input "REQ" = FALSE.

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
FB 2	CALL FB 2
— EN — ENO —	REQ :=
— REQ — ERROR —	NUMVAR :=
— NUMVAR — NDR —	ADDR1 :=
— ADDR1 — STATE —	UNIT1 :=
— UNIT1 —	COLUMN1 :=
— COLUMN1 —	LINE1 :=
— LINE1 —	ADDR2 :=
— ADDR2 —	UNIT2 :=
— UNIT2 —	COLUMN2 :=
— COLUMN2 —	LINE2 :=
— LINE2 —	... :=
— ... —	ADDR8 :=
— ADDR8 —	UNIT8 :=
— UNIT8 —	COLUMN8 :=
— COLUMN8 —	LINE8 :=
— LINE8 —	FMDB_NO :=
— FMDB_NO —	ERROR :=
— RD1 —	NDR :=
— RD2 —	STATE :=
— ... —	RD1 :=
— RD8 —	RD2 :=
	... :=
	RD8 :=

Notice

A maximum of 3 FM services (FB 2, 3, 4 calls) can be started in each OB1 cycle.

FB 2 can only read variables if parameter "FMCOMM" has been set to TRUE (in OB 100: FC 1).

Input parameter "REQ" must not be reset until output parameter "NDR" or "ERROR" = TRUE (see Figure 6-4).

If communication between CPU and FM is aborted (FB 2, 3, 4) by POWER OFF or FM restart, the start requests must be deleted in the first OB 1 pass after startup (call FB 2/3/4 with signal REQ = FALSE).

Description of parameters

The following table describes the parameters of FC 2.

Table 6-8 FB 2 parameters

Name	Data type	I/O type	Value range	Meaning
REQ	BOOL	I	–	Start request on positive signal
NUMVAR	INT	I	1...8 (corresponds to the use of ADDR1...ADDR8)	Number of variables to be read
ADDR1...ADDR8	ANY	I	vardbname.varname	Variable names (e.g. from DB 121)
UNIT1...UNIT8	BYTE	I	–	Optional for corresponding variable, see FM-VAR list
COLUMN1... COLUMN8	WORD	I	–	Optional for corresponding variable, see FM-VAR list
LINE1...LINE8	WORD	I	–	Optional for corresponding variable, see FM-VAR list
FMDB_NO	INT	I	–	No. of assigned user DB "FMx"
ERROR	BOOL	O	–	Request was given negative acknowledgement or could not be executed
NDR	BOOL	O	–	Request was successfully executed. Data are available
STATE	WORD	O	–	See "Troubleshooting"
RD1 to RD8	ANY	I/O	dbname.varname or P#Mn.n BYTEEnum or P#DBnr.DBXm.n BYTEEnum	Destination area for read data (do not use local variables)

Parameter types: I = input parameter, O = output parameter, I/O= I/O parameter

Troubleshooting

Status parameter “ERROR” is set to TRUE if it is not possible to execute a request. The cause of the error is encoded at the “STATE” block output. The error code is deleted when the start signal is reset (FB 2 call with “REQ” = FALSE) after an error message.

Table 6-9 FB 2 troubleshooting

W#16#...	STATE		Meaning	Note
	High byte	Low byte		
xx01	1...8	1	Access error	The high byte contains the number of the variable in which the error has occurred
0002	0	2	Error in request	Incorrect combination of variables in a request (please follow the rules for group combinations of variables)
0003	0	3	Negative acknowledgement, request not executable	Internal error, remedy: Reset
xx04	1...8	4	Not enough local user memory available	Read variable is longer than specified in RD1 to RD8; high byte contains the number of the variable in which the error has occurred
0005	0	5	Format conversion error	Error in converting variable type double: Variable is not in the S7-REAL range
0006	0	6	Serial request buffer is full	The request must be repeated because the queue is full (max. 3 FM services can be called).
0007	0	7	Variable does not exist	“FMCOMM” parameter is not set
xx08	1...8	8	Invalid destination area (RD)	“RD1...RD8” must not be local data
xx0A	1...8	10	Error in addressing a variable	“UNIT” or “COLUMN/LINE” contains value 0
000B	0	11	Variable address of FM 357-2 invalid	Check the following: <ul style="list-style-type: none"> • Variable name in the generated DB of the NC-VAR Selector • Parameters “ADDR”, “UNIT” and “COLUMN” • Parameter “LINE” for indirect addressing
000C	0	12	Parameter “NUMVAR” invalid	Check input parameter “NUMVAR” (see Tab. 6-8, Value range)
000D	0	13	Task terminated by FM restart	After restart of the FM, you must initialize the FB with REQ = FALSE. Then you can start a new job with REQ = TRUE.
000E	0	14	Parameter “FMDB_NO” is zero	Check the input parameter “FMDB_NO” and enter the correct user DB number.

Timing diagram

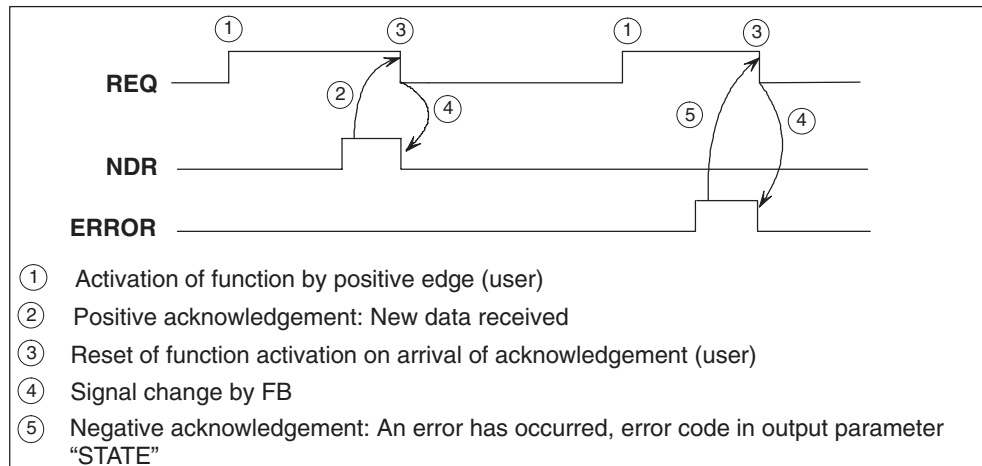


Fig. 6-4 Timing diagram for FB 2

NC-VAR Selector

If you need further data/variables of the FM 357-2 which are not available in the selected FM variable list (in DB 121), you must install the NC-VAR selector. You can use the NC-VAR selector to select the variables you need for your application and add them to the FM variable list.

The NC-VAR selector is included in the Configuring Package.

How you work with the NC-VAR selector is described in the following:

- In the “NC-VAR Selector” tool (online help)
 - This tool is designed for use on a control family. Use only the variables which are relevant for your application.
- or
- Description of Functions *Basic Machine (Part 1), Basic PLC Program (P3)* Order No.: 6FC5 297-6AC20-0BP1

Installation: You install the Windows application **NC-VAR Selector** using the **SETUP** program supplied with the software.

To add a variable to DB 121, proceed as follows:

1. Select the NC-VAR Selector.
2. Open the file DB121.var in the NC-VAR Selector, which is to be found in the [STEP7 directory]\S7LIBS\FM357_2L.
3. Select the **Complete list > Select** menu to open the **Select complete list** dialog box.
4. Selected there the directory [nc_var directory]\data\sw[version] (e.g. SW64).
 - Open the displayed file ncv_NcData.mdb.
5. Select the variables you need for your project (use Help).
 - You may need to assign the variables an axis number and/or a channel number or appropriate other parameters.
 - Click on OK to accept the selected variables into your project.
6. Save the DB121.var.
7. Select the menu **Code > Settings** and enter there the DB number 121 under **DB settings**.
8. Use the **Code > Generate** menu to create a STEP 7 source file DB121.awl, which contains a DB in the ASCII format.
9. Using the **Code > To STEP7 project** menu, the STEP 7 source file DB121.awl is transferred into the selected project into the directory **Sources**, and the block is compiled.
 - As a result, DB 121 is created with the relevant address specifications.
10. Parameterize FB 2.

Data types

The data types of the FMs are listed with the variables in the NC-VAR selector.

The following table shows the data type assignments to S7 data types.

FM data type	S7 data type
double	REAL
float	REAL
long	DINT
integer	DINT
uint_32	DWORD
int_16	INT
uint_16	WORD
unsigned	WORD
char	CHAR or BYTE
string	STRING
bool	BOOL

For an example call, see also example application 5 in Section 6.8.5

In the SIMATIC Manager, select **File > Open... > Project** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Quellen\EXAMPLE5”.

Extract from symbol table:

Symbol	Address	Data type	Comments
FM_GET	FB 2	FB 2	Read FM variable
DI_FB2	DB 117	FB 2	Instance DB for FB 2
USERDB	DB 115	DB 115	User DB for examples
VAR_ADDR	DB 121	DB 121	Extract from FM variable list

```

STL
U   "USERDB".EX5.VAR_RD           // Start request
FP  "USERDB".EX5.HBIT1           // Edge flag
S   "USERDB".EX5.FB2_REQ         // Start request FB 2

O   "USERDB".EX5.NDR             // Request terminated
O(
U   "USERDB".EX5.ACKN_ERR_RD     // Error acknowledgement
U   "USERDB".EX5.ERR_RD         // FB 2 execution error
)
R   "USERDB".EX5.FB2_REQ         // Reset FB 2 request
R   "USERDB".EX5.VAR_RD         // Reset request start

CALL "FM_GET" , "DI_FB2"        // FB 2
REQ  := "USERDB".EX5.FB2_REQ     // Start function FB 2
NUMVAR := 4                      // Number of variables
ADDR1 := "VAR_ADDR".SET_POS     // Structure: position setpoint
UNIT1 := B#16#1                 // Index UNIT: Channel 1
COLUMN1 :=
LINE1 := W#16#3                 // Index LINE: Axis number 3
ADDR2 := "VAR_ADDR".CMD_FDRATE  // Structure for velocity
                                   setpoint // (positioning axis)
UNIT2 := B#16#1                 // Index UNIT: Channel 1
COLUMN2 :=
LINE2 := W#16#3                 // Index LINE: Axis number 3
ADDR3 := "VAR_ADDR".R_PARAM     // Structure: R parameters
UNIT3 := B#16#1                 // Index UNIT: Channel 1
COLUMN3 :=
LINE3 := W#16#2                 // Index LINE: R parameter 1
                                   // (number + 1)
ADDR4 := "VAR_ADDR".PATH_OVERR  // Structure: Path override
UNIT4 := B#16#1                 // Index UNIT: Channel 1
COLUMN4 :=
LINE4 :=
ADDR5 :=
UNIT5 :=
COLUMN5 :=
LINE5 :=
ADDR6 :=
UNIT6 :=
COLUMN6 :=
LINE6 :=
ADDR7 :=
UNIT7 :=
COLUMN7 :=
LINE7 :=
ADDR8 :=
UNIT8 :=
COLUMN8 :=
LINE8 :=
FMDB_NO:=21                     // Assigned number for user DB
ERROR := "USERDB".EX5.ERR_RD    // "FMx"
NDR   := "USERDB".EX5.NDR       // Request error
STATE := #w_STATE              // Request finished message
RD1   := "USERDB".EX5.SET_POSITION // FB error number
RD2   := "USERDB".EX5.FDRATE    // Position setpoint target range
RD3   := "USERDB".EX5.RPARAM_1  // Velocity setpoint target
RD4   := "USERDB".EX5.OVERRIDE  // range.
RD5   :=                        // Target range for R parameters
RD6   :=                        // Target range for path override
RD7   :=
RD8   :=

L   #w_STATE                    // On error, enter
T   "USERDB".EX5.STATE_RD       // error status ... in "USERDB"

```

6.3.5 FB 3: FM_PUT – Write FM variable

Task

FB 3 can be used to write variables (e.g. machine data, R parameters) to the FM 357-2.

A request is started by calling FB 3 and with control input “REQ”. The variables referenced by “ADDR1...ADDR8”, with the data/values stored in CPU operand areas (a user DB is recommended) referenced by “SD1...SD8”, are written.

Successful completion of the write operation is indicated by status parameter “DONE” = TRUE .

The write operation may last for several CPU cycles (generally 1 to 3 cycles for central configuration).

Any errors which occur are indicated by “ERROR” and “STATE”.

Parameters “UNIT1...UNIT8”, “COLUMN1...COLUMN8” and “LINE1...LINE8” are optional parameters which only have to be specified for the corresponding variables.

The write operation is executed in the FC 22 block after any write requests activated there.

A user DB is required as an instance DB or a multi-instance DB (for storing the parameters/values of the variables).

Addressing and handling of variables

An extract of the possible variables is summarized in an FM-VAR list, which is provided, and is generated and stored in DB 121 (**see Section 6.3.7**).

Group combination of variables

Only FM variables within a group can be combined in a request.

	Area				
Group 1	C[1]	N	B	O	T
Group 2	C[2]	–	–	–	–
Group 3	C[3]				
Group 4	C[4]	–	–	–	–

N = FM variable; B/C = Channel variable; A = Axis; T = Tools

The variables of channel 1 can be combined with the variables from the N/B/A/T areas and written in the same request. Variables from channels 2 to 4 may **not** be combined in a request with variables from areas N/B/A/T.

Call options

The block should be called cyclically (OB 1). Before you start a new request, you must call FB 3 at least once with control input "REQ" = FALSE.

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
FB 3	CALL FB 3
— EN	REQ :=
— REQ	NUMVAR :=
— NUMVAR	ADDR1 :=
— ADDR1	UNIT1 :=
— UNIT1	COLUMN1 :=
— COLUMN1	LINE1 :=
— LINE1	ADDR2 :=
— ADDR2	UNIT2 :=
— UNIT2	COLUMN2 :=
— COLUMN2	LINE2 :=
— LINE2	...
— ...	ADDR8 :=
— ADDR8	UNIT8 :=
— UNIT8	COLUMN8 :=
— COLUMN8	LINE8 :=
— LINE8	FMDB_NO :=
— FMDB_NO	ERROR :=
— SD1	DONE :=
— SD2	STATE :=
— ...	SD1 :=
— SD8	SD2 :=
	.. :=
	SD8 :=

Notice

A maximum of 3 FM services (FB 2, 3, 4 calls) can be started in each OB1 cycle.

FB 3 can write variables only if parameter "FMCOMM" has been set to TRUE (in OB 100: FC 1).

Input parameter "REQ" must not be reset until output parameter "DONE" or "ERROR" = TRUE (see Figure 6-5).

If communication between CPU and FM is aborted (FB 2, 3, 4) by POWER OFF or FM restart, the start requests must be deleted in the first OB 1 pass after startup (call FB 2/3/4 with signal REQ = FALSE).

Description of parameters

The following table describes the parameters of FC 3.

Table 6-10 FB 3 parameters

Name	Data type	I/O type	Value range	Meaning
REQ	BOOL	I	–	Start request on positive signal
NUMVAR	INT	I	1...8 (corresponds to the use of ADDR1...ADDR8)	Number of variables to be written
ADDR1...ADDR8	ANY	I	vardbname.varname	Variable names (e.g. from DB 121)
UNIT1...UNIT8	BYTE	I	–	Optional for corresponding variable, see FM-VAR list
COLUMN1... COLUMN8	WORD	I	–	Optional for corresponding variable, see FM-VAR list
LINE1...LINE8	WORD	I	–	Optional for corresponding variable, see FM-VAR list
FMDB_NO	INT	I	–	No. of assigned user DB “FMx”
ERROR	BOOL	O	–	Request was given negative acknowledgement or could not be executed
DONE	BOOL	O	–	Request was successfully executed. Data are available
STATE	WORD	O	–	See “Error evaluation”
SD1 to SD8	ANY	I/O	dbname.varname or P#Mn.n BYTEnum or P#DBnr.DBXm.n BY- TEnum.	Destination area for write data (do not use local variables)

Parameter types: I = input parameter, O = output parameter, I/O= I/O parameter

Troubleshooting

Status parameter “ERROR” is set to TRUE if it is not possible to execute a request. The cause of the error is encoded at the “STATE” block output. The error code is deleted when the start signal is reset (FB 3 call with “REQ” = FALSE) after an error message.

Table 6-11 FB 3 troubleshooting

STATE		Meaning	Note
W#16#...	High byte Low byte		
xx01	1...8 1	Access error	The high byte contains the number of the variable in which the error has occurred.
0002	0 2	Error in request	Incorrect combination of variables in a request (please follow the rules for group combinations of variables)
0003	0 3	Negative acknowledgement, request not executable	Internal error, remedy: Reset
xx04	1...8 4	Data area or data type mismatch	Check data to be written in SD1 to SD8; high byte contains the number of the variable in which the error has occurred
0006	0 6	Serial request buffer is full	The request must be repeated because the queue is full (max. 3 FM services can be called).
0007	0 7	FB 3 not allowed	“FMCOMM” parameter is not set
xx08	1...8 8	Invalid source area (SD)	“SD1...SD8” cannot be local data
xx0A	1...8 10	Error in addressing a variable	“UNIT” or “COLUMN/LINE” contains value 0
000B	0 11	Variable address of FM 357-2 invalid	Check the following: <ul style="list-style-type: none"> • Variable name in the generated DB of the NC-VAR Selector • Parameters “ADDR”, “UNIT” and “COLUMN” • Parameter “LINE” for indirect addressing
000C	0 12	Parameter “NUMVAR” invalid	Check input parameter “NUMVAR” (see Tab. 6-10, Value range)
000D	0 13	Task terminated by FM restart	After restart of the FM, initialize the FB with REQ = FALSE. Then you can restart a new job with REQ = TRUE.
000E	0 14	Parameter “FMDB_NO” is zero	Check the input parameter “FMDB_NO” and enter the correct user DB number.

Timing diagram

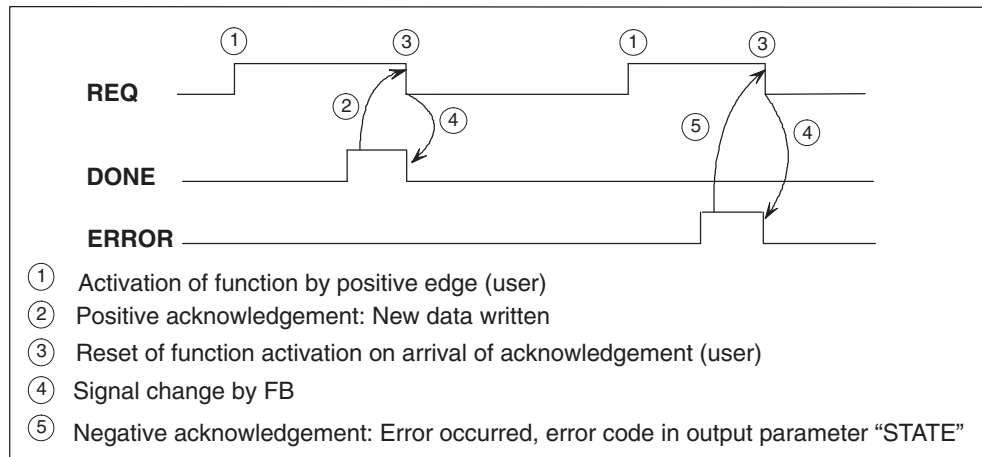


Fig. 6-5 Timing diagram for FB 3

NC-VAR selector

see Section 6.3.4 below "NC-VAR Selector" line

For an example call, see also example application 5 in Section 6.8.5

In the SIMATIC Manager, select **File > Open... > Project** and open the example project "zEn16_01_FM357-2_BF_EX\EXAMPLES\Quellen\EXAMPLE5".

Extract from symbol table:

Symbol	Address	Data type	Comments
FM_PUT	FB 3	FB 3	Write FM variable
DI_FB3	DB 118	FB 3	Instance DB for FB 3
USERDB	DB 115	DB 115	User DB for examples
VAR_ADDR	DB 121	DB 121	Extract from FM variable list

```

STL
U   "USERDB".EX5.VAR_WR           // Start request
FP  "USERDB".EX5.HBIT2           // Edge flag
S   "USERDB".EX5.FB3_REQ         // Start request FB 3

O   "USERDB".EX5.DONE            // Request terminated
O(
U   "USERDB".EX5.ACKN_ERR_WR     // Error acknowledgement
U   "USERDB".EX5.ERR_WR         // FB 3 execution error
)
R   "USERDB".EX5.FB3_REQ         // Reset FB 3 request
R   "USERDB".EX5.VAR_WR         // Reset request start

CALL "FM_PUT" , "DI_FB3"        // FB 3
  REQ   := "USERDB".EX5.FB3_REQ // Start function FB 3
  NUMVAR := 3                    // Number of variables
  ADDR1  := "VAR_ADDR".R_PARAM  // Structure of R parameters
  UNIT1  := B#16#1              // Index UNIT: Channel 1
  COLUMN1 :=
  LINE1  := W#16#1              // Index LINE: R parameter 0
  ADDR2  := "VAR_ADDR".R_PARAM  // Structure of R parameters
  UNIT2  := B#16#1              // Index UNIT: Channel 1
  COLUMN2 :=
  LINE2  := W#16#2              // Index LINE: R parameter 1
  ADDR3  := "VAR_ADDR".R_PARAM  // Structure of R parameters
  UNIT3  := B#16#1              // Index UNIT: Channel 1
  COLUMN3 :=
  LINE3  := W#16#3              // Index LINE: R parameter 2
  ADDR4  :=
  UNIT4  :=
  COLUMN4 :=
  LINE4  :=
  ADDR5  :=
  UNIT5  :=
  COLUMN5 :=
  LINE5  :=
  ADDR6  :=
  UNIT6  :=
  COLUMN6 :=
  LINE6  :=
  ADDR7  :=
  UNIT7  :=
  COLUMN7 :=
  LINE7  :=
  ADDR8  :=
  UNIT8  :=
  COLUMN8 :=
  LINE8  :=
  FMDB_NO := 21                 // Assigned no. of user DB "FMx"
  ERROR   := "USERDB".EX5.ERR_WR // Request error
  DONE    := "USERDB".EX5.DONE   // Request finished message
  STATE   := #w_STATE           // FB error number
  SD1     := "USERDB".EX5.VAL_RPARAM_0 // Source area of R parameter 0
  SD2     := "USERDB".EX5.VAL_RPARAM_1 // Source area of R parameter 1
  SD3     := "USERDB".EX5.VAL_RPARAM_2 // Source area of R parameter 2
  SD4     :=
  SD5     :=
  SD6     :=
  SD7     :=
  SD8     :=

L   #w_STATE                     // On error, enter
T   "USERDB".EX5.STATE_WR        // error status ... in "USERDB"

```

6.3.6 FB 4: FM_PI – General services

Task

FB 4 can be used to select an NC program, acknowledge an error or activate machine data on the FM 357-2.

- **Select program, “SELECT”:**

A program stored on the FM 357-2 is selected for execution when you enter the program type (main program or subprogram) and the program name, as specified in the Programming Guide, Chapter 10.

- **Acknowledge error, “CANCEL”:**

Errors on the FM 357-2 are usually acknowledged on reset. If you want to continue selection on error acknowledgement, you can acknowledge certain errors on the FM 357-2 (see Chapter 12, Table 12-3 “Acknowledgement”) with the “CANCEL” service.

Note: “CANCEL” is transferred directly to the FM 357-2 with “Parameterize FM 357-2” or via OP.

- **Activate machine data, “CONFIG”**

After machine data have been modified (e.g. using FB 3), the change must be activated on the FM 357-2 with this service. You can only do this if the FM 357-2 has been reset or the program has been interrupted (activate stop at block boundary).

- **Activate FM restart, “FMRES”**

With the service “FMRES”, you can initiate a restart of the FM via the CPU. This will set the signal “FM restart” (user DB “FMx”, DBX10.1) and then the signal “FM start” (user DB “FMx”, DBX10.2). Only if the signals “FM start” = FALSE and “DSR ready” (user DB “FMx”, DBX10.0) = TRUE, you can continue working in the cyclic mode.

- **Activate user frames, “SETUFR”**

All required values of the frame must be transmitted to the FM by means of FB3. Then you can activate the user frames in the FM via the service “SETUFR”.

An FB 4 call with control input “REQ” starts a service referenced by “PISERVICE”.

Successful completion is indicated by status parameter “DONE” = TRUE.

The operation can take several CPU cycles before it is executed on the FM 357-2.

Any errors which occur are indicated by “ERROR” and “STATE”.

Data block DB 15, which is supplied, contains the internal structure of PI services.

A user DB is required as an instance DB or a multi-instance DB (for storing the parameters/program name).

Call options

The block should be called cyclically (OB 1). Before you start a new request, you must call FB 4 at least once with control input "REQ" = FALSE.

Call in LAD notation (ladder diagram)		Call in STL notation (statement list)
FB 4		CALL FB 4
— EN	ENO	REQ :=
— REQ	ERROR	PISERVICE :=
— PISERVICE	DONE	UNIT :=
— UNIT	STATE	ADDR1 :=
— ADDR1		ADDR2 :=
— ADDR2		FMDB_NO :=
— FMDB_NO		ERROR :=
		DONE :=
		STATE :=

Input parameters "ADDR1 and ADDR2" are only required for program selection.

Notice

A maximum of 3 FM services (FB 2, 3, 4 calls) can be started in each OB1 cycle.

FB 4 can only be activated if parameter "FMCOMM" has been set to TRUE (in OB 100: FC 1).

Input parameter "REQ" must not be reset until output parameter "DONE" or "ERROR" = TRUE (see Figure 6-6).

If communication between CPU and FM is aborted (FB 2, 3, 4) by POWER OFF or FM restart, the start requests must be deleted in the first OB 1 pass after startup (call FB 2/3/4 with signal REQ = FALSE).

Description of parameters

SELECT

The following table describes the parameters for program selection with “SELECT”.

Name	Data type	I/O type	Value range	Meaning
REQ	BOOL	I	–	Start request
PISERVICE	ANY	I	PI_DI.SELECT	PI service: Select program
UNIT	INT	I	1 to 4	Channel number
ADDR1	ANY	I		Path name ¹⁾ [STRING] (do not use local variables)
ADDR2	ANY	I		Program name ²⁾ [STRING] do not use local variables)
FMDB_NO	INT	I		No. of assigned user DB “FMx”
ERROR	BOOL	O	–	Request was given negative acknowledgement or could not be executed
DONE	BOOL	O	–	Request was successfully executed
STATE	WORD	O	–	See “Troubleshooting”

Parameter types: I = input parameter, O = output parameter,

1) Main program: ‘/_N_MPF_DIR/’; subroutine: ‘/_N_SPF_DIR/’ (see Section 6.8.4)

2) Main program: ‘_N_<Name>_MPF’; subprogram: ‘_N_<Name>_SPF’
<Name> = Name of program (see Section 6.8.4)

CANCEL

The following table describes the parameters for error acknowledgement “CANCEL”.

Name	Data type	I/O type	Value range	Meaning
REQ	BOOL	I	–	Start job request
PISERVICE	ANY	I	PI_DI.CANCEL	PI service: Acknowledge error
UNIT	INT	I	1 to 4	Channel number
FMDB_NO	INT	I		No. of assigned user DB “FMx”
ERROR	BOOL	O	–	Request was given negative acknowledgement or could not be executed
DONE	BOOL	O	–	Request was successfully executed
STATE	WORD	O	–	See “Troubleshooting”

Parameter types: I = input parameter, O = output parameter,

CONFIG

The following table describes the parameters for machine data activation “CONFIG”.

Name	Data type	I/O type	Value range	Meaning
REQ	BOOL	I	–	Start job request
PISERVICE	ANY	I	PI_DI.CONFIG	PI service: Activate machine data
UNIT	INT	I	1	channel-independent
FMDB_NO	INT	I		No. of the assigned user user DB “FMx”
ERROR	BOOL	O	–	Request was given negative acknowledgement or could not be executed
DONE	BOOL	O	–	Request was successfully executed
STATE	WORD	O	–	See “Troubleshooting”

Parameter types: I = input parameter, O = output parameter,

FMRES

The Table below describes the parameters for “Activate FM restart” “FMRES”.

Name	Data type	P type	Range of values	Meaning
REQ	BOOL	I	–	Start job request
PISERVICE	ANY	I	PI_DI.FMRES	PI service: Activate FM Restart
UNIT	INT	I	–	channel-independent
FMDB_NO	INT	I		No. of assigned user DB “FMx”
ERROR	BOOL	O	–	Request was given negative acknowledgement or could not be executed
DONE	BOOL	O	–	Request was successfully executed
STATE	WORD	O	–	See “Troubleshooting”

Parameter types: I = input parameter, O = output parameter

SETUFR

The Table below describes the parameters for activating the user frame "SETUFR".

Name	Data type	P type	Range of values	Meaning
REQ	BOOL	I	–	Start job request
PISERVICE	ANY	I	PI_DI.SETUFR	PI service: Activate user frames
UNIT	INT	I	1...4	Channel number
FMDB_NO	INT	I		No. of assigned user DB "FMx"
ERROR	BOOL	O	–	Job was given negative acknowledgement or could not be executed
DONE	BOOL	O	–	Job was successfully executed
STATE	WORD	O	–	See "Troubleshooting"

Parameter types: I = input parameter, O = output parameter

Troubleshooting

Status parameter "ERROR" is set to TRUE if it is not possible to execute a request. The cause of the error is stored at the "STATE" block output. The error code is deleted when the start signal is reset (FB 4 call with "REQ" = FALSE) after an error message.

Table 6-12 FB 4 troubleshooting

STATE	Significance	Note
3	Negative acknowledgement, request not executable	Internal error, remedy: <ul style="list-style-type: none"> • Reset and • Stop at block boundary
6	Serial request buffer is full	Request must be repeated, because the queue is full
7	Option not enabled	"FMCOMM" parameter (FC 1) is not set
13	Job terminated by FM restart	After restart of the FM, you must initialize the FB with REQ = FALSE. Then you can start a new job with REQ = TRUE.
14	Parameter "FMDB_NO" ist Null	Check the input parameter "FMDB_NO" and enter the correct user DB number.

Timing diagram

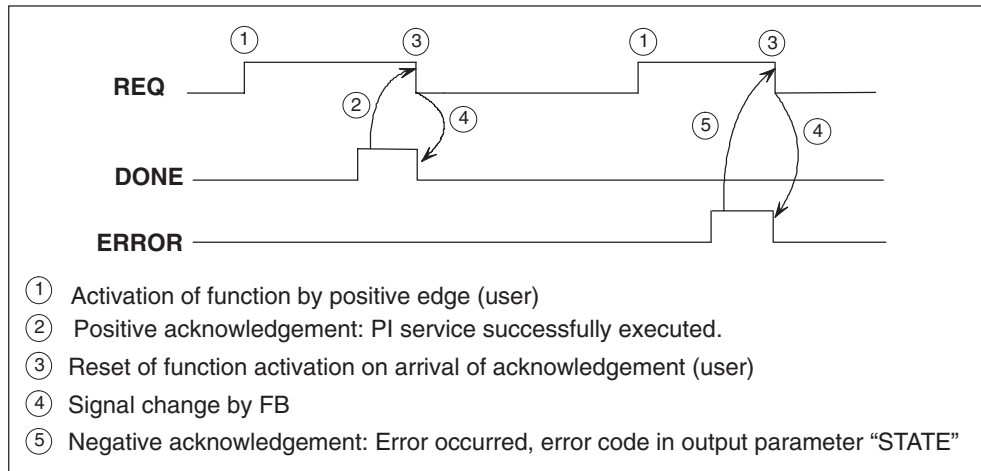


Fig. 6-6 Timing diagram for FB 4

For an example program selection call, see also example application 4 in Section 6.8.4

In the SIMATIC Manager, select **File > Open... > Project** and open the example project "zEn16_01_FM357-2_BF_EX\EXAMPLES\Quellen\EXAMPLE4".

Extract from symbol table:

Symbol	Address	Data type	Comments
FM_PI	FB 4	FB 4	General services
DI_FB4	DB 119	FB 4	Instance DB for FB 4
USERDB	DB 115	DB 115	User DB for examples

Extract from "USERDB" structure "EX4" with main program path and name entry:

Name	Type	Initial value	Comments
PATH	STRING[32]	'/_N_MPF_DIR/'	Main program path
P_NAME	STRING[32]	'_N_PROG_MPF'	Program name PROG
...
	END_STRUCT		

```

STL
U   "USERDB".EX4.START           // Start request
FP  "USERDB".EX4.HBIT           // Edge flag
S   "USERDB".EX4.FB4_REQ        // Start request FB 4

O   "USERDB".EX4.DONE           // Request terminated ?
O(
U   "USERDB".EX4.ACKN_ERR        // Error acknowledgement
U   "USERDB".EX4.ERROR          // FB 4 execution error
)
R   "USERDB".EX4.FB4_REQ        // Reset FB 4 request

CALL  "FM_PI" , "DI_FB4"        // Select program (FB 4)
REQ   := "USERDB".EX4.FB4_REQ    // Start function FB 4
PISERVICE := "PI_DI".SELECT     // DB 15 - function "SELECT"
UNIT   := 1                      // UNIT: Channel 1
ADDR1  := "USERDB".EX4.PATH      // Path
ADDR2  := "USERDB".EX4.P_NAME    // Program name
ADDR3  :=
ADDR4  :=
WVAR1  :=
WVAR2  :=
WVAR3  :=
WVAR4  :=
WVAR5  :=
WVAR6  :=
WVAR7  :=
WVAR8  :=
WVAR9  :=
WVAR10 :=
FMDB_NO := 21                    // Assigned no. of user DB "FMx"
ERROR  := "USERDB".EX4.ERROR     // Request error
DONE   := "USERDB".EX4.DONE      // Request finished message
STATE  := #w_STATE              // FB error number

L   B#16#0
U   "USERDB".EX4.ERROR          // Error ?
SPBN M001
L   #w_STATE                    // On error, enter
M001: T   "USERDB".EX4.STATE     // error status ... in "USERDB"

```

6.3.7 DB 121: VAR_ADDR – Extract from FM variable list

Task

Variable descriptions are required for the functions Read FM variable (FB 2) and Write FM variable (FB 3). The complete list of FM variable descriptions is available via the NC-VAR Selector tool.

DB 121 contains an extract of the complete list with some important FM variables for rapid use even without the NC-VAR Selector.

Some variables exist several times on the FM, e.g. axis-specific variables (as array with index “LINE”).

For these variables, the basic type has been selected from the complete list for DB 121 (“line = 0”, invalid index). Consequently, the line parameter (LINE) must be passed in the FB 2/FB 3 call. The input parameter “COLUMN 1...8” can be ignored in this example.

The appropriate channel number must be entered in the “UNIT” index. The number of the axis assigned to the channel to be read must be entered in the “LINE” index.

Table 6-13 Extract from variable list

FB 2/FB 3: ADDR1...8 Variable name in DB 121; VAR_ADDR. ...	Significance	S7 type	Access	Index FB 2/FB 3:	Variable name in NC-VAR Selector
				UNIT1...8 LINE1...8	Area Block Name
... SET_POS	Set position of axis (MCS)	REAL	FB 2	Channel number Axis number	C SMA cmdToolBasePos[.]
... ACT_FDFRATE	Actual axis velocity of axis in MCS (positioning axis)	REAL	FB 2	Channel number Axis number	C SEMA actFeedRate[.]
... CMD_FDFRATE	Axis velocity set-point of axis in MCS (positioning axis)	REAL	FB 2	Channel number Axis number	C SEMA cmdFeedRate[.]
... R_PARAM	R parameters	REAL	FB 2/ FB 3 1)	Channel number R-Nr. + 1	C RP rpa[.]
... FMERR_NO	FM error number (first error)	DINT	FB 2	– Number corresponds to sequence (1=first error)	N SALA textIndex[.]
... FMSTOP_COND	Hold or wait state of FM (e.g.: 7 = “hold, stop active”)	WORD	FB 2	Channel number –	C S stopCond

Table 6-13 Extract from variable list, continued

FB 2/FB 3: ADDR1...8 Variable name in DB 121; VAR_ADDR. ...	Significance	S7 type	Access	Index FB 2/FB 3:		Variable name in NC-VAR Selector	
				UNIT1...8 LINE1...8		Area Block Name	
... ACT_FDRATEPATH	Actual path velocity in the WCS	REAL	FB 2	Channel number —		C S	actFeedRateIpo
... CMD_FDRATEPATH	Path velocity set- point in the WCS	REAL	FB 2	Channel number —		C S	cmdFeedRateIpo
... PATH_OVERR	Path override in % (velocity override for path motion)	REAL	FB 2	Channel number —		C S	feedRateIpoOvr
... COUPACT_FOAX	System variables: Type of axis cou- pling of slave axis	WORD	FB 2	Channel number Axis number		C SEMA	aaCoupAct[.]
... SYNC_AX	System variables: Coupling status of slave axis	WORD	FB 2	Channel number Axis number		C SEMA	aaSync[.]
... SWCAM_M	Software cam table minus	REAL	FB 2 FB 3 2)	— Cam number		N SE	SW_CAM_MI- NUS_POS_TAB_1[.]
... SWCAM_P	Software cam table plus	REAL	FB 2 FB 3 2)	— Cam number		N SE	SW_CAM_PLUS_POS_ TAB_1[.]
... FIXED_STOPWIN	Fixed stop monitor- ing window	REAL	FB 2 FB 3 1)	— Axis number		A SE	FIXED_STOP_WINDOW
... MAX_AX_ACCEL	Axis acceleration	REAL	FB 2 FB 3 2)	— Axis number		A M	MAX_AX_ACCEL
... POSCTRL_GAIN	K _V factor	REAL	FB 2 FB 3 2)	— Axis number		A M	POSCTRL_GAIN[.]
... CONTOUR_TOL	Contour monitoring	REAL	FB 2 FB 3 2)	— Axis number		A M	CONTOUR_TOL
... POS_LIMIT_M	1. Software limit switch minus	REAL	FB 2 FB 3 2)	— Axis number		A M	POS_LIMIT_MINUS
... POS_LIMIT_P	1. Software limit switch plus	REAL	FB 2 FB 3 2)	— Axis number		A M	POS_LIMIT_PLUS

- 1) Effective immediately
- 2) Effective with execution of NewConf

For an example call, see also example application 5 in Section 6.8.5

6.4 Distributed installations

Information for writing a user program

If you want to operate the FM module via PROFIBUS DP (ET 200M module) in the distributed mode, you must embed the OB 86 in your application program. As a basis, the call sample OB 86 contained in the FM357_2L library can be used. The appropriate calls are to be found in the “BF_xxyyzz_distributed” directory in the OB 86 block under “Blocks” or under “Sources” in the “fm357ob_n1.awl” STL source.

To ensure a correct restart after a rack failure (DP bus fault), it is imperative that the FC 5 (BF_Diag) and FC 1 (RUN_UP) functions are embedded correctly. In case of a rack failure, FC 5 (BF Diag) of the appropriate FM must be called in OB 86, and you must ensure that the communication with the disturbed FM is prevented in OB 1.

If the distributed station logs in with the CPU again (rack recovery), you must call the block FC 1 (RUN_UP) for the relevant FM (analogously to the sequence in OB 100). This block ensures that the CPU program resynchronizes itself with the FM. In addition, OB 122 (I/O access error) must also be loaded into the CPU.

Notice

When configuring your hardware project, make sure that you have selected the FM 357-2 from the hardware catalog for distributed operation (supplementary information: “also for distributed use“, see Section 5.3 “Configuring”).

When parameterizing your hardware project, make sure that you parameterize the “PROFIBUS: DP master system” with a minimum of 1.5 Mbit/s.

You will find this setting at **HW Config; PROFIBUS: DP Master System > Object Properties > Subnetwork: PROFIBUS > Properties > Network Settings > Data Transfer Rate**

Transfer times of the interface between CPU ↔ FM

The transfer times of the interface and the cycle time are longer in distributed installations; see Section 6.9. The cycle time also increases in installations with several FMs. The cycle time is not doubled for each FM, however.

Several cycles may be necessary in order to transfer signals/data via jobs 1/2 and via FB 2, 3, 4.

6.5 User handling, function sequences

Overview

In this section, you can find information about:

- Axis control from the CPU, Section 6.5.1, page 6-49
- Auxiliary functions, Section 6.5.2, page 6-57
- ASUB – Starting asynchronous subprograms, Section 6.5.3, page 6-58

6.5.1 Axis control from the CPU

Axis control from the CPU is only allowed in “Jog” and “Automatic” modes.

A CPU axis can be traversed absolutely or relatively even without reference point.

Relevant signals/data for axis control from the CPU

Name	Signal/data User DB “AXy”	Signal type	Value range/ data type	Meaning
Request CPU axis	DBX50.7+m	ST	BOOL	Request axis for CPU
Start CPU axis	DBX51.7+m	ST	BOOL	Start CPU axis
Traverse shortest path	DBX51.1+m	ST	BOOL	Traverse shortest path, rotary axis only
Traversing path	DBX51.0+m	ST	BOOL	Traversing path: FALSE = absolute (G90) TRUE = incremental (G91)
Traversing dimension	DBX51.2+m	ST	BOOL	Traversing dimension: FALSE = mm TRUE = inch
Position of CPU axis	DBD52+m	ST	± 0.01 to $\pm 10^8$ / REAL	Position of the <ul style="list-style-type: none"> • Linear axis: mm, inch • Rotary axis: degrees
Feedrate of CPU axis	DBD56+m	ST	± 0.001 to $\pm 10^6$ / REAL	Feedrate of <ul style="list-style-type: none"> • Linear axis: mm/min, inch/min • Rotary axis: rpm If feed = 0.0, the configured feed of machine data “positioning velocity” is active
CPU axis	DBX66.7+m	RM	BOOL	Axis is CPU axis
Position reached	DBX67.6+m	RM	BOOL	The signal “Position reached” is TRUE if the axis has reached the fine target range (user DB “AXy, DBX20.7).
CPU axis active	DBX67.7+m	RM	BOOL	CPU axis active
CPU axis error	DBX67.1+m	RM	BOOL	Traversing error
Error number of CPU axis	DBB68+m	RM	BYTE	Error number of CPU axis

Signal type: ST = control signal, RM = checkback signal

Call, axis control from CPU with axis request from FC 22

For the axis to be traversed from the CPU, it may not already be activated by the FM 357-2, e.g. travel plus/minus checkback signals (user DB "AXy", DBX23.6/7+m) are not set.

The axis is controlled by FC 22 using the relevant control signals. In order position an axis from the CPU, the axis must be under the control of the CPU.

When the "Start CPU axis" signal (DBX51.7+m) is activated, FC 22 initially establishes whether the axis is assigned to the CPU. If not, it is requested by the FM (axis replacement by setting internal signal. The request is successful if checkback signal "CPU axis" (DBX66.7+m) is set. The positioning operation subsequently takes place.

When the positioning operation is complete, (signal "position reached" DBX67.6+m is set), "start CPU axis" signal (DBX51.7+m) must be reset by the user. FC 22 then terminates the positioning operation and returns the axis to the FM (axis replacement, signal "CPU axis", DBX66.7+m is reset). The axis cannot be restarted while this procedure is still running. The "position reached" signal remains set to TRUE during this period. As soon as the signal is FALSE, the axis can be restarted.

Timing diagram

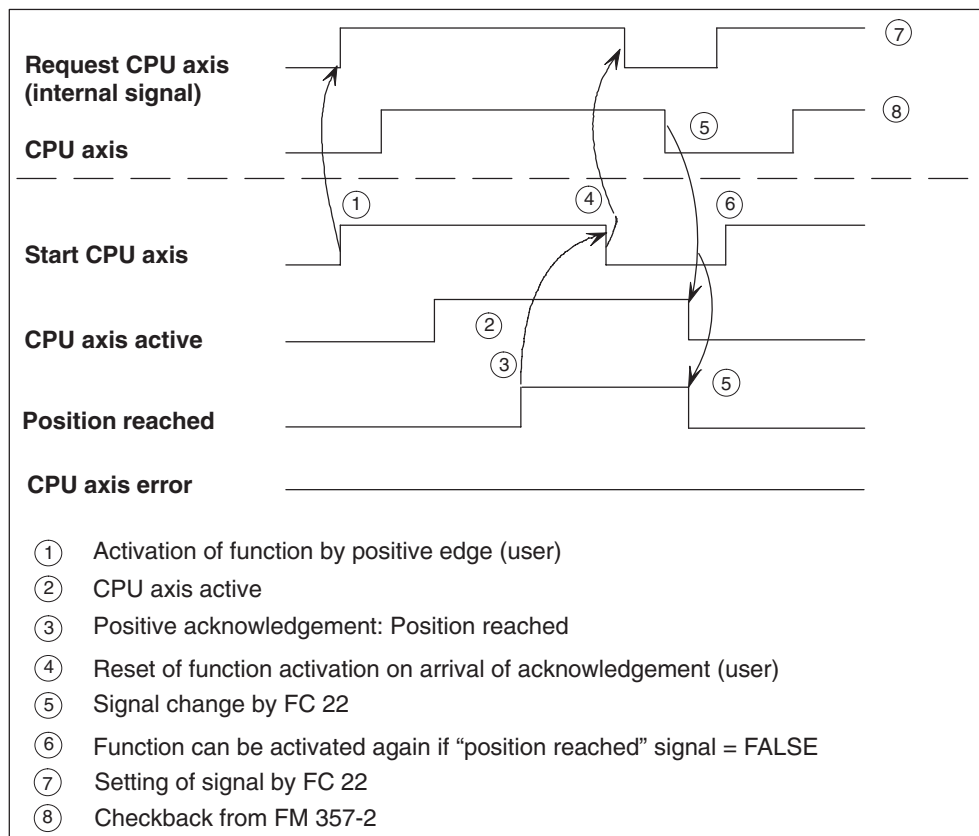


Fig. 6-7 Timing diagram, axis request by FC 22

Call, axis control from CPU with axis request from user

For the axis to be traversed from the CPU, it may not already be activated by the FM 357-2, e.g. travel plus/minus checkback signals (user DB "AXy", DBX23.6/7+m) are not set.

To achieve a high-speed sequence of positioning operations or to ensure exclusive use by the CPU, it is also possible to set a continuous CPU axis request from the FM (axis replacement) by setting signal "request CPU axis" (DBX50.7+m). This signal must be set by the user. A successful request is indicated by checkback signal "CPU axis" (DBX66.7+m).

As a consequence, the user cycles needed for an axis exchange are omitted between successive positioning operations.

The axis replacement is not performed if the axis is already assigned to the CPU when the "start CPU axis" signal (DBX51.7+m) is activated. The positioning operation takes place immediately.

When the positioning operation is complete, (signal "position reached" DBX67.6+m is set), "start CPU axis" signal (DBX51.7+m) must be reset by the user. FC 22 subsequently terminates the positioning operation. The axis cannot be restarted while this procedure is still running. The "position reached" signal remains set to TRUE during this period. As soon as the signal is FALSE, the axis can be restarted.

The axis is switched back to the neutral status when signal "request CPU axis" (DBX50.7+m) is reset by the user. The successful return is indicated by checkback signal "CPU axis" (DBX66.7+m).

For timing diagram, see Figure 6-8

Timing diagram

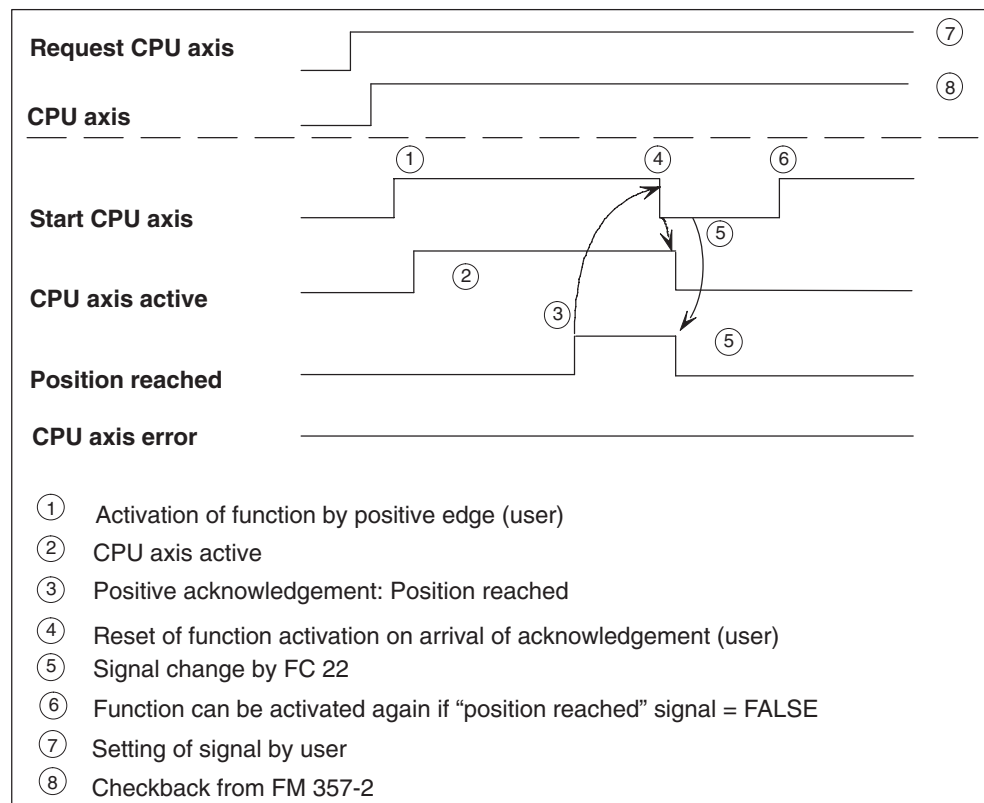


Fig. 6-8 Timing diagram, axis request by user

Interruption/cancelation of motion

The motion can be interrupted with:

- "Feed stop" (user DB "AXy", DBX4.3+m) = TRUE – can be resumed with "Feed stop" = FALSE
- "Stop" (user DB "FMx", DBX108.3+n) – can be resumed with "Start" (user DB "FMx", DBX108.1+n)

The motion can be canceled with:

- "Delete distance to go axial" (user DB "AXy", DBX2.2+m) and
- Reset "Start CPU axis" (DBX51.7+m) when "Position reached" (DBX67.6+m) or "CPU axis error" (DBX67.1+m) is set.

Troubleshooting

If signal “CPU axis error” = TRUE (DBX67.1+m), an error number is entered in DBB68+m.

The error number is deleted when signal “Start CPU axis” (DBX51.7+m) is reset after an error message.

Another “Start CPU axis” cannot be executed until signal “CPU axis error” has been set to FALSE.

Table 6-14 Troubleshooting, axis control from CPU

STATE (Hex)	Significance
02 (02)	Axis cannot be started
03 (03)	Servo enable of axis to be started missing
10 (0A)	Shortest path and incremental travel not possible at same time
30 (1E)	The programmed position was not reached
115 (73)	The programmed position was not reached
125 (7D)	Shortest path not possible
126 (7E)	Minus absolute value not possible
127 (7F)	Plus absolute value not possible
130 (82)	Software limit switch plus
131 (83)	Software limit switch minus
132 (84)	Working area limitation plus
133 (85)	Working area limitation minus

Note

In addition to the error evaluation (Table 6-14), the error messages must be evaluated as given in Chapter 11 (evaluation of channel errors).

Timing diagram (error condition)

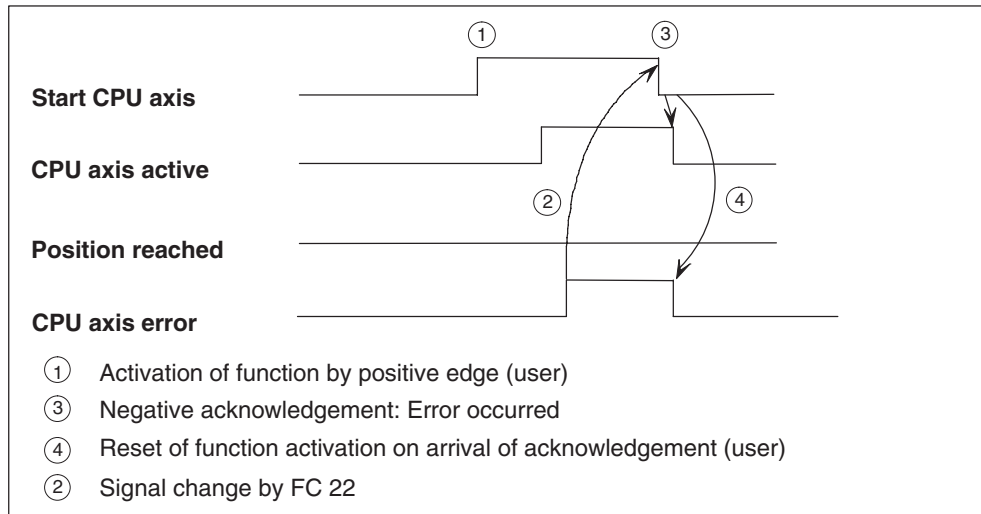


Fig. 6-9 Timing diagram (error condition), axis control from CPU

Independent CPU axis

With software version 5 and higher, a CPU axis or an axis started within a static synchronized action (Section 10.32) can be traversed as an independent CPU axis.

The known programming of the axis motion of a CPU axis or a traversing motion started within a static synchronized action will not change.

Properties of an independent CPU axis

On the one hand, an independent CPU axis traverses independently of the channels states Stop, Reset and alarm response, and, on the other hand, it has no influence on the NC program currently running.

The axis motion can be influenced directly from the CPU using the following interface signals:

- Stop (AX-DB "AXy", DBX65.6+m)
- Continue motion (AX-DB "AXy", DBX65.2+m)
- Reset (AX-DB "AXy", DBX65.1+m)

Preconditions for an independent CPU axis

The number of the independent CPU axes you want to configure must be set in the parameter "Independent CPU axes" (see Section 9.1).

To run an axis as an independent CPU axis, proceed as follows:

1. Set the interface signal "Request CPU axis" (user DB "AXy", DBX50.7+m) at least once after turning on the FM 357-2.
2. Set the interface signal "Request independent CPU axis" (user DB "AXy", DBX65.7+m).
3. Wait for the checkback signal "Independent CPU axis is requested" (user DB "AXy", DBX69.1+m).

When requested, the axis must not be traversed by the NC program. In this case, the checkback signal "Independent CPU axis is requested" (user DB "AXy", DBX69.1+m) not set, and the error 26072 "Channel... axis... cannot be traversed as an independent CPU axis".

Relevant signals/data for the independent CPU axis

Name	Signal/data of user DB "AXy"	Signal type	Range of values/ data type	Meaning
Request	DBX65.7+m	ST	BOOL	Request independent CPU axis
Stop	DBX65.6+m	ST	BOOL	Stop independent CPU axis
Continue motion	DBX65.2+m	ST	BOOL	Continue motion of an independent CPU axis after stop
Reset	DBX65.1+m	ST	BOOL	Abort motion of an independent CPU axis
Error	DBX69.2+m	RM	BOOL	Error traversing an independent CPU axis
is requested	DBD69.1+m	RM	BOOL	Axis is independent CPU axis
Reset acknowledgement	DBD69.0+m	RM	BOOL	Reset carried out for independent CPU axis

Signal type: ST = control signal, RM = checkback signal

Aborting the motion of a CPU axis

The axis motion of a CPU axis can be aborted by the following events:

- The selected CPU axis is being traversed. During the motion, the signal "Request independent CPU axis" is set to TRUE and then set back to FALSE.
- The selected CPU axis is being traversed. In addition, the signal "Request independent CPU axis" has been set. While the axis is moving, the signal "Independent CPU axis – Reset" is set.

Result:

The motion of the CPU axis is interrupted; the signal "Error CPU axis" is set (user DB "AXy", DBX67.1+m).

Error message, see Section 12.3.

6.5.2 Auxiliary functions

Notice

Please note that auxiliary functions which have been output (including M02 and M30) must be acknowledged with the “acknowledge auxiliary function” signal. Auxiliary functions must also be acknowledged on a “Reset” or program abort.

The availability of auxiliary functions in the channel is indicated by group signal “change auxiliary function” (user DB “FMx”, DBX127.0+n). The signals:

- “Change auxiliary function”,
- “M function (decoded)” and
- “Change M/T/H function”

are only set for one user cycle.

Signal “change auxiliary function” must be acknowledged by control signal “acknowledge auxiliary function” (user DB “FMx”, DBX109.0+n) so that the program can be resumed.

The auxiliary function numbers are retained until the next change.

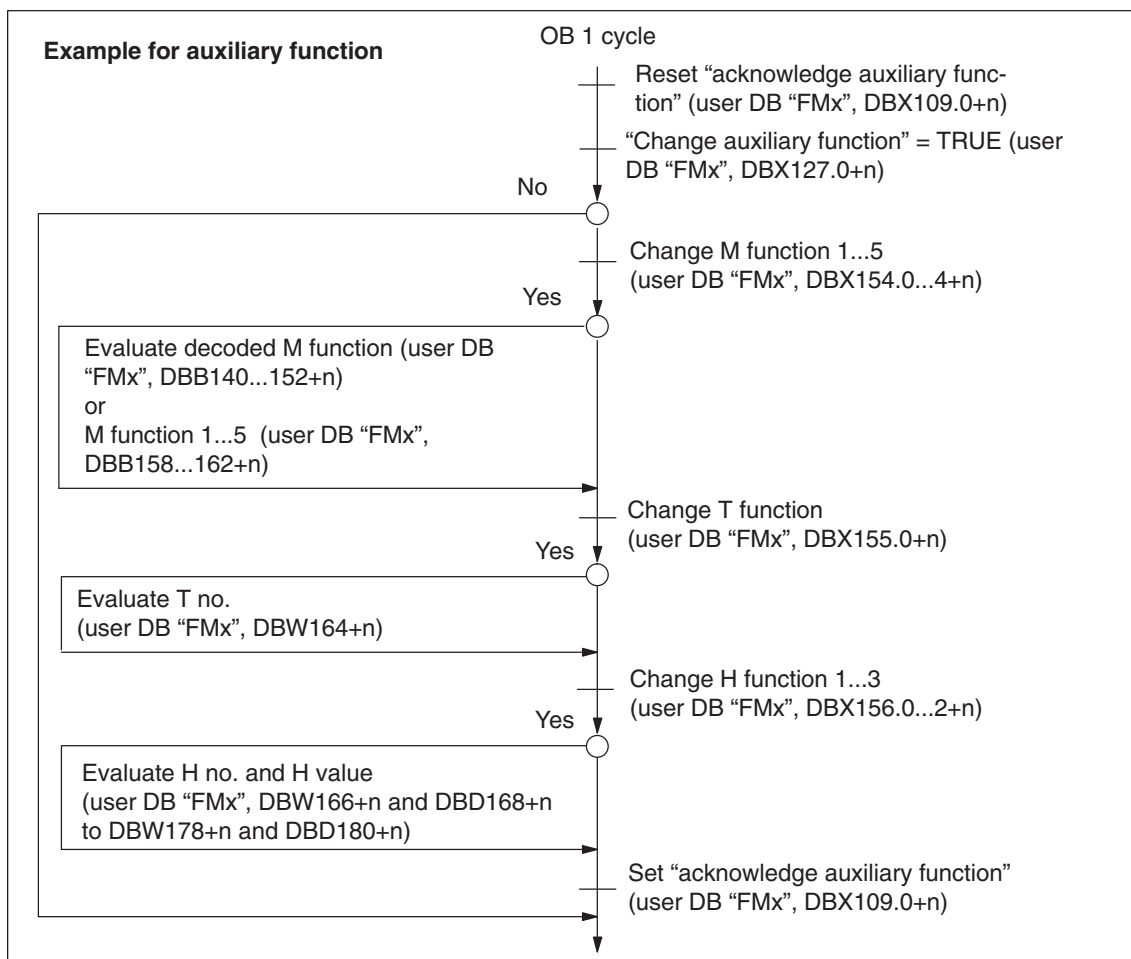


Fig. 6-10 Example of auxiliary function

6.5.3 ASUB – Start asynchronous subprograms

Relevant signals for calling an asynchronous subprogram

Name	Signal/data User DB "FMx"	Signal type	Meaning
Start ASUB	DBX110.7+n	ST	Start asynchronous subprogram
ASUB status: Active	DBX128.0+n	RM	Subprogram is active/running
ASUB status: Terminated	DBX128.1+n	RM	The asynchronous subprogram has terminated
ASUB status: Error	DBX128.2+n	RM	Error
ASUB status: Start error	DBX128.3+n	RM	Interrupt number 8 not allocated, see NC Programming, Chapter 10.30 and Functions, Chapter 9.14,

Signal type: ST = control signal, RM = checkback signal

“Start ASUB” can be used to start subprograms on the FM. The running NC program is interrupted by the asynchronous subprogram. An NC program must already have been created (see Section 10.30 and Section 9.14. “Interrupt 8” must be declared in the NC program for this function. A subprogram which has been prepared in this way can be started by the CPU at any time. Only one asynchronous subroutine can be started.

Timing diagram

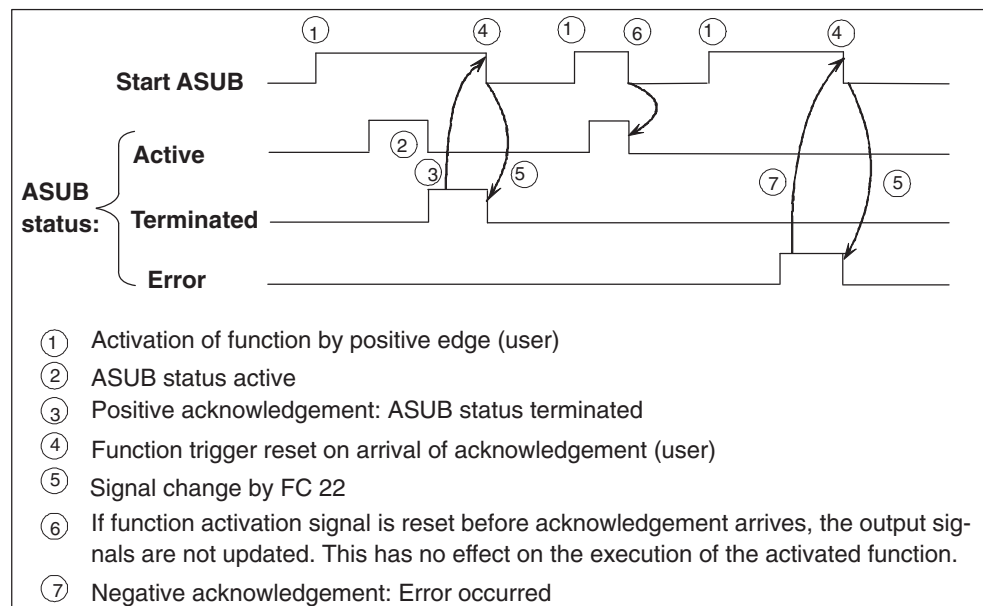


Fig. 6-11 Timing diagram for ASUB

Another “Start ASUB” cannot be executed until the “Start ASUB” signal has been reset after termination or an error (FC 22 must have been called at least once with the signal reset).

6.6 User data blocks (user DB) of the interface to the FM 357-2

Overview

In this section, you can find information about:

- User data block “FMx”, Section 6.6.1, page 6-61
- User data block “AXy”, Section 6.6.2, page 6-72
- Description of signals, Section 6.6.3, page 6-75

Overview of interface

The following table provides an overview of the control/checkback signal areas of user DB “FMx”.

Notice

Please note that you cannot write signals/data to the reserved areas or areas which are not depicted (reserved internally) of user DBs “FMx” and “AXy”.

Table 6-15 Control/checkback signals of user DB “FMx”

Absolute address	Relative address, in channel	Significance	
User DB “FMx”			
DBB0...19	DBB0	FM-specific	Status signals of FC 22 (general data, FM states)
DBB20...29	DBB20		Control signals
DBB30...99	DBB30		Checkback signals
DBB100	DBB00 + n	Channel 1 n = 100	Control signals, channel 1
	DBB20 + n		Checkback signals, channel 1
	DBB40 + n		Auxiliary functions, channel 1
DBB200	DBB00 + n	Channel 2 n = 200	Control signals, channel 2
	DBB20 + n		Checkback signals, channel 2
	DBB40 + n		Auxiliary functions, channel 2
DBB300	DBB00 + n	Channel 3 n = 300	Control signals, channel 3
	DBB20 + n		Checkback signals, channel 3
	DBB40 + n		Auxiliary functions, channel 3
DBB400	DBB00 + n	Channel 4 n = 400	Control signals, channel 4
	DBB20 + n		Checkback signals, channel 4
	DBB40 + n		Auxiliary functions, channel 4

Table 6-15 Control/checkback signals of user DB “FMx”, continued

Absolute address	Relative address, in channel	Significance	
User DB “FMx”			
DBB500	DBB500	Write request 1	Digital I/Os of FM
DBB540	DBB540	Read request 1	Digital I/Os of FM
DBB580	DBB580	Write request 2	Channel signals
DBB620	DBB620	Read request 2	Channel signals

The following table provides an overview of the control/checkback signal areas of user DB “AXy”.

Table 6-16 Control/checkback signals of user DB “AXy”

Absolute address	Relative address, in channel	Significance	
User DB “AXy”			
DBB0	DBB0 + m	Axis 1 m = 0	Control signals, axis 1
	DBB20 + m		Checkback signals, axis 1
	DBB50 + m		Control signals, axis 1, axis control from CPU
	DBB66 + m		Checkback signals, axis 1, axis control from CPU
DBB100	DBB0 + m	Axis 2 m = 100	Control signals, axis 2
	DBB20 + m		Checkback signals, axis 2
	DBB50 + m		Control signals, axis 2, axis control from CPU
	DBB66 + m		Checkback signals, axis 2, axis control from CPU
DBB200	DBB0 + m	Axis 3 m = 200	Control signals, axis 3
	DBB20 + m		Checkback signals, axis 3
	DBB50 + m		Control signals, axis 3, axis control from CPU
	DBB66 + m		Checkback signals, axis 3, axis control from CPU
DBB300	DBB0 + m	Axis 4 m = 300	Control signals, axis 4
	DBB20 + m		Checkback signals, axis 4
	DBB50 + m		Control signals, axis 4, axis control from CPU
	DBB66 + m		Checkback signals, axis 4, axis control from CPU

6.6.1 User data block “FMx”

The following table describes the structure of user DB “FMx”.

Table 6-17 User DB “FMx”

User DB “FMx”								
FM 357-2 and channel signals								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
General data, FM 357 states (status signals of FC 22)								
DBW0	Module address							
DBB2	Number of channels							
DBB3	No. of axes							
DBW4	Error in basic function							
DBB6							Write request 2: Channel signals	Write request 1: General FM data
DBB7							Error, write request 2	Error, write request 1
DBB8							Read request 2: Channel signals	Read request 1: General FM data
DBB9							Error, read request 2	Error, read request 1
DBB10						FM startup	FM restart	Communication ready
DBB11							User: Test enable	Startup window: test request
DBB12 to DBB19	Reserved							
Control signals of the FM 357-2 (general FM signals)								
DBB20							Acknowledge EM-ERG. STOP	EMERG. OFF
DBB21							Request distance to go	Request actual value

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”		FM 357-2 and channel signals							
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
DBW22	CPU variable (write)								
DBB24 to DBB29	Reserved								
Checkback signals of the FM 357-2 (general FM signals)									
DBB30	FM ready						Probe activated Probe 2 Probe 1		
DBB31	FM battery error active						EMERG. STOP active	FM error active	
DBB32	Software cam minus								
	8	7	6	5	4	3	2	1	
DBB33	Software cam plus								
	8	7	6	5	4	3	2	1	
DBW34	CPU variable (read)								
DBB36 to DBB99	Reserved								
Channel control signals [channel 1 starts at byte 100; each further channel (n) plus offset 100]									
DBB100 + n						Jog	Mode MDI	Automatic	
DBB101 + n						Reference-Point Approach	Submode REPOS ¹⁾	TEACH IN ¹⁾	
DBB102 + n		Continuous traversing	INC variable ¹⁾	Incremental dimension					
				10 000 INC	1 000 INC	100 INC	10 INC	1 INC	
DBB103 + n		Activate dry run feed ¹⁾	Activate M01	Activate single block					
DBB104 + n	Activate program test ¹⁾								
DBB105 + n								Activate skip block	
DBB106 + n	Path override								

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”									
FM 357-2 and channel signals									
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
DBB107 + n	Path override Feed over-ride active		Rapid tra-verse over-ride active				Delete the distance to go	Read-in disable	Activate feed dis-able
DBB108 + n	Reset				Stop		Activate stop at block boundary	Start	Activate start dis-able
DBB109 + n									Acknowl-edge aux. function
DBB110 + n	Start ASUB								
DBB111 to DBB113	Reserved								
DBB114	Geometry axis 1 ¹⁾								
	Direction plus	Direction minus	Rapid tra-verse over-ride	Travel dis-able	Feed STOP				
DBB115	Geometry axis 2 ¹⁾								
	Direction plus	Direction minus	Rapid tra-verse over-ride	Travel dis-able	Feed STOP				
DBB116	Geometry axis 3 ¹⁾								
	Direction plus	Direction minus	Rapid tra-verse over-ride	Travel dis-able	Feed STOP				
DBB117	Orientation axis 1 ¹⁾								
	Direction plus	Direction minus	Rapid tra-verse over-ride	Travel dis-able	Feed STOP				
DBB118/DBB119	Reserved								
Channel checkback signals [channel 1 starts at byte 120; each further channel (n) plus offset 100]									
DBB120 + n						Active mode			
						Jog	MDI	Automatic	
DBB121 + n						Active submode			
						Reference-Point Ap-proach	REPOS ¹⁾	TEACH IN ¹⁾	

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”									
FM 357-2 and channel signals									
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
DBB122 + n		Continuous traversing active	INC variable 1)	Active increment					
				10 000 INC	1 000 INC	100 INC	10 INC	1 INC	
DBB123 + n			M00/M01 active						
DBB124 + n	Program test active ¹⁾	Transformation active ¹⁾	M02/M30 active						
DBB125 + n	Channel reset			Program					
				aborted	interrupted	stopped	waiting	running	
DBB126 + n	C. error, program stopped	Channel error	Channel ready		All axes stationary	All axes referenced			
DBB127 + n								Change auxiliary function	
DBB128 + n					ASUB status				
					Start error	Error	Terminated	Active	
DBB129 to DBB133	Reserved								
DBB134	Geometry axis 1 ¹⁾								
	Go_plus	Travel minus							
DBB135	Geometry axis 2 ¹⁾								
	Go_plus	Travel minus							
DBB136	Geometry axis 3 ¹⁾								
	Go_plus	Travel minus							
DBB137	Orientation axis 1 ¹⁾								
	Go_plus	Travel minus							
DBB138/DBB139	Reserved								
Channel auxiliary functions [channel 1 starts at byte 140; each further channel (n) plus offset 100]									
DBB140 + n	M functions (decoded)								
	M7	M6	M5	M4	M3	M2	M1	M0	
DBB141 + n	M functions (decoded)								
	M15	M14	M13	M12	M11	M10	M9	M8	

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”		FM 357-2 and channel signals						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB142 + n	M functions (decoded)							
	M23	M22	M21	M20	M19	M18	M17	M16
DBB143 + n	M functions (decoded)							
	M31	M30	M29	M28	M27	M26	M25	M24
DBB144 + n	M functions (decoded)							
	M39	M38	M37	M36	M35	M34	M33	M32
DBB145 + n	M functions (decoded)							
	M47	M46	M45	M44	M43	M42	M41	M40
DBB146 + n	M functions (decoded)							
	M55	M54	M53	M52	M51	M50	M49	M48
DBB147 + n	M functions (decoded)							
	M63	M62	M61	M60	M59	M58	M57	M56
DBB148 + n	M functions (decoded)							
	M71	M70	M69	M68	M67	M66	M65	M64
DBB149 + n	M functions (decoded)							
	M79	M78	M77	M76	M75	M74	M73	M72
DBB150 + n	M functions (decoded)							
	M87	M86	M85	M84	M83	M82	M81	M80
DBB151 + n	M functions (decoded)							
	M95	M94	M93	M92	M91	M90	M89	M88
DBB152 + n	M functions (decoded)							
					M99	M98	M97	M96
DBB153	Reserved							
DBB154 + n	Change M functions							
				M function 5	M function 4	M function 3	M function 2	M function 1
DBB155 + n	Change T functions							
								T function 1
DBB156 + n	Change H functions							
						H function 3	H function 2	H function 1
DBB157	Reserved							
DBB158 + n	M function number 1							

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”		FM 357-2 and channel signals						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB159 + n	M function number 2							
DBB160 + n	M function number 3							
DBB161 + n	M function number 4							
DBB162 + n	M function number 5							
DBB163	Reserved							
DBW164 + n	T function number							
DBW166 + n	H function number 1							
DBD168 + n	H function value 1 (data type REAL)							
DBW172 + n	H function number 2							
DBD174 + n	H function value 2 (data type REAL)							
DBW178 + n	H function number 3							
DBD180 + n	H function value 3 (data type REAL)							
DBB184 to DBB200	Reserved							
Write request 1 (digital I/Os of FM); length: 24 bytes								
DBB500	Reserved							
DBB501	Set digital on-board inputs from CPU							
	Input 8	Input 7	Input 6	Input 5	Input 4	Input 3	Input 2	Input 1

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”								
FM 357-2 and channel signals								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB502	Reserved							
DBB503	Set digital on-board inputs from CPU							
					Input 12	Input 11	Input 10	Input 9
DBB504	Disable digital on-board outputs							
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
DBB505	Reserved							
DBB506	Set digital on-board outputs from CPU							
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
DBB507	Define digital on-board outputs from CPU ↔ FM							
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
DBB508	Reserved							
DBB509	Set digital inputs on local P bus							
	Input 24	Input 23	Input 22	Input 21	Input 20	Input 19	Input 18	Input 17
DBB510	Reserved							
DBB511	Set digital inputs on local P bus							
	Input 32	Input 31	Input 30	Input 29	Input 28	Input 27	Input 26	Input 25
DBB512 to DBB515	Reserved							
DBB516	Disable digital outputs on local P bus							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
DBB517	Reserved							
DBB518	Set digital outputs on local P bus							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
DBB519	Define digital outputs on local P bus							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
DBB520	Disable digital outputs on local P bus							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”		FM 357-2 and channel signals						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB521	Reserved							
DBB522	Set digital outputs on local P bus							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
DBB523	Define digital outputs on local P bus							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
DBB524 to DBB539	Reserved							
Read request 1 (digital I/Os of FM); length: 12 bytes								
DBB540	Status of digital on-board inputs of FM							
	Input 8	Input 7	Input 6	Input 5	Input 4	Input 3	Input 2	Input 1
DBB541	Status of digital on-board inputs of FM							
					Input 12	Input 11	Input 10	Input 9
DBB542	Status of digital on-board outputs of FM							
	Output 8	Output 7	Output 6	Output 5	Output 4	Output 3	Output 2	Output 1
DBB543	Reserved							
DBB544	Status of digital inputs on local P bus							
	Input 24	Input 23	Input 22	Input 21	Input 20	Input 19	Input 18	Input 17
DBB545	Status of digital inputs on local P bus							
	Input 32	Input 31	Input 30	Input 29	Input 28	Input 27	Input 26	Input 25
DBB546/ DBB547	Reserved							
DBB548	Status of digital outputs on local P bus							
	Output 16	Output 15	Output 14	Output 13	Output 12	Output 11	Output 10	Output 9
DBB549	Status of digital outputs on local P bus							
	Output 24	Output 23	Output 22	Output 21	Output 20	Output 19	Output 18	Output 17
DBB550 to DBB579	Reserved							

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”								
FM 357-2 and channel signals								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Write request 2 (channel signals); length: 24 bytes								
Channel 1								
DBB580	Disable synchronized action							
	ID7	ID6	ID5	ID5	ID4	ID3	ID2	ID1
DBB581	Reserved							
DBB582	Activate protection zone							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB583	Reserved							
DBB584				Activate point-to-pt. traversing ¹⁾				
DBB585	Reserved							
Channel 2								
DBB586	Disable synchronized action							
	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1
DBB587	Reserved							
DBB588	Activate protection zone							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB589 to DBB591	Reserved							
Channel 3								
DBB592	Disable synchronized action							
	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1
DBB593	Reserved							
DBB594	Activate protection zone							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB595 to DBB597	Reserved							

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”								
FM 357-2 and channel signals								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Channel 4								
DBB598	Disable synchronized action							
	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1
DBB599	Reserved							
DBB600	Activate protection zone							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB601 to DBB619	Reserved							
Read request 2 (channel signals); length: 32 bytes								
Channel 1								
DBB620	Synchronized action disable possible from CPU							
	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1
DBB621	Reserved							
DBB622	Protection zone preactivated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB623	Reserved							
DBB624	Protection zone violated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB625	Reserved							
DBB626		Point-to-pt. traversing active ¹⁾						
DBB626		Dry run feed active ¹⁾						
Channel 2								
DBB628	Synchronized action disable possible from CPU							
	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1
DBB629	Reserved							

1) with FM 357-2 handling only

Table 6-17 User DB “FMx”, continued

User DB “FMx”		FM 357-2 and channel signals						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB630	Protection zone preactivated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB631	Reserved							
DBB632	Protection zone violated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB633 to DBB635	Reserved							
	Channel 3							
DBB636	Synchronized action disable possible from CPU							
	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1
DBB637	Reserved							
DBB638	Protection zone preactivated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB639	Reserved							
DBB640	Protection zone violated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB641 to DBB643	Reserved							
	Channel 4							
DBB644	Synchronized action disable possible from CPU							
	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1
DBB645	Reserved							
DBB646	Protection zone preactivated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB647	Reserved							
DBB648	Protection zone violated							
					Zone 4	Zone 3	Zone 2	Zone 1
DBB649 to end	Reserved							

1) with FM 357-2 handling only

6.6.2 User data block “AXy”

The table below shows you the structure of user DB “AXy”.

Table 6-18 User DB “AXy”

User DB “AXy”		Axis signals						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Axis control signals [axis 1 starts at byte 0; each further axis (m) plus offset 100]								
DBB0 + m	Override							
DBB1 + m	Activate override			Follow-up mode		Fixed stop sensor	Acknowl- edge fixed stop reached	
DBB2 + m					Activate terminals	Axial dele- tion of dis- tance to go	Enable CL controller	Activate software cam
DBB3 + m							Enable travel to fixed stop	
DBB4 + m	Direction Plus Minus		Activate ra- pid traverse override		Feed STOP			
DBB5/ DBB6	Reserved							
DBB7 + m	Delay Ref- erence- Point Ap- proach				2. Software limit switches Plus Minus		Hardware limit switch Plus Minus	
DBB8 + m				Enable following axis override			Axis con- trol	Rotation monitoring stepper motor
DBB9 + m		Oscillation, Stop	O. , stop at next rever- sal point					
DBB10 + m				Start gantry syn- chroniza- tion run				
DBB11	Reserved							
DBB12 + m					Speed set- point smoothing 1)		Ramp-func- tion genera- tor rapid stop 1)	
DBB13 + m	Pulse en- able	Integrator disable, speed con- troller 1)				Select parameter set 1)		

1) only for SIMODRIVE 611-U drives via PROFIBUS DP

2) with FM 357-2 handling only

Table 6-18 User DB “AXy”, continued

User DB “AXy”		Axis signals						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB13 + m	Pulse enable	Integrator disable, speed controller ¹⁾				C	B	O
DBB14 to DBB19	Reserved							
Axis checkback signals [axis 1 starts at byte 20; each further axis (m) plus offset 100]								
DBB20 + m	Position reached, stop			synchronized, referenced				
	Fine target range	Coarse target range						
DBB21 + m		Speed controller active	Position controller active	Axis stationary	Follow-up mode active			
DBB22 + m			Fixed stop reached	Travel to fixed stop active	Measurement active			Software cam active
DBB23 + m	Travel			Synchronization running	Axis accelerating		Following axis active	Master axis active
	Plus	Minus						
DBB24 + m		Continuous traversing active	INC variable ²⁾	Active increment				
				10 000 INC	1 000 INC	100 INC	10 INC	1 INC
DBB25 + m		Neutral axis	Axis change possible		Axis in channel			
					D	C	B	A
DBB26 to DBB27	reserved							
DBB28 + m		Warning threshold		Superimposed motion			Synchronism	
		Acceleration reached	Velocity reached				coarse	fine
DBB29 + m	Axis is oscillation axis	Oscillation motion active					Axis control active	F. rotation monitoring stepper motor
DBB30 + m	Gantry axis	Gantry master axis	Gantry grouping is synchronized	Gantry synchronization run ready	G. limit for warning exceeded	G. shut-down limit exceeded		
DBB31 + m					Speed set-point smoothing ¹⁾	Torque limit ^{2 1)}	Ramp-function generator disable ¹⁾	Setup mode ¹⁾
DBB32 + m	Pulse enable	Integrator disable, speed controller ¹⁾	Drive ready ¹⁾			Active parameter set ¹⁾		
						C	B	O

1) only for SIMODRIVE 611-U drives via PROFIBUS DP

2) with FM 357-2 handling only

Table 6-18 User DB "AXy", continued

User DB "AXy" Axis signals								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB33 + m	Variable message function ¹⁾	$n_{act}=n_{set}$ ¹⁾	$ n_{act} < n_x$ ¹⁾	$ n_{act} < n_{min}$ ¹⁾	$ M_d < M_{dx}$ ¹⁾	Startup operation finished ¹⁾	Temperature prewarning ¹⁾ Heatsink Motor	
DBB34 to DBB39	Reserved							
DBD40 + m	Actual value (data type REAL)							
DBD44 + m	Distance to go (data type REAL)							
DBW48	Reserved							
Control signals, axis control from the CPU [axis 1 starts at byte 50; each further axis (m) plus offset 100]								
DBB50 + m	Request CPU axis							
DBB51 + m	Start CPU axis					Traversing dimension inches	Traverse shortest path	Traversing path incremental
DBD52 + m	Position of CPU axis (data type REAL)							
DBD56 + m	Feedrate of CPU axis (data type REAL)							
Independent CPU axis								
DBB65 + m	Request	Stop				Continue motion	Reset	
Checkback signals, axis control from the CPU [axis 1 starts at byte 66; each further axis (m) plus offset 100]								
DBB66 + m	CPU axis							
DBB67 + m	CPU axis active	Position reached					CPU axis error	
DBB68 + m	Error number of CPU axis							
DBB69 + m						Independent CPU axis Error is requested Reset ackn.		
DBB70 to DBB99	reserved							

1) only for SIMODRIVE 611-U drives via PROFIBUS DP
 2) with FM 357-2 handling only

6.6.3 Description of signals

Control and checkback signals

The axis is operated and controlled by means of control signals.

All signals (including edge signals) must be active for at least one interpolator cycle in order to be detected on the FM 357-2.

The checkback signals indicate the signal state of the axis and report it to the user DBs.

Table 6-19 describes the control/checkback signals of user data block "FMx" and their function.

Table 6-19 Signals for user DB "FMx"

Address symbol	Name Function
General signals, FM 357-2 states	
DBW0 MOD_ADDR	Module address (checkback signal) acc. to input parameter of FC 1, is required by FC 22
DBB2 CH_NO	Number of channels (checkback signal) Number of configured channels, is entered in cyclic startup (FC 22)
DBB3 AX_NO	Number of axes (checkback signal) Number of configured axes, is entered in cyclic startup (FC 22)
DBW4 BF_ERR	Basic function error (checkback signal) Error code acc. to FC 5, FC 22 (FM error, communication error)
DBX6.0/1 WRJOB1/2	Write request 1 or 2 (control signal) <ul style="list-style-type: none"> • Activation of a data transfer to FM (user) Data areas user DB "FMx", DBB500...523 or user DB "FMx", DBB580...600 • The request is cancelled after successful transmission (FC 22). • Where several requests are set simultaneously, request 1 is processed first
DBX7.0/1 ERR_ WRJOB1/2	Error on write request 1 or 2 (checkback signal) TRUE = Data transfer error (the communication ready signal is reset simultaneously) FALSE = On new request This signal is set/cleared by FC 22.
DBX8.0/1 RDJOB1/2	Read request 1 or 2 (control signal) <ul style="list-style-type: none"> • Activation of a data transfer from FM (user) Data areas user DB "FMx", DBB540...549 or user DB "FMx", DBB620...648 • The request is cancelled after successful transmission (FC 22). • Where several requests are set simultaneously, request 1 is processed first

1) with FM 357-2 handling only

Table 6-19 Signals for user DB “FMx”, continued

Address symbol	Name Function
DBX9.0/1 ERR_ RDJOB1/2	Error on read request 1 or 2 (checkback signal) TRUE = Data transfer error (the communication ready signal is reset simultaneously) FALSE = On new request This signal is set/cleared by FC 22.
DBX10.0 CONNECT	Communication ready (checkback signal) Communication link between CPU and FM is ready If the “Communication ready” status signal is reset, the communication link between the FM and CPU is not ready (during power-up) or a communication error has occurred. If the “communication ready” status signal is reset during operation, an error has occurred and your plant must be shut down (incorporate the signal into the EMERGENCY STOP circuit). This signal is set/cleared by FC 22 and FC 5.
DBX10.1 RESTART	FM restart (checkback signal) An FM restart is initiated manually. The FM is reset and a new startup is performed. see Section 6.3.2, FC 5 This signal is set/cleared by FC 22.
DBX10.2 STARTUP	FM startup (checkback signal) The cyclic startup (OB 1) with the FM is not yet complete. As long as “FM startup” has not been canceled, no user program may be started for the FM. The user memory must be initialized. This signal is set/cleared by FC 22.
DBX11.0 PRES_TEST	Startup window: Test request (checkback signal) Start test mode with “Parameterize FM357-2”. The signal is set and cancelled in the “Startup window”.
DBX11.1 ACT_TEST	User: enable test (control signal) Enable test mode from user program (signal is only required for test mode with parameterization tool)
Control signals of the FM 357-2 (general FM signals)	
DBX20.1 EM_STOP	EMERGENCY STOP This signal must be transferred to the FM when the EMERGENCY STOP button is actuated. The FM initiates the following reactive measures: <ul style="list-style-type: none"> • NC program execution is aborted. • All axes are decelerated within the period set in parameter “Braking time EMERGENCY STOP”. • Interface signal “channel ready” (user DB “FMx”, DBX126.5+n) is reset. • Interface signal “EMERGENCY STOP active” is set (user DB “FMx”, DBX31.1). • Error 3 000 “EMERGENCY STOP” is output. • The controller enable signal to the drive is cancelled after the period set in parameter “Cutout delay controller enable EMERGENCY STOP”. • All axes are switched internally to follow-up.

1) with FM 357-2 handling only

Table 6-19 Signals for user DB “FMx”, continued

Address symbol	Name Function
<p>DBX20.2 ACKN_EM_STOP</p>	<p>EMERGENCY STOP acknowledgement</p> <p>The EMERGENCY STOP status is reset by the sequence of operations shown in the following diagram.</p> <p>Interface signals user DB “FMx”</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;"> <p>Acknowledgement effective</p> </div> <div style="text-align: center;"> <p>Acknowledgement ineffective</p> </div> </div> <p>① Interface signals “EMERGENCY STOP acknowledgement” and “Reset” reset “EMERGENCY STOP active”.</p> <p>② Interface signal “EMERGENCY STOP acknowledgement” has no effect.</p> <p>③ Interface signal “Reset” has no effect</p> <p>Sequence in FM after reset of EMERGENCY STOP status:</p> <ul style="list-style-type: none"> • The controller enable (user DB “AXy”, DBX2.1+m) to the drive is activated. • Follow-up mode (user DB “AXy”, DBX1.4+m) is canceled. • Interface signal “Position controller active” (user DB “AXy”, DBX21.5+m) is set. • Interface signal “Channel ready” (user DB “FMx”, DBX126.5+n) is set. • Interface signal “EMERGENCY STOP active” (user DB “FMx”, DBX31.1) is reset. • Error 3 000 “EMERGENCY STOP” is deleted.
<p>DBX21.1 ACT_POS</p>	<p>Request actual value</p> <p>The actual value of all axes is read cyclically (on each FC 22 call).</p> <p>(In order to minimize the communication runtime in the FC 22/OB 1 cycle, particularly in distributed configurations, the “actual value” should only be read cyclically if it is needed for the user function.)</p> <p>The actual value can be read out from user DB “AXy”, DBD40+m</p>
<p>DBX21.2 DIS_TOGO</p>	<p>Request distance to go</p> <p>The actual distance to go of all axes is read cyclically (on each FC 22 call).</p> <p>(In order to minimize the communication runtime in the FC 22/OB 1 cycle, particularly in distributed configurations, the “distance to go” should only be read cyclically if it is needed for the user function.)</p> <p>The actual distance to go can be read out from user DB “AXy”, DBD44+m</p>
<p>DBW22 CPU_VAR</p>	<p>CPU variable (write)</p> <p>Data word for optional use in synchronized actions (see Section 10.32). Readable in the FM 357-2 via system variable \$A_DBW[0].</p>

1) with FM 357-2 handling only

Table 6-19 Signals for user DB “FMx”, continued

Address symbol	Name Function
Checkback signals of the FM 357-2 (general FM signals)	
DBX30.0/1 PROBE1/2	Probe 1/2 activated With the “measure” function, these signals indicate whether probe 1 or 2 is active.
DBX30.7 FM_RDY	FM ready The controller is ready for operation. This signal represents the relay contact “FM-READY”. The signal is activated if <ul style="list-style-type: none"> • the “FM-READY” relay contact is closed • all internal controller power circuits have been set up • the controller is in cyclical mode
DBX31.0 FM_ERR	FM error active Group error of channels 1 to 4 (see Section 12.2 or, for handling, Section 13.7).
DBX31.1 EM_STOP	EMERGENCY STOP active FM 357-2 signal in EMERGENCY STOP status (see signals in user DB “FMx”, DBX20.1 and DBX20.2)
DBX31.7 BATT_ERR	FM battery error active The battery has no contact or must be replaced. The FM can continue to be operated. A power failure or disconnection from the mains will result in the loss of data from the volatile memory.
DBB32 SWCAM_M0 to SWCAM_M7	Software cams minus Displays the status of software cams 0 to 7 minus.
DBB33 SWCAM_P0 to SWCAM_P7	Software cams plus Displays the status of software cams 0 to 7 plus.
DBW34 CPU_VAR	CPU variable (read) Data word for optional use of synchronized actions (see Section 10.32). Can be written in the FM 357-2 via system variable \$A_DBW[2].
Channel control signals [starts at channel 1; each further channel (n) plus offset 100]	
DBX100.0+n MODE_A	“Automatic” mode This signals selects “Automatic” mode for running NC programs. see Section 9.12.
DBX100.1+n MODE_MDI	“MDI” mode Possible only through an operator input in startup with “Parameterize FM 357-2” tool or with “FM 357-2 Handling”. This signals selects “MDI” mode for running blocks entered manually. See Section 9.12 or Section 13

1) with FM 357-2 handling only

Table 6-19 Signals for user DB "FMx", continued

Address symbol	Name Function
DBX100.2+n MODE_JOG	<p>"Jog" mode</p> <p>This signal selects "Jog" mode (see Section 9.12). Traversing movement with control signals "Direction plus", "Direction minus" (user DB "AXy", DBX4.6/7).</p> <p>Note: After deselecting "Incremental dimensions", "Continuous travel" must be set for "Jog" mode.</p>
DBX101.0+n TEACH_IN	<p>"TEACH IN" submode of "MDI" ¹⁾</p> <p>"TEACH IN" is selected in addition to "MDI" mode. e.g. for entering axis positions (program development using current positions) see Section 13</p>
DBX101.1+n REPOS	<p>"REPOS" (repositioning) submode of "Jog" or "MDI" ¹⁾</p> <p>"REPOS" is selected in addition to "Jog" or "MDI" mode. e.g. repositioning at contour see Section 13.</p>
DBX101.2+n REF_POINT	<p>"Reference point approach" submode of "Jog"</p> <p>"Reference point approach" is selected in addition to "Jog" mode. Reference point approach (axis synchronization) see Section 9.12.</p>
DBX102.0+n to DBX102.4+n INC1...10000	<p>Incremental dimension (submode of "Jog")</p> <p>If an incremental dimension is selected in "Jog" mode, the resulting submode is "Incremental relative" (see Section 9.12). Selects the incremental dimension (value 1, 10, 100, 1 000, 10 000). "Continuous traversing" must be canceled. If several increments are set at the same time, "Incremental travel relative" mode is cancelled.</p>
DBX102.5 INC_VAR	<p>Increments variable ¹⁾</p> <p>The INC value (incremental dimension) can be set variably (via HPU only).</p>
DBX102.6+n CONT_TRAV	<p>Continuous traversing</p> <p>Traverse for as long as Minus or Plus direction is actuated in "Jog" mode. Set: On deactivation of increment for traversal of axes in "Jog" mode Reset: When increment is set The signal is set as a default when the FM 357-2 is switched on.</p>
DBX103.4+n SGL_BL	<p>Activate single-block (submode of "Automatic")</p> <p>If the "Activate single-block" signal is selected in "Automatic" mode, the resulting submode is "Automatic single block" (see Section 9.12). Non-modal execution of NC programs</p>
DBX103.5+n M1	<p>Activate M01</p> <p>Activates the "M01" function in "Automatic" mode.</p>

1) with FM 357-2 handling only

Table 6-19 Signals for user DB "FMx", continued

Address symbol	Name Function
DBX103.6 DRUN_FD	<p>Activate dry run feed ¹⁾</p> <p>Instead of the programmed feedrate (with G1, G2, G3), the dry run feedrate programmed via SD 42100: DRY_RUN_FEED will be used for traversing if the dry run feed is larger than the programmed one.</p> <p>The interface signal is evaluated at start if the channel is in the initial status "Reset".</p> <p>With selection via CPU, the interface signal "Enable dry run feed" must be set from the user program.</p>
DBX104.7 PROG_TEST	<p>Activate program test ¹⁾</p> <p>Activates the "Program test" function in "Automatic" mode.</p> <p>An internal axis disable is defined for all axes. The axes do not move when an NC program block or NC program is executed. The axis position values are generated from the calculated setpoints.</p>
DBX105.0+n SKIP_BL	<p>Activate skip block</p> <p>Skips identified blocks in the program</p> <p>Is only effective in "Automatic" mode.</p> <p>The signal should be set before block coding.</p>

1) with FM 357-2 handling only

Table 6-19 Signals for user DB "FMx", continued

Address symbol	Name Function																																																																																																																																
DBB106+n FD_OVERR	<p>Path override</p> <p>This signal defines the override for rapid traverse and feed.</p> <p>The "Binary code or Gray code" coding must be set with parameter "Override coding" in the "Parameterize FM 357-2" tool (default setting: Gray code).</p> <p>The override is set by way of a 5-digit Gray code (can be preset by means of a switch element) or a binary code (bits 0 to 6).</p> <p>Range: 0 to 120 % for feed, 0 to 100 % for rapid traverse</p> <p>An override of 0 % has the same effect as a feed disable.</p> <p>The override must be set with "rapid traverse override active" (user DB "FMx", DBX107.6+n) or "feedrate override active" (user DB "FMx", DBX107.7+n).</p> <p>If the override is inactive, an override of 100 % is used (exception: setting 1 causes an override of 0 %).</p> <table border="1"> <thead> <tr> <th>Switch setting</th> <th>Gray code Bit 4, 3, 2, 1, 0</th> <th>Feed</th> <th>Rapid traverse</th> </tr> </thead> <tbody> <tr><td>1</td><td>00001</td><td>0.0 = 0 %</td><td>0.0 = 0 %</td></tr> <tr><td>2</td><td>00011</td><td>0.01 = 1 %</td><td>0.01 = 1 %</td></tr> <tr><td>3</td><td>00010</td><td>0.02</td><td>0.02</td></tr> <tr><td>4</td><td>00110</td><td>0.04</td><td>0.04</td></tr> <tr><td>5</td><td>00111</td><td>0.06</td><td>0.06</td></tr> <tr><td>6</td><td>00101</td><td>0.08</td><td>0.08</td></tr> <tr><td>7</td><td>00100</td><td>0.10</td><td>0.10</td></tr> <tr><td>8</td><td>01100</td><td>0.20</td><td>0.20</td></tr> <tr><td>9</td><td>01101</td><td>0.30</td><td>0.30</td></tr> <tr><td>10</td><td>01111</td><td>0.40</td><td>0.40</td></tr> <tr><td>11</td><td>01110</td><td>0.50</td><td>0.50</td></tr> <tr><td>12</td><td>01010</td><td>0.60</td><td>0.60</td></tr> <tr><td>13</td><td>01011</td><td>0.70</td><td>0.70</td></tr> <tr><td>14</td><td>01001</td><td>0.75</td><td>0.75</td></tr> <tr><td>15</td><td>01000</td><td>0.80</td><td>0.80</td></tr> <tr><td>16</td><td>11000</td><td>0.85</td><td>0.85</td></tr> <tr><td>17</td><td>11001</td><td>0.90</td><td>0.90</td></tr> <tr><td>18</td><td>11011</td><td>0.95</td><td>0.95</td></tr> <tr><td>19</td><td>11010</td><td>1.00</td><td>1.00</td></tr> <tr><td>20</td><td>11110</td><td>1.05</td><td>1.00</td></tr> <tr><td>21</td><td>11111</td><td>1.10</td><td>1.00</td></tr> <tr><td>22</td><td>11101</td><td>1.15</td><td>1.00</td></tr> <tr><td>23</td><td>11100</td><td>1.20</td><td>1.00</td></tr> <tr><td>24</td><td>10100</td><td>1.20</td><td>1.00</td></tr> <tr><td>25</td><td>10101</td><td>1.20</td><td>1.00</td></tr> <tr><td>26</td><td>10111</td><td>1.20</td><td>1.00</td></tr> <tr><td>27</td><td>10110</td><td>1.20</td><td>1.00</td></tr> <tr><td>28</td><td>10010</td><td>1.20</td><td>1.00</td></tr> <tr><td>29</td><td>10011</td><td>1.20</td><td>1.00</td></tr> <tr><td>30</td><td>10001</td><td>1.20</td><td>1.00</td></tr> <tr><td>31</td><td>10000</td><td>1.20</td><td>1.00</td></tr> </tbody> </table>	Switch setting	Gray code Bit 4, 3, 2, 1, 0	Feed	Rapid traverse	1	00001	0.0 = 0 %	0.0 = 0 %	2	00011	0.01 = 1 %	0.01 = 1 %	3	00010	0.02	0.02	4	00110	0.04	0.04	5	00111	0.06	0.06	6	00101	0.08	0.08	7	00100	0.10	0.10	8	01100	0.20	0.20	9	01101	0.30	0.30	10	01111	0.40	0.40	11	01110	0.50	0.50	12	01010	0.60	0.60	13	01011	0.70	0.70	14	01001	0.75	0.75	15	01000	0.80	0.80	16	11000	0.85	0.85	17	11001	0.90	0.90	18	11011	0.95	0.95	19	11010	1.00	1.00	20	11110	1.05	1.00	21	11111	1.10	1.00	22	11101	1.15	1.00	23	11100	1.20	1.00	24	10100	1.20	1.00	25	10101	1.20	1.00	26	10111	1.20	1.00	27	10110	1.20	1.00	28	10010	1.20	1.00	29	10011	1.20	1.00	30	10001	1.20	1.00	31	10000	1.20	1.00
Switch setting	Gray code Bit 4, 3, 2, 1, 0	Feed	Rapid traverse																																																																																																																														
1	00001	0.0 = 0 %	0.0 = 0 %																																																																																																																														
2	00011	0.01 = 1 %	0.01 = 1 %																																																																																																																														
3	00010	0.02	0.02																																																																																																																														
4	00110	0.04	0.04																																																																																																																														
5	00111	0.06	0.06																																																																																																																														
6	00101	0.08	0.08																																																																																																																														
7	00100	0.10	0.10																																																																																																																														
8	01100	0.20	0.20																																																																																																																														
9	01101	0.30	0.30																																																																																																																														
10	01111	0.40	0.40																																																																																																																														
11	01110	0.50	0.50																																																																																																																														
12	01010	0.60	0.60																																																																																																																														
13	01011	0.70	0.70																																																																																																																														
14	01001	0.75	0.75																																																																																																																														
15	01000	0.80	0.80																																																																																																																														
16	11000	0.85	0.85																																																																																																																														
17	11001	0.90	0.90																																																																																																																														
18	11011	0.95	0.95																																																																																																																														
19	11010	1.00	1.00																																																																																																																														
20	11110	1.05	1.00																																																																																																																														
21	11111	1.10	1.00																																																																																																																														
22	11101	1.15	1.00																																																																																																																														
23	11100	1.20	1.00																																																																																																																														
24	10100	1.20	1.00																																																																																																																														
25	10101	1.20	1.00																																																																																																																														
26	10111	1.20	1.00																																																																																																																														
27	10110	1.20	1.00																																																																																																																														
28	10010	1.20	1.00																																																																																																																														
29	10011	1.20	1.00																																																																																																																														
30	10001	1.20	1.00																																																																																																																														
31	10000	1.20	1.00																																																																																																																														

1) with FM 357-2 handling only

Table 6-19 Signals for user DB "FMx", continued

Address symbol	Name Function
DBX107.0+n FD_DISA	Activate feed disable Causes the axes of an FM 357-2 to stop (Feed Stop). After cancelation of the feed disable (FALSE), the feed is enabled for all axes of the channel (e.g. the interrupted NC program is resumed). If a travel request is outstanding when "Feed disable" is active, it is stored. The travel request is executed immediately on cancelation of the "Feed disable".
DBX107.1+n RDIN_DISA	Read-in disable This signal prevents read-in (processing) of the next block Is only effective in "Automatic" and "MDI" mode.
DBX107.2+n DELDIS_ TOGO	Delete distance to go "Edge signal" This signal is only effective in "Automatic" mode for path axes (geometry axes). The axes are stopped as quickly as possible and their residual distance (difference between setpoint and actual value) is deleted. Any outstanding following error is cleared. The next program block is then started.
DBX107.6+n FD_OVERR_ RTR	Path override, rapid traverse override active When traversing with rapid traverse, the override set in the "Path override" byte (user DB "FM", DBB106+n) is used.
DBX107.7+n FD_OVERR_ FDR	Path override, feedrate override active When traversing with feed, the override set in the "Path override" byte (user DB "FM", DBB106+n) is used.
DBX108.0+n START_DISA	Activate start disable Inhibits starting of an NC program. e.g. another start is inhibited because lubricant is missing
DBX108.1+n START	Start Starts movement in "Automatic" and "MDI" modes (see Section 9.12). Starting a motion interrupted by Stop. "Edge signal"
DBX108.2+n STOP_BL	Activate stop at block boundary This signal causes an NC program to stop at the block boundary.
DBX108.3+n STOP	Stop Interrupts movement or processing of the program Continuation of motion with Start. "Edge signal"
DBX108.7+n RESET	Reset Triggers a reset <ul style="list-style-type: none"> • Axes are decelerated • Program execution is interrupted (execution starts at beginning of program again) • The associated errors are cleared • "Channel ready" is set after a serious error (except for hardware errors)

1) with FM 357-2 handling only

Table 6-19 Signals for user DB “FMx”, continued

Address symbol	Name Function
DBX109.0+n ACKN_AUXILF	<p>Acknowledge auxiliary function</p> <p>The signal must be set only to acknowledge receipt of the auxiliary functions. Each auxiliary function which is output must be acknowledged before other actions (e.g. mode change). After acknowledgement, the program continues execution.</p> <p>“Edge signal”</p> <p>Please follow the instructions in Section 6.5.2!</p>
DBX110.7+n START_ASUB	<p>Start ASUB (asynchronous subprogram)</p> <p>see Section 6.5.3</p>
DBX114.3 DBX115.3 DBX116.3 DBX117.3 FD_STOP_GEOA1...3, FD_STOP_ORIA1	<p>Geometry axis 1...3, orientation axis 1¹⁾</p> <p>Feed stop</p> <p>see interface signal “Feed stop”, user DB “AXy”, DBX4.3+m</p>
DBX114.4 DBX115.4 DBX116.4 DBX117.4 TR_DISA_GEOA1...3, TR_DISA_ORIA1	<p>Geometry axis 1...3, orientation axis 1¹⁾</p> <p>Traversing disable</p> <p>This signal is used to disable traversing of the geometry axis/orientation axis in “Jog” mode using the “Direction minus or plus” signals (user DB “FMx”, DBX114.6/7; 115.6/7; 116.6/7; 117.6/7).</p> <p>The axis cannot be moved with the “Direction minus or plus” signals until the traversing disable signal has been reset.</p>
DBX114.5 DBX115.5 DBX116.5 DBX117.5 RTR_OVERL_GEOA1...3, RTR_OVERL_ORIA1	<p>Geometry axis 1...3, orientation axis 1¹⁾</p> <p>rapid traverse override</p> <p>see interface signal “Activate rapid traverse override”, user DB “AXy”, DBX4.5+m</p>
DBX114.6 DBX115.6 DBX116.6 DBX117.6 DIR_M_GEOA1...3, DIR_M_ORIA1	<p>Geometry axis 1...3, orientation axis 1¹⁾</p> <p>Direction minus</p> <p>see interface signal “Direction minus”, user DB “AXy”, DBX4.6+m</p> <p>Note:</p> <p>Only one geometry axis can be traversed at a time via “Direction minus”.</p>
DBX114.7 DBX115.7 DBX116.7 DBX117.7 DIR_P_GEOA1...3, DIR_P_ORIA1	<p>Geometry axis 1...3, orientation axis 1¹⁾</p> <p>Direction plus</p> <p>see interface signal “Direction plus”, user DB “AXy”, DBX4.7+m</p> <p>Note:</p> <p>Only one geometry axis can be traversed at a time via “Direction plus”.</p>

1) with FM 357-2 handling only

Table 6-19 Signals for user DB "FMx", continued

Address symbol	Name Function
Channel checkback signals [starts at channel 1; each further channel (n) plus offset 100]	
DBX120.0+n MODE_A	"Automatic" mode active "Automatic" mode is active If the "Start" signal (user DB "FMx", DBX108.1+n) is connected directly to the AUTOMATIC mode (user DB "FMx", DBX108.1+n), error message 16 925 may occur, depending on the CPU cycle time. In this case, delay the "Start" signal by one CPU cycle or additionally link the "Channel reset" signal (user DB "FMx", DBX125.7+n) accordingly.
DBX120.1+n MODE_MDI	"MDI" mode active Can only be set with "Parameterize FM 357-2" tool or with "FM 357-2 Handling". "MDI" mode is active
DBX120.2+n MODE_JOG	"Jog" mode active "Jog" mode is active
DBX121.0+n TEACH_IN	"TEACH IN" active submode of "MDI"¹⁾ Submode "TEACH IN" is active
DBX121.1+n REPOS	"REPOS" active submode of "Jog" or "MDI"¹⁾ Submode "REPOS" is active
DBX121.2+n REF_POINT	"Reference point approach" active submode of "Jog" Submode "Reference point approach" is active
DBX122.0+n to DBX122.4+n INC1...10000	Active incremental dimension (submode of "Jog") This signal indicates which incremental dimension is selected in "Incremental Relative" mode. The signal is only active for geometry axes. For special axes, the "Active increment" is indicated in user DB "AXy", DBX24.0...4.
DBX122.5+n INC_VAR	Incremente variable ¹⁾ This signal indicates which incremental dimension is selected in "Incremental Relative" mode. The signal is only active for geometry axes. For special axes, the "Active increment" is indicated in user DB "AXy", DBX24.5.
DBX122.6+n CONT_TRAV	Continuous traversing active Traversing in plus/minus direction is possible in "Jog" mode. The signal is only active for geometry axes. For special axes, "Continuous traversing" is indicated in user DB "AXy", DBX24.6.
DBX123.5+n M0_M1	M00/M01 active M functions are active, for meaning see Section 10.14 This signal is not active until the associated auxiliary functions have been acknowledged.
DBX124.5+n M2_M30	M02/M30 active M functions are active, for meaning see Section 10.14 This signal is not active until the associated auxiliary functions have been acknowledged.

1) with FM 357-2 handling only

Table 6-19 Signals for user DB “FMx”, continued

Address symbol	Name Function
DBX124.6+n TRANSF	Transformation active ¹⁾ Transformation is active
DBX124.7+n PROG_TEST	Program test active ¹⁾ Program test is active
DBX125.0+n DBX125.1+n DBX125.2+n DBX125.3+n DBX125.4+n PROG_RUN PROG_WAIT PROG_STOP PROG_INT PROG_AB	Program running Program is waiting Program is stopped Program is interrupted Program is aborted Shows the status of the program (see Chapter 10).
DBX125.7+n RESET	Channel reset ... acknowledges reset triggered for the channel or the channel which is already in the reset condition. In addition to the reset acknowledgment, this signal indicates the following: <ul style="list-style-type: none"> • A mode change is possible. • A program can be selected or started in the AUTOMATIC mode.
DBX126.2+n ALLAX_REF	All axes referenced This signal indicates that all axes which need to be referenced have been referenced.
DBX126.3+n ALLAX_STOP	All axes stationary This signal indicates that all axes of the FM 357-2 are stationary
DBX126.5+n CH_RDY	Channel ready This signal is set after the power has been switched on and all voltages have been built up. The channel is now ready for operation and NC programs can be run or axes traversed. The signal is reset on: <ul style="list-style-type: none"> • Serious errors (e.g. hardware errors of active encoders) • Serious axis errors • Incorrect parameterization on the FM
DBX126.6+n ERR	Channel error An error has occurred in the channel. The “Error, program stopped” indicates whether NC program execution has been interrupted or aborted.
DBX126.7+n ERR_STOP	Channel error, program stopped An error has occurred in the channel. NC program execution has been interrupted or aborted (machining standstill).
DBX127.0+n CHANG_AUXILF	Change auxiliary function This signal indicates that at least one auxiliary function has been output The signal is reset in the next CPU cycle. For evaluation, see Section 6.5.2

1) with FM 357-2 handling only

Table 6-19 Signals for user DB "FMx", continued

Address symbol	Name Function
DBX128.0+n DBX128.1+n DBX128.2+n DBX128.3+n ASUB_ACT ASUB_DONE ASUB_ERR ASUB_ STARTERR	ASUB status: active ASUB status: terminated ASUB status: error ASUB status: start error see Section 6.5.3
DBX134.6 DBX135.6 DBX136.6 DBX137.6 GO_M_ GEO1...3, GO_M_ORIA1	Geometry axis 1...3, orientation axis 1¹⁾ Travel minus see interface signal "Travel minus", user DB "AXy", DBX23.6+m
DBX134.7 DBX135.7 DBX136.7 DBX137.7 GO_P_ GEOA1...3, GO_P_ORIA1	Geometry axis 1...3, orientation axis 1¹⁾ Travel plus see interface signal "Travel plus", user DB "AXy", DBX23.7+m
Auxiliary functions [starts at channel 1; each further channel (n) plus offset 100]	
DBB140+n to DBB152+n MF0...99	M functions (decoded) M0...M99 Result from M function number 1...5 (user DB "FMx", DBB158...162+n) The signal is reset in the next CPU cycle.
DBX154.0+n to DBX154.4+n STR_MF1...5	Change M functions (M function 1...5) This signal indicates which M function number (user DB "FMx", DBB158...162+n) has changed. The signal is reset in the next CPU cycle.
DBX155.0+n STR_TF1	Change T functions (T function 1) This signal indicates that the T function number (user DB "FMx", DBW164+n) has changed. The signal is reset in the next CPU cycle.
DBX156.0+n to DBX156.2+n STR_HF1...3	Change H functions (H function 1...3) This signal indicates which H function number/value (user DB "FMx", DBB166...180+n) has changed. The signal is reset in the next CPU cycle.
DBB158+n to DBB162+n NUM_M1...5	M function number 1...5 Number according to NC program

1) with FM 357-2 handling only

Table 6-19 Signals for user DB “FMx”, continued

Address symbol	Name Function
DBW164+n NUM_T1	T function number Number according to NC program
DBW166+n to DBD180+n NUM_H1...3 FVAL_H1...3	H function number 1...3 H function value 1...3 (REAL) Number according to NC program
Write request 1 (general FM 357-2 signals)	
DBB501 DBX503.0...3 INP1...12	Set digital on-board inputs 1...12 from CPU see Section 9.10.
DBB504 DISA_INP1...8	Disable digital on-board outputs 1...8 A disable can be defined for output 1...8. The output is then set to FALSE and can no longer be influenced. see Section 9.10
DBB506 OUTP1...8	Set digital on-board outputs 1...8 from CPU see Section 9.10.
DBB507 ST_OUTP1...8	Define digital on-board outputs 1...8 from CPU ↔ FM Assignment: FALSE: Digital output set by FM TRUE: Digital output set by CPU see Section 9.10.
DBB509 DBB511 EINP17...32	Set digital inputs on local P bus (input 17 to 32) see Section 9.10.
DBB516 DBB520 DISA_EOUTP 9...24	Disable digital outputs on local P bus (output 9 to 24) A disable can be defined for output 9...24. The output is then set to FALSE and can no longer be influenced. see Section 9.10
DBB518 DBB522 EOUTP9...24	Set digital outputs on local P bus (output 9 to 24) see Section 9.10.
DBB519 DBB523 ST_EOUTP9... 24	Define digital outputs on local P bus (output 9 to 24) see Section 9.10.

1) with FM 357-2 handling only

Table 6-19 Signals for user DB "FMx", continued

Address symbol	Name Function
Read request 1 (general FM 357-2 signals)	
DBB540 DBX541.0...3 INP1...12	Status of digital on-board inputs 1...12 from FM see Section 9.10.
DBB542 OUTP1...8	Status of digital on-board outputs 1...8 from FM see Section 9.10.
DBB544 DBB545 EINP17...32	Status of digital inputs on local P bus (input 17 to 32) This signal indicates the status of digital inputs 17...32 on the local P bus. see Section 9.10
DBB548 DBB549 EOUTP9...24	Status of digital outputs on local P bus (output 9 to 24) This signal indicates the status of digital outputs 9...24 on the local P bus. see Section 9.10
Write request 2 (channel signals)	
DBB580 DBB586 DBB592 DBB598 DISA_SYN1...8	Channel 1; Disable synchronized action ID1...8 Channel 2; Disable synchronized action ID1...8 Channel 3; Disable synchronized action ID1...8 Channel 4; Disable synchronized action ID1...8 Disable modal or static synchronized action with ID no. 1 to 8 (see Section 10.32)
DBX582.0...3 DBX588.0...3 DBX594.0...3 DBX600.0...3 PRZONE1...4	Channel 1; Activate protection zone 1...4 Channel 2; Activate protection zone 1...4 Channel 3; Activate protection zone 1...4 Channel 4; Activate protection zone 1...4 see Sections 9.15 and 10.17
DBX584.4 PTP_TRAV	Point-to-point traversing ¹⁾ = TRUE, Suppress transformation, enable → traversing in machine coordinate system (MCS) = FALSE, CP traversing activate (CP = cartesian path movement) see Section 13.
Read request 2 (channel signals)	
DBB620 DBB628 DBB636 DBB644 DISA_SYN1...8	Channel 1; Synchronized action ID1...8, disable possible from CPU Channel 2; Synchronized action ID1...8, disable possible from CPU Channel 3; Synchronized action ID1...8, disable possible from CPU Channel 4; Synchronized action ID1...8, disable possible from CPU Checkback signal of FM 357-2 indicating that the synchronized actions (ID no. 1...8 =TRUE) can be disabled from the CPU.
DBX622.0...3 DBX630.0...3 DBX638.0...3 DBX646.0...3 PRZONE1...4	Channel 1; Protection zone 1...4 preactivated Channel 2; Protection zone 1...4 preactivated Channel 3; Protection zone 1...4 preactivated Channel 4; Protection zone 1...4 preactivated This signal indicates the protection zones preactivated by the NC program.

1) with FM 357-2 handling only

Table 6-19 Signals for user DB “FMx”, continued

Address symbol	Name Function
DBX624.0...3 DBX632.0...3 DBX640.0...3 DBX648.0...3 PRZONE1...4_ VIOL	Channel 1; Protection zone 1...4 violated Channel 2; Protection zone 1...4 violated Channel 3; Protection zone 1...4 violated Channel 4; Protection zone 1...4 violated This signal indicates the preactivated protection zones which are violated by the NC program without an override movement.
DBX626.6 PTP_TRAV	Point-to-point traversing active ¹⁾ = TRUE, Transformation is suppressed → traversing in machine coordinate system (MCS) = FALSE, CP traversing activate (CP = cartesian path movement) see Section 13.
DBX627.6 DRUN_FD	Dry run feed active ¹⁾ Dry run feed has been selected. The signal becomes active with the next start.

1) with FM 357-2 handling only

Table 6-20 describes the control/checkback signals of user data block “AXy” and their function.

Table 6-20 Signals for user DB “AXy”

Address symbol	Name Function
Axis control signals [starts at axis 1; each further axis (m) plus offset 100]	
DBB0+m OVERR	<p>Override</p> <p>This signal indicates the override value (see path override user DB “FMx” DBB106+n) for the axis.</p> <p>The specifications for the feed apply. The override must be activated with “Activate override” (user DB “AXy”, DBX1.7+m).</p>
DBX1.1+m ACKN_FISTOP	<p>Acknowledge fixed stop reached</p> <p>This signal is relevant only if parameter “Acknowledgement signal” is set (see Section 9.18). A block change can be executed.</p> <p>This signal cannot be reset until the “Travel to fixed stop” function is deselected. If it is reset prematurely, an error message is output and the function aborted.</p>
DBX1.2+m SENS_FISTOP	<p>Fixed stop sensor</p> <p>This signal is relevant only if parameter “Fixed stop detection” is parameterized with “external sensor” (see Section 9.18).</p>
DBX1.4+m FOLLOWUP	<p>Follow-up mode</p> <p>This signal switches the axis to follow-up mode, i.e. the set position follows the actual position. Follow-up mode is indicated by the checkback signal “Follow-up mode active” (user DB “AXy”, DBX21.3+m).</p> <p>When follow-up mode is canceled it is not necessary to re-reference the axis.</p> <p>The response is as follows, according to the servo enable:</p> <ul style="list-style-type: none"> • Follow-up mode = TRUE <p>When the servo enable is canceled, the position setpoint follows the actual value.</p> <p>When the “servo enable” is reactivated, all further axis movements begin at the new actual position.</p> • Follow-up mode = FALSE <p>When the servo enable is canceled, the old setpoint is retained. If the axis is moved out of position, a following error is produced. The system compensates for this error when you activate the servo enable.</p> <p>When the servo enable is active, follow-up mode has no effect.</p>
DBX1.7+m ACT_OVERR	<p>Activate override</p> <p>When the axis traverses, the override set in “Override” (user DB “AXy”, DBB0+m) is applied.</p>
DBX2.0+m SWCAM	<p>Activate software cam</p> <p>All cam pairs assigned to this axis are activated with this signal (see Section 9.11).</p>

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
- 2) with FM 357-2 handling only

Table 6-20 Signals for user DB "AXy", continued

Address symbol	Name Function
DBX2.1+m CTR_EN	<p>Controller enable</p> <p>Closes the position control loop of the axis.</p> <p>When the controller enable signal is cancelled, the position control loop is opened.</p> <p>The controller enable signal must always be TRUE for axes with a stepper motor.</p> <p>You can set and cancel the servo enable as follows:</p> <ul style="list-style-type: none"> • using this signal (normal condition) • within the control (error condition)
DBX2.2+m DELDIS_ TOGO	<p>Axial deletion of distance to go</p> <p>"Edge signal"</p> <p>"Jog" mode: The axis is stopped as quickly as possible and its residual distance (difference between setpoint and actual value) is deleted. Any outstanding following error is cleared.</p> <p>"Automatic" and "MDI" mode (with FM 357-2 Handling): Is effective only for axes which are not geometry axes. The axes are stopped and their residual distance (difference between setpoint and actual value) is deleted. The next program block can then be started.</p>
DBX2.3+m CLAMP	<p>Activate clamp</p> <p>Activates clamp monitoring (travel to fixed stop)</p>
DBX3.1+m FISTOP_EN	<p>Enable travel to fixed stop</p> <p>This signal is relevant only if parameter "Acknowledgement signal" is set (see Section 9.18).</p> <p>If the signal is reset before the fixed stop has been reached, an error message is output and the function aborted.</p>
DBX4.3+m FD_STOP	<p>Feed stop</p> <p>The signal initiates a "Feed stop" on the corresponding axis. If the axis is moving, it initiates a controlled deceleration to standstill.</p> <p>If a travel request is outstanding when "Feed stop" is activated, it is stored. The travel request is executed immediately on cancellation of the "Feed stop".</p>
DBX4.5+m RTR_OVERL	<p>Activate rapid traverse override</p> <p>When traversing in "Jog" mode and "Incremental relative", this signal activates rapid traverse.</p>
DBX4.6+m DIR_M	<p>Direction minus</p> <p>Moves the axis in the negative direction in the following modes:</p> <ul style="list-style-type: none"> • "Jog" • "Incremental Relative" • "Reference-Point Approach" <p>If the "direction minus and plus" signals are set simultaneously, there is no traversing movement or an existing traversing movement is aborted.</p>

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
- 2) with FM 357-2 handling only

Table 6-20 Signals for user DB "AXy", continued

Address symbol	Name Function
DBX4.7+m DIR_P	Direction plus Moves the axis in the positive direction in the following modes: <ul style="list-style-type: none"> • "Jog" • "Incremental Relative" • "Reference-Point Approach" If the "direction plus and minus" signals are set simultaneously, there is no traversing movement or an existing traversing movement is aborted.
DBX7.0/1+m HWLIMIT_M HWLIMIT_P	Hardware limit switch minus/plus The user program must transfer the signal of the hardware limit switch minus or plus for an axis to this signal via an input (see Section 9.6).
DBX7.2/3+m SWLIMIT2_M SWLIMIT2_P	2nd software limit switch minus/plus This signal can be used to activate the 2nd software limit switch minus or plus (see Section 9.6). The 1st software limit switch is then no longer active.
DBX7.7+m REF_DELAY	Delay reference point approach The user program must transfer the reference point switch signal to this signal via an input (see Section 9.7, Figure 9-14).
DBX8.0+m ROT_MON	Stepper motor rotation monitoring This signal can be used on axes with stepper motors without encoders to activate/de-activate the rotation monitoring (see Section 9.6).
DBX8.1+m MODE_CON- TROL	Controlling an axis Traversing an axis in speed-controlled mode of the drive (see Section 9.18)
DBX8.4+m FOAX_OVERL	Enable following axis override see Section 9.16.4
DBX9.5+m OSSTOP_ REVP	Oscillation, stop at next reversal point Oscillation axis stops at reversal point
DBX9.6+m OSSTOP	Oscillation, Stop Oscillation axis entered
DBX10.4+m START_ GSYNRUN	Start gantry synchronization run (gantry leading axis) Interface signal for master axis in gantry grouping The FM 357-2 starts the synchronization process (see Section 9.16.2).
DBX12.1+m RUP_ENC_ QSTOP	Ramp-function generator rapid stop ¹⁾ The user program requests a rapid stop for the drive. The drive is then brought to a standstill (with speed setpoint 0). The controller enable remains active (see SIMODRIVE 611 universal, control word 1, bits 4 and 5).
DBX12.3+m SPEED_ SMOOTH	Speed setpoint smoothing ¹⁾ The user program requests a filter for smoothing the speed setpoint for the axis (see SIMODRIVE 611 universal, control word 2, bits 3).

1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation

2) with FM 357-2 handling only

Table 6-20 Signals for user DB “AXy”, continued

Address symbol	Name Function
DBX13.0...2+m PAR_A PAR_B PAR_C	Select parameter set A/B/C ¹⁾ The user program can select up to eight different parameter sets with bit combination A, B and C (see SIMODRIVE 611 universal, control word 2, bits 0 to 2).
DBX13.6+m INTEGR_DISA	Integrator disable, speed controller ¹⁾ The user program disables the integrator of the speed controller for the drive. This action switches the speed controller from PI to P control (see SIMODRIVE 611 universal, control word 2, bits 6).
DBX13.7+m PULSE_EN	Pulse enable The user program issues the pulse enable for the axis. The pulse enable for the drive is only issued if all enable signals are active.
Axis checkback signals [starts at axis 1; each further axis (m) plus offset 100]	
DBX20.4+m REF_SYNC	Synchronized, referenced The axis is referenced/synchronized
DBX20.6/7+m POSROD_C POSROD_F	Position reached, stop (target range coarse/fine) The axis is within the target range coarse or fine (see Section 9.6).
DBX21.3+m FOLLOWUP	Follow-up mode active Follow-up mode is active in this axis.
DBX21.4+m STAT	Axis stationary The axis has a velocity that is smaller than the value in the parameter “Threshold velocity for stationary axis”.
DBX21.5+m POSCTR	Position controller active This signal indicates whether the position controller is active (see Section 9.3).
DBX21.6+m SPEEDCTR	Speed controller active This signal indicates whether the speed controller is active (see Section 9.3).
DBX22.0+m SWCAM	Software cams active This signal indicates that all cam pairs assigned to an axis and the cam signal generation are active (see Section 9.11).
DBX22.3+m MSR_EN	Measurement active The “Measurement” function is active.
DBX22.4+m TRAVFI_STOP	Travel to fixed stop is active The “Travel to fixed stop” function is active.
DBX22.5+m FI_STOP	Fixed stop reached The fixed stop has been reached
DBX23.0+m MASTERAX	Master axis active (electronic gear) see Section 10.35

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
 2) with FM 357-2 handling only

Table 6-20 Signals for user DB "AXy", continued

Address symbol	Name Function
DBX23.1+m FOAX	Following axis active (electronic gear) see Section 10.35
DBX23.3+m AX_ACCEL	Axis is accelerating (electronic gear) see Section 9.16.4
DBX23.4+m SYNC_RUN	Synchronization is running (electronic gear) see Section 10.35
DBX23.6+m GO_M	Travel minus The signal is set as soon as an active traversing movement in the negative direction is pending.
DBX23.7+m GO_P	Travel plus The signal is set as soon as an active traversing movement in the positive direction is pending.
DBX24.0+m to DBX24.4+m INC1...10000	Active incremental dimension (submode of "Jog") This signal indicates which incremental dimension (value 1, 10, 100, 1 000, 10 000) is selected in "Incremental Relative" mode. The signal is only active for special axes. For geometry axes, the active increment is indicated in user DB "FMx", DBX122.0...4.
DBX24.5+m INC_VAR	Incremente variable ²⁾ This signal indicates which incremental dimension is selected in "Incremental Relative" mode. The signal is only active for special axes. For geometry axes, the active increment is indicated in user DB "FMx", DBX122.5.
DBX24.6+m CONT_TRAV	Continuous traversing Traversing in plus/minus direction is possible in "Jog" mode. The signal is only active for special axes. For geometry axes, "Continuous traversing" is indicated in user DB "FMx", DBX122.6.
DBX25.0...3+m ATCHAN_A ATCHAN_B ATCHAN_C ATCHAN_D	Axis in channel A/B/C/D (binary) The signal specifies in which channel the axis is configured.
DBX25.5+m INTER- CHANGE_P	Axis change possible An axis change is possible via appropriate NCK statements (see Section 10.36).
DBX25.6+m NEUTRAL	Neutral axis The axis is in a neutral state (see Section 10.36).
DBX28.0/1+m SYNC_F SYNC_C	Synchronism fine/coarse Status of master-value link (see Section 9.16.3)

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
- 2) with FM 357-2 handling only

Table 6-20 Signals for user DB "AXy", continued

Address symbol	Name Function
DBX28.4+m BLEN- DED_MOT	Overlaid motion see Section 10.35
DBX28.5+m SPEED_WAR N	Warning threshold "Velocity reached" see Section 10.35
DBX28.6+m ACCEL_WARN	Warning threshold "Acceleration reached" see Section 10.35
DBX29.0+m ERR_ROT_ MON	Error in stepper motor rotation monitoring The rotation monitoring circuit for this axis has detected an error.
DBX29.1+m MODE_CON- TROL	Axis control active The axis is traversing in speed-controlled mode of the drive. The signal is only set as long as the axis is traversed. When the axis stops, it is switched to position control (see section 9.18).
DBX29.6+m OS_MOT	Oscillation motion active Oscillation axis is in motion
DBX29.7+m OS_ACT	Axis is oscillation axis Axis is oscillation axis
DBX30.2+m GAN_TRLIM_ EX	Gantry trip limit exceeded (gantry synchronized axis) This signal indicates whether the value defined in parameter "Trip limit" has been exceeded (see Section 9.16.2).
DBX30.3+m GAN_LIM_EX	Gantry limit value for warning exceeded (gantry synchronized axis) This signal indicates whether the value defined in parameter "Limit value for warning" has been exceeded (see Section 9.16.2).
DBX30.4+m GSYNRUN_ RDY	Gantry synchronization ready to start (gantry leading axis) The synchronization run can be started with interface signal "Start gantry synchronization" (user DB "AXy", DBX10.4+m, see Section 9.16.2).
DBX30.5+m GANGR_SYN	Gantry grouping is synchronized (gantry leading axis) This signal indicates that the synchronization run is finished (see Section 9.16.2).
DBX30.6+m GAN_MASTAX	Gantry master axis Axis is the master axis in the gantry grouping (see Section 9.16.2).
DBX30.7+m GAN_AX	Gantry axis Axis is master or synchronized axis (see Section 9.16.2)
DBX31.0+m SETUP_MD	Setup mode ¹⁾ Setup mode is active on the drive.
DBX31.1+m RUP_ENC_ DISA	Ramp-function generator disable¹⁾ The drive indicates to the CPU that the ramp-function generator rapid stop is active. This brings the drive to a standstill (with speed setpoint 0).

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
 2) with FM 357-2 handling only

Table 6-20 Signals for user DB "AXy", continued

Address symbol	Name Function
DBX31.2+m TORQ_LIM2	Torque limit 2 ¹⁾ The drive indicates to the CPU that torque limit 2 is active for the axis. The torque limit value is defined by the drive parameters.
DBX31.3+m SPEED_SMOOTH	Speed setpoint smoothing ¹⁾ The user program requests a filter for smoothing the speed setpoint for the axis.
DBX32.0...2+m PAR_A PAR_B PAR_C	Active parameter set A/B/C ¹⁾ The drive indicates to the CPU which set of parameters is currently active.
DBX32.5+m DRV_RDY	Drive ready ¹⁾ Checkback signal indicating that the drive is ready for operation. The drive conditions necessary in order to move the axis have thus been met.
DBX32.6+m INTEGR_DISA	Integrator disable, speed controller ¹⁾ The integrator of the speed controller is disabled. The speed controller has therefore been switched from PI to P control.
DBX32.7+m PULSE_EN	Pulse enable The pulse enable for the drive is active. The axis can be moved.
DBX33.0+m TEMP_MOTOR	Motor temperature prewarning ¹⁾ The drive indicates to the CPU that the motor temperature has exceeded the warning threshold. If the motor temperature remains too high, the drive is shut down after a defined time (in the drive machine data) and the pulse enable is canceled.
DBX33.1+m TEMP_HSINK	Heatsink temperature prewarning ¹⁾ The drive indicates to the CPU that the heatsink temperature has exceeded the warning threshold. The pulse enable for the drive in question is canceled after 20 seconds.
DBX33.2+m RAMP_COMPL	Ramp-up operation finished ¹⁾ The signal indicates that the actual speed value has reached the new setpoint, allowing for the tolerance band defined in drive machine data (MD 1426: SPEED_DES_EQ_ACT_TOL). The ramp-up procedure is thus finished. Subsequent speed fluctuations resulting from load changes have no effect on the signal.
DBX33.3+m MD_MDX	$M_d < M_{dx}$ ¹⁾ The signal indicates that the current torque $ M_d $ is less than the threshold torque M_{dx} set in drive machine data (MD 1428: TORQUE_THRESHOLD_X).
DBX33.4+m NACT_NMIN	$n_{act} < n_{min}$ ¹⁾ The signal indicates that the actual speed value $ n_{act} $ is less than the minimum speed n_{min} defined in drive machine data (MD 1418: SPEED_THRESHOLD_MIN).
DBX33.5+m NACT_NX	$n_{act} < n_x$ ¹⁾ The signal indicates that the actual speed value $ n_{act} $ is less than the speed threshold n_x defined in drive machine data (MD 1417: SPEED_THRESHOLD_X).

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
 2) with FM 357-2 handling only

Table 6-20 Signals for user DB “AXy”, continued

Address symbol	Name Function
DBX33.6+m NACT_NSET	$n_{act} = n_{set}$ ¹⁾ The signal indicates to the CPU that the actual speed value n_{act} has reached the new setpoint, allowing for the tolerance band defined in drive machine data (MD 1426: SPEED_DES_EQ_ACT_TOL) and that it is still within the tolerance band.
DBX33.7+m MSG_FUNCT	Variable message function ¹⁾ The variable message function can be used by the drive to monitor any variable for each axis and to send a signal to the CPU if a definable threshold is exceeded. The variable to be monitored is defined in the machine data for the drive.
DBD40+m ACT_POS	Actual value The actual value of the axis after “Request actual value” (user DB “FMx”, DBX21.1) has been set.
DBD44+m DIS_TOGO	Distance to go The current distance to go of the axis after “Request distance to go” (user DB “FMx”, DBX21.2) has been set.
Control signals axis control from the CPU [starts at axis 1; each further axis (m) plus offset 100]	
DBX50.7+m CPUAX	Request CPU axis Requests an axis for positioning and parameter input via the CPU
DBX51.0+m INCR	Traversing path incremental see Section 6.5.1
DBX51.1+m DC	Traverse shortest path see Section 6.5.1
DBX51.2+m INCH	Traversing dimension inches see Section 6.5.1
DBX51.7+m START	Start CPU axis see Section 6.5.1
DBD52+m POS	Position of CPU axis see Section 6.5.1
DBD56+m FRATE	Feedrate of CPU axis see Section 6.5.1
DBX65.1+m RESET_ INDEPCPUAX	Independent CPU axis – Reset The motion of an independent CPU axis is aborted (see Section 6.5.1)
DBX65.2+m RESUME_ INDEPCPUAX	Independent CPU axis – Continue motion The motion of an independent CPU axis is continued after stop (see Section 6.5.1)
DBX65.6+m STOP_ INDEPCPUAX	Independent CPU axis – Stop The independent CPU axis is stopped (see Section 6.5.1)

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
 2) with FM 357-2 handling only

Table 6-20 Signals for user DB "AXy", continued

Address symbol	Name Function
DBX65.7+m INDEPCPUAX	Independent CPU axis – Request The independent CPU axis is requested (see Section 6.5.1)
Checkback signals axis control from the CPU [starts at axis 1; each further axis (m) plus offset 100]	
DBX66.7+m CPUAX	CPU axis The axis is assigned to the CPU (positioning). Acknowledgement signal of "Request CPU axis" (user DB "AXy", DBX50.7+m).
DBX67.1+m ERR	CPU axis error see Section 6.5.1
DBX67.6+m INPOS	Position reached see Section 6.5.1
DBX67.7+m ACTIV	CPU axis active see Section 6.5.1
DBB68+m ERR_NO	Error number of CPU axis see Section 6.5.1
DBX69.0+m RESET_ INDEPCPUAX	Independent CPU axis – Reset acknowledgement Reset is carried out for an independent CPU axis (see Section 6.5.1)
DBX69.1+m INDEPCPUAX	Independent CPU axis – is requested The axis is an independent CPU axis (see Section 6.5.1)
DBX69.2+m ERR_ INDEPCPUAX	Independent CPU axis – error An error has occurred when traversing an independent CPU axis. Error message, see Section 12.3.

- 1) Only for SIMODRIVE 611-U drives via PROFIBUS DP, for further information on signal descriptions, please refer to the drive documentation
- 2) with FM 357-2 handling only

6.7 Timing diagrams, communication

Example 1

Centralized installation (transfer time is not increased by other call levels (OBs) in the user program/CPU).

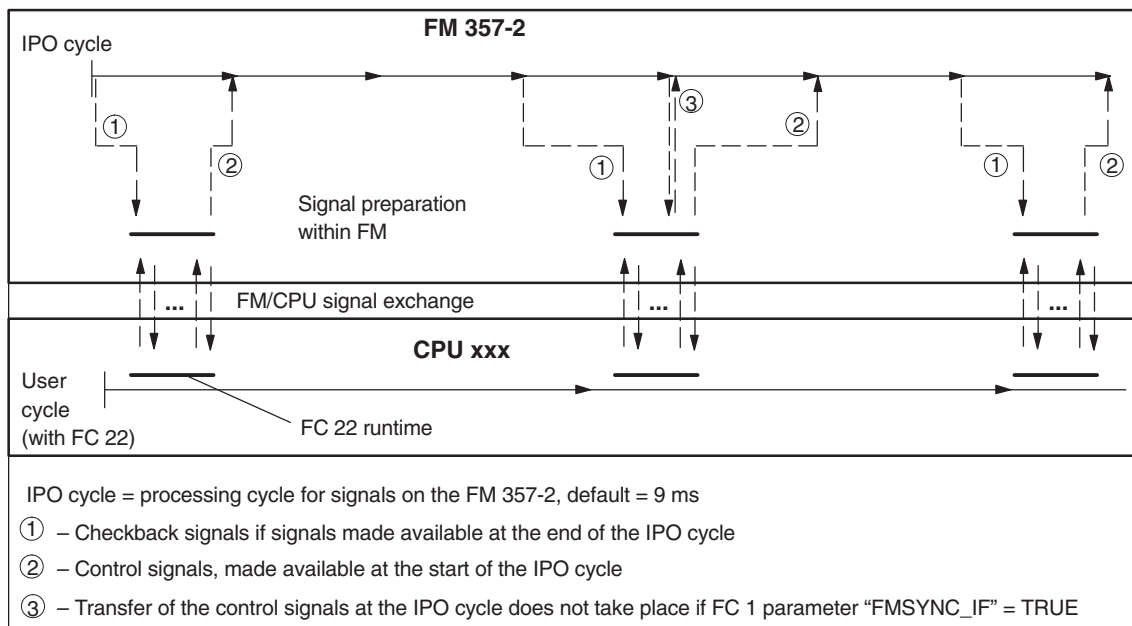


Fig. 6-12 Timing diagram, communication (centralized installation)

Example 2

Distributed installation or centralized installation (transfer time is increased by other call levels (OBs) in user program/CPU).

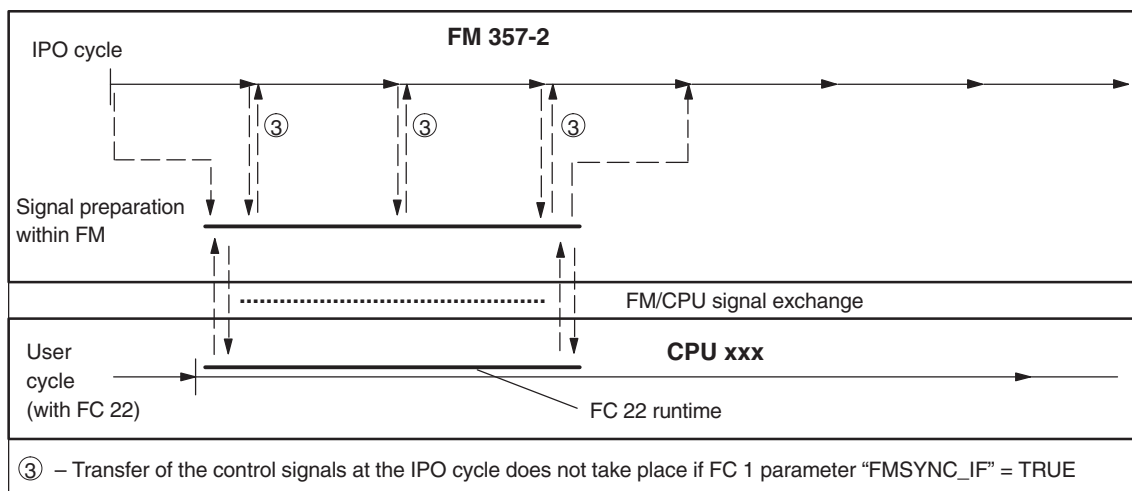


Fig. 6-13 Timing diagram, communication

6.8 Application examples

Overview

In this section, you can find information about:

- Example 1, axes moving in “Reference point approach” submode, page 6-101
- Example 2, axes moving in “Jog” mode, page 6-101
- Example 3, Starting an axis with axis control from CPU, page 6-102
- Example 4, “Automatic mode” with program selection (with FB 4), page 6-103
- Example 5, Reading and writing FM variables (with FB 2 and FB 3), page 6-104
- Example 6, Reading and writing R parameters (with FB 2 and FB 3), page 6-106
- Example 7, Fault diagnostics, page 6-108

General

After installation of the FM 357-2 configuration package, example project “[**STEP7 directory**]\EXAMPLES\zEn16_01_FM357-2_BF_EX” is installed.

The STL source “OB_EXAMPLE” (template for further examples) contains processing levels OB 100, OB 82 and OB 1 and the corresponding FCs (FC 1, FC 5, FC 22). DB 115 (USERDB) with the user signals/user data necessary for all the examples is contained in the “USERDB” source.

Each example is programmed as an FC (example 1 → FC 100 etc.). To use the functions of examples 2 to 7, the appropriate FCs should be called up in OB 1 as described in example 1. Examples 1 to 7 are self-contained.

Parameter “FMCOMM” of FC 1 is set by default in OB 100 (FMCOMM=TRUE). This parameter is required for CPU ↔ FM communication services in examples 4, 5 and 6.

Since the “USERDB” is a retentive DB, it must be initialized by the user during start-up (see OB 1, initialization of examples 4, 5 and 6 after calling FC 22).

Notice

If you are using simulation axes, you must set the machine data “VDI output” (see Sections 5.5.1 and 9.1).

Library “FM357_2L” contains all basic function blocks and OB 1, 82, 100. The appropriate basic function blocks (FC 22, FC 5, FC 1) must be called in the OBs. A suggestion is programmed in OB 1 in the “USER PROGRAM” section indicating how the user can integrate the bits for test mode using the startup user interface (see Section 6.2). These calls and the OBs were the basis on which STL source “OB_EXAMPLE” was created. The communication ready signal scanning routine and “EXAMPLE1” call are also integrated.

6.8.1 Example 1, axes moving in “Reference point approach” sub-mode

In the SIMATIC Manager, select **File > Open... > Projects** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Quellen\EXAMPLE1”.

The signals are stored in “USERDB”.

Reference point approach is a submode of “Jog” mode. Both modes are therefore preset in the example. Pulse enable of the appropriate axis is set.

Set bit “CTR_EN_AX1” (controller enable) = TRUE in “USERDB”. The controller enable is set for axis 1. By setting the bits in “USERDB” “DIR_M_AX1” (direction minus) = TRUE or “DIR_P_AX1” (direction plus) = TRUE (depending on the entry in the “Reference point approach direction” parameter, you can move axis 1 in the negative or positive direction until the reference point is found. The axis is stopped when synchronization is activated. The traversing movement is indicated by bits “GO_M_AX1”=TRUE (travel minus for axis 1) or “GO_P_AX1”=TRUE (travel plus for axis 1).

Extract from “USERDB” (variables used for examples 1 and 2)

Name	Type	Initial value	Comments
ACT_TEST	BOOL	FALSE	Enable test mode from user program
CTR_EN_AX1	BOOL	FALSE	Controller enable for axis 1
DIR_M_AX1	BOOL	FALSE	Travel command, minus direction for axis 1
DIR_P_AX1	BOOL	FALSE	Travel command, plus direction for axis 1
GO_M_AX1	BOOL	FALSE	Axis 1 traversing movement, minus direction
GO_P_AX1	BOOL	FALSE	Axis 1 traversing movement, plus direction

6.8.2 Example 2, axes moving in “Jog” mode

In the SIMATIC Manager, select **File > Open... > Projects** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Quellen\EXAMPLE2”.

The signals are stored in “USERDB”.

“Jog” mode is preset. Pulse enable of the appropriate axis is set. The override is 100 % (default value according to machine data in parameterization tool). Set bit “CTR_EN_AX1” (controller enable) = TRUE in “USERDB”. The controller enable is set for axis 1. By setting bits “DIR_M_AX1” (direction minus) = TRUE or “DIR_P_AX1” (direction plus) = TRUE in “USERDB”, you can move axis 1 in the negative or positive direction. The traversing movement is indicated by bits “GO_M_AX1”=TRUE (travel minus for axis 1) or “GO_P_AX1”=TRUE (travel plus for axis 1).

For extract from “USERDB” (variables used) see Example 1

6.8.3 Example 3, starting an axis with axis control from CPU

In the SIMATIC Manager, select **File > Open... > Projects** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Quellen\EXAMPLE3”.

In this example, axis 3 is to be positioned under the control of the CPU.

The signals are stored in “USERDB” in structure “EX3”.

Pulse enable of the appropriate axis is set.

Set bit “CTR_EN_AX3” (controller enable) = TRUE in “USERDB”. The controller enable is set for axis 3. The values for “POS” and “FRATE” are initialized with default values (position 300 mm with feed 500 mm/min). You can change the values. You can set the following bits:

- “INCR” (traversing path incremental) is set by default
- “DC” (traverse across shortest path) is only set for rotary axes
- “INCH” (traversing dimensions in inches) is not set in the example; traversing dimensions are in mm

see Section 6.5.1, Axis control from the CPU

The axis is started with the initialized values when bit “START” = TRUE is set. Bit “ACTIV”=TRUE indicates that the axis is being positioned. When the position has been reached (“INPOS” = TRUE), if no errors have occurred, the positioning operation is terminated by resetting control signal “Start” = FALSE. If an error occurs during the positioning operation (“ERR” = TRUE), this error is stored temporarily in byte “ERR_NO” for evaluation (see Table 6-14).

Extract from “USERDB” (variables used for example 3)

Name	Type	Initial value	Comments
EX3	STRUCT		Signals for example 3
POS	REAL	3.000000e+002	Position of CPU axis 3
FRATE	REAL	5.000000e+002	Feedrate of CPU axis 3
ERR_NO	BYTE	B#16#0	Error number of CPU axis 3
CTR_EN_AX3	BOOL	FALSE	Controller enable for axis 3
START	BOOL	FALSE	Start CPU axis 3
INCR	BOOL	FALSE	Traverse incremental
DC	BOOL	FALSE	Traverse shortest path
INCH	BOOL	FALSE	Traversing dimension inches
ERR	BOOL	FALSE	CPU axis 3 error
INPOS	BOOL	FALSE	CPU axis 3 in position
ACTIV	BOOL	FALSE	CPU axis 3 active
HBIT	BOOL	FALSE	Edge flag
	END_STRUCT		

6.8.4 Example 4, “Automatic mode” with program selection (with FB 4)

In the SIMATIC Manager, select **File > Open... > Projects** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Quellen\EXAMPLE4”.

The example is programmed for one channel and four axes.

The signals are stored in “USERDB” in structure “EX4”.

The program to be selected in the example is named “PROG.MPF”. The program must be available on the FM in order to be successfully selected.

“Automatic” mode is preset. Pulse enable is set for all four axes.

Set bit “CTR_EN_AAX” (controller enable) = TRUE in “USERDB”. The controller enable is set for all four axes. The NC program is selected and started when bit “START” (positive edge) is set. The axes must already be synchronized.

The following are set internally:

- Parameter “REQ” in FB 4, program is selected
- “Start” signal of channel 1 in the user interface (user DB “FMx”, DBX108.1)

If bit “CH1_RESET” is set, a reset is triggered and the program is aborted.

If the program selection is terminated with an error, “ERR” = TRUE indicates this condition and error word “STATE” must be evaluated (see Table 6-12). The program is not started. Bit “ACKN_ERR” must be set for the error acknowledgement.

Extract from “USERDB” (variables used for example 4)

Name	Type	Initial value	Comments
EX4	STRUCT		Signals for example 4
PATH	STRING[32]	'/_N_MPF_DIR'	Main program path
P_NAME	STRING[32]	'_N_PROG_MPF'	Program name PROG
STATE	WORD	W#16#0	Error status if ERR=TRUE
CTR_EN_AAX	BOOL	FALSE	Set controller enable for all 4 axes
START	BOOL	FALSE	Start NC program
CH1_RESET	BOOL	FALSE	Channel 1 – Reset
ACKN_ERR	BOOL	FALSE	Error acknowledgement
FB4_REQ	BOOL	FALSE	Start FB 4
DONE	BOOL	FALSE	Program selection finished signal
ERROR	BOOL	FALSE	Error on program selection
HBIT	BOOL	FALSE	Edge flag
	END_STRUCT		

6.8.5 Example 5, reading and writing FM variables (with FB 2 and FB 3)

In the SIMATIC Manager, select **File > Open... > Projects** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Sources\EXAMPLE5”.

The example is programmed for one channel and four axes.

The signals are stored in “USERDB” in structure “EX5”.

Write variable:

By setting bit “VAR_WR” (positive edge) you can write R parameters (R 0, R 1, R 2) to the FM.

If the write operation is terminated with an error, this is indicated by “ERR_WR” = TRUE and error word “STATE_WR” must be evaluated in “USERDB” (see Table 6-11). Bit “ACKN_ERR_WR” must be set for the error acknowledgement.

If the write operation was completed without an error, the “REQ” parameter must be reset. The values of the variables to be written are stored in “USERDB” in the structure “EX5” (symbolic name).

Read variable:

By setting bit “VAR_RD” (positive edge) you can read the following FM variables:

- Position setpoint (channel 1, axis 3)
- Velocity setpoint for positioning axis (K1, A1)
- R parameter (R 1)
- Path override

If the read operation was completed without an error, the “REQ” parameter is reset in FB2. The values of the variables read are stored in “USERDB” in the structure “EX5” (symbolic name).

If the read operation is terminated with an error, this is indicated by “ERR_RD” = TRUE and error word “STATE_RD” must be evaluated in “USERDB” (see Table 6-9). Bit “ACKN_ERR_RD” must be set for the error acknowledgement.

Extract from "USERDB" (variables used for example 5)

Name	Type	Initial value	Comments
EX5	STRUCT		Signals for example 5
VAL_RPARAM_0	REAL	1.000000e+000	Value for R parameter 0
VAL_RPARAM_1	REAL	2.000000e+000	Value for R parameter 1
VAL_RPARAM_2	REAL	3.000000e+000	Value for R parameter 2
SET_POSITION	REAL	0.000000e+000	Programmed position setpoint
FDRATE	REAL	0.000000e+000	Velocity setpoint
R_PARAM_1	REAL	0.000000e+000	Value of R parameter (R1)
OVERRIDE	REAL	0.000000e+000	Path override
STATE_RD	WORD	W#16#0	Error number if ERR_RD=TRUE
STATE_WR	WORD	W#16#0	Error number if ERR_WR=TRUE
VAR_WR	BOOL	FALSE	Start variable write operation
VAR_RD	BOOL	FALSE	Start variable read operation
FB2_REQ	BOOL	FALSE	Internal start call of FB 2
FB3_REQ	BOOL	FALSE	Internal start call of FB 3
ACKN_ERR_RD	BOOL	FALSE	Error acknowledgement for read variable
ACKN_ERR_WR	BOOL	FALSE	Error acknowledgement for write variable
NDR	BOOL	FALSE	Done signal FB 2
DONE	BOOL	FALSE	Done signal FB 3
ERR_RD	BOOL	FALSE	Error bit for read variable
ERR_WR	BOOL	FALSE	Error bit for write variable
HBIT1	BOOL	FALSE	Edge flag
HBIT2	BOOL	FALSE	Edge flag
	END_STRUCT		

6.8.6 Example 6, reading and writing R parameters (with FB 2 and FB 3)

In the SIMATIC Manager, select **File > Open... > Projects** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Sources\EXAMPLE6”.

The example reads and writes 24 R parameters.

The signals are stored in “USERDB” in structure “EX6”. The R parameters to be read or written are stored in structure “R_PARAM” in DB 116.

The example is programmed for one channel and four axes.

In the first section, when bit “VAR_RD” 24 is set, R parameters (R0 to R23) are read from the FM in blocks of 8. They are stored in the R_PARAM structure in DB 116.

In the second section, when bit “VAR_WR” is set, 24 R parameters (R0 to R23) are written to the FM in blocks of 8. The contents of the data are read from structure “R_PARAM” in DB 116. Request bits “VAR_RD” and “VAR_WR” must be set with a positive edge. If the read or write operation was completed without an error, the call parameters are reset.

If an error occurs, the error code is stored in the variable “STATE_RD” (error code for read) or “STATE_WR” (error code for write); (see Table 6-9 or 6-11).

Notice

By changing the “CONT_CYCL” variable (default: 3) in “USERDB”, the number of variables can be increased or decreased in increments of 8 (e.g. 5 means 40 R parameters, R0 to R39). Please note that, if the number of R parameters is increased, the size of structure “R_PARAM” in DB 116 must also be increased.

Extract from "USERDB" (variables used for example 6)

Name	Type	Initial value	Comments
EX6	STRUCT		Signals for example 6
COUNT_FB2	INT	0	Call counter for FB 2
STATE_RD	WORD	W#16#0	Error number if ERR=TRUE
LINE1	ARRAY[1..8] WORD		
SOURCE_ADR1	DWORD	DW#16#0	Source address 1
TARGET_ADR1	DWORD	DW#16#0	Target address 1
RD	ARRAY[1..8] REAL		
COUNT_FB3	INT	0	Counter
STATE_WR	WORD	W#16#0	Error number if ERR=TRUE
LINE2	ARRAY[1..8] WORD		
SD	ARRAY[1..8] REAL		
SOURCE_ADR2	DWORD	DW#16#0	Source address 2
TARGET_ADR2	DWORD	DW#16#0	Target address 2
CONT_CYCL	BYTE	B#16#3	Factor for the number of variables (3 x 8)
VAR_WR	BOOL	FALSE	Start variable write operation
VAR_RD	BOOL	FALSE	Start variable read operation
FB2_REQ	BOOL	FALSE	Internal start call of FB 2
FB3_REQ	BOOL	FALSE	Internal start call of FB 3
ACKN_ERR_RD	BOOL	FALSE	Error acknowledgement for read variable
ACKN_ERR_WR	BOOL	FALSE	Error acknowledgement for write variable
NDR	BOOL	FALSE	Done signal FB 2
DONE	BOOL	FALSE	Done signal FB 3
ERR_RD	BOOL	FALSE	Error bit for read variable
ERR_WR	BOOL	FALSE	Error bit for write variable
HBIT1	BOOL	FALSE	Edge flag
HBIT2	BOOL	FALSE	Edge flag
	END_STRUCT		

6.8.7 Example 7, error diagnosis

In the SIMATIC Manager, select **File > Open... > Projects** and open the example project “zEn16_01_FM357-2_BF_EX\EXAMPLES\Sources\EXAMPLE7”.

The example is used for error diagnostics. The example is programmed for two channels and four axes.

The program was developed with reference to Figure 6-2, page 6-12.

The signals are stored in “USERDB” in structure “EX7”.

Part of the diagnostics is presented in the first network. The user can program his own diagnostic routines in the second network; the third network is used for error acknowledgement.

Important signal of example 7 in “USERDB”:

PRG_INIT:	User program initialization
EMERGENCY:	Open EMERGENCY STOP circuit (evaluate error number “ERR_NO”)
ERR_NO_CPUAXx:	CPU axis error number (evaluate only if ERR_CPUAXx=TRUE)
ERR_NO:	Error number (evaluate if FM_ERR=TRUE see Section 12.2)
ERROR:	Group error bit for various error messages from the application interface (user DB “FMx”) <ul style="list-style-type: none"> – FM_ERR=TRUE (error in FM) – ERR=TRUE (error in corresponding channel) – CH_RDY=FALSE (channel ready)
CHx_RESET:	Reset signal of corresponding channel

The error number of the FM357-2 is read out automatically in the example if group error bit “ERROR”=TRUE.

The “ERROR” bit is reset (ERROR=FALSE) if

- No FM error is active (FM_ERR=FALSE)
- No error is active in both channels (ERR=FALSE)
- The “channel ready” signal of the corresponding channel is set (CH_RDY=TRUE)

The error in the corresponding channel is acknowledged in the third network by means of “CH1_RESET” or “CH2_RESET” (further optional error acknowledgement is possible using the “CANCEL” function with FB 4).

Extract from "USERDB" (variables used for example 7)

Name	Type	Initial value	Comments
EX7	STRUCT		Signals for example 7
ERR_NO	DINT	L#0	FM error number
STATE_RD	WORD	W#16#0	Error number of FB 2
ERR_NO_CPUAX1	BYTE	B#16#0	Error number of CPU axis 1
ERR_NO_CPUAX2	BYTE	B#16#0	Error number of CPU axis 2
ERR_NO_CPUAX3	BYTE	B#16#0	Error number of CPU axis 3
ERR_NO_CPUAX4	BYTE	B#16#0	Error number of CPU axis 4
PRG_INIT	BOOL	FALSE	User program initialization
EMERGENCY	BOOL	FALSE	User EMERGENCY STOP circuit
ERR_CPUAX1	BOOL	FALSE	Error on CPU axis 1
ERR_CPUAX2	BOOL	FALSE	Error on CPU axis 2
ERR_CPUAX3	BOOL	FALSE	Error on CPU axis 3
ERR_CPUAX4	BOOL	FALSE	Error on CPU axis 4
ERROR	BOOL	FALSE	Group error bit (FM_ERR/ERR/ CH_RDY)
CH1_RESET	BOOL	FALSE	Reset for channel 1
CH2_RESET	BOOL	FALSE	Reset for channel 2
ERRNO_RD	BOOL	FALSE	Read out "FM error number" re- quest
ACKN_ERR_RD	BOOL	FALSE	Error acknowledgement of FB 2
ERR_RD	BOOL	FALSE	Error in FB 2
FB2_REQ	BOOL	FALSE	Request FB 2
NDR	BOOL	FALSE	Done signal of FB 2
HBIT	BOOL	FALSE	Edge flag
	END_STRUCT		

6.9 Technical data, runtime specifications

Memory allocation

The memory requirements of the blocks are specified for one FM 357-2.

If several FMs (max. 3) are used, the memory required is increased by the amount used by the user DBs.

Table 6-21 Memory requirements of blocks

Function Block	Function	Comment	Load/work memory in bytes
User DB "FMx"	Interface (FM signals)		2604 / 962
User DB "AXy"	Interface (axis signals)		1058 / 500
FC 1	Startup/initialization	No. can be changed!	530 / 364
FC 5	Diagnostic alarms	No. can be changed!	344 / 210
FC 22	Basic functions and operating modes	No. can be changed!	5002 / 4348
FB 2 FB 3 FB 4	Read variable Read variable General services	No. can be changed!	572 / 340 576 / 344 466 / 290
DB 15	Internal	No. cannot be changed!	598 / 190
DB 16	Internal	No. cannot be changed!	142 / 30
DB 121	Extract from FM variable list	No. can be changed!	810 / 190
FB 6	Internal	No. cannot be changed!	5706 / 4882
FC 23	Internal	No. cannot be changed!	1552 / 1262
			Total: 19960/13912

Timers

Timers 0, 1 are assigned or reserved internally for standard function blocks.

Notice

In order to minimize the communication runtime in the FC 22/OB 1 cycle, particularly in distributed configurations, the "Actual value" and the "Distance to go" should only be read cyclically if it is needed for the user function.)

Example of transfer times of the interface between CPU 315DP ↔ FM 357-2 at a CPU cycle time of approx. 20 ms

Table 6-22 Transfer times of the interface between CPU and FM 357-2

Function Block	Request	Centralized installations
FC 22	Block time (4 axes, single-channel)	6.0...6.2 ms ± 6.1 ms
	Block time, auxiliary function transfer (single-channel) plus actual values cyclically	7.08...7.44 ms ± 7.23 ms
	Block time with write data trigger (4 axes)	8.0...8.9 ms ± 8.5 ms
	Block time with read data trigger (4 axes)	8.2...8.6 ms ± 8.4 ms
	Block time (4 axes, multi-channel)	8.4...9.0 ms ± 8.6 ms
FB 2	Read data from FM	70.4...80.4 ms ± 76.7 ms
FB 3	Write data from FM	79.2...88.4 ms ± 81.4 ms
FB 4	General services	79.2...89.6 ms ± 84.7 ms

Processing time on FM 357-2: < 2 IPO cycles

In the distributed mode, the block runtimes can be up to 5 times longer due to the transfer rates.

In the case of a small user program, the FC 22 ensures signal exchange within an IPO cycle (block time prolongation of up to approx. 9 ms = IPO cycle).



Start-Up

7

Section overview

Section	Title	Page
7.1	Installation and wiring	7-2
7.2	FM 357-2 power-up	7-3
7.3	Procedure for parameterization	7-5
7.4	Testing and optimization	7-7

General

This chapter contains checklists for starting up the positioning module. The checklists will help you:

- Check all steps until the module is running.
- Prevent malfunctions of the module once it is in operation.

You are guided through the start-up of the machine axes.

7.1 Installation and wiring

Information about installation

For information about installing the module, please refer to

- Chapter 3 of this manual
- The installation manual *S7-400/M7-400 Programmable Controller, Hardware and Installation*

Installing firmware/firmware update

For information about installing or updating the firmware, please refer to Section 3.2 of this manual.

Information about wiring

For information about wiring up the module, please refer to

- Chapter 4 of this manual
- The installation manual *S7-400/M7-400 Programmable Controller, Hardware and Installation*

Checklist

The following checklist will help you to keep a check of the most important steps in installing and parameterizing the FM 357-2 multi-axis module.

Table 7-1 Installation and wiring checklist

Step	Check	What to do:	Ok ✓
1	Slots	Plug the module into one of the suitable slots.	
2	Shielding	Check the shielding of the FM 357-2 multi-axis module! <ul style="list-style-type: none"> • To ensure proper shielding, the module must be screwed down firmly on the rail. • The shielding for shielded lines for digital I/O modules must be connected to the shielding terminal element. • The shielding for the setpoint cable should not be grounded on the drive-unit end. 	
3	Limit switches	Check the start/stop hardware limit switches. The limit-switch connections must be connected to the power section. It is not permitted to connect the start/stop hardware limit switches to the digital inputs.	
4	Parameterize	Make sure the FM 357-2 multi-axis module setup is consistent with the parameterization. Check in particular that: <ul style="list-style-type: none"> • the attached encoder and drives match the machine data. • the wiring of the digital I/O modules matches the machine data. 	

7.2 Powering up the FM 357-2

Important operating and display elements for power-up

The following operating and display elements are important for FM 357-2 power-up:

- Error and status LEDs
- Start-up switch of the FM 357-2 and CPU

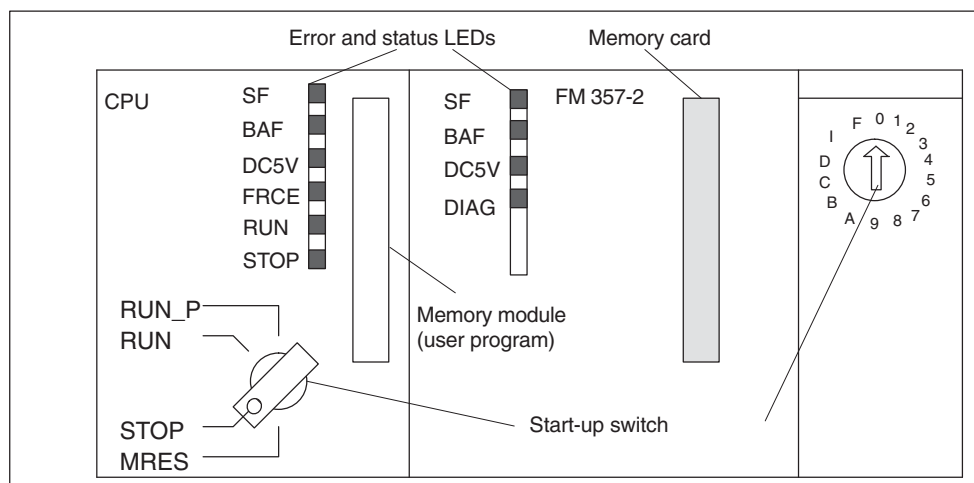


Fig. 7-1 Operating and display elements for power-up

Start-up switch

A start-up switch is installed on the FM 357 (see Figure 7-1). It is used to support start-up. You can operate this switch with a screwdriver.

The following switch settings are relevant for powering up the FM 357-2.

Table 7-2 Settings with the start-up switch of the FM 357-2

Setting	Significance
0	Normal Powerup
1	Power-up and start-up with default values
2	Reserved
3	For information about data backup on memory card, see Section 3.3
4...9	Reserved
A	Copying of the complete firmware and parameter data (if saved previously on a memory card by means of a data backup) from the memory card to the module
B...F	Reserved

License

The license on the memory card is checked against the version of the firmware loaded on the FM each time the FM 357-2 is powered up.

The following must match:

- FM 357-2 memory card license **L** with firmware **L**
- FM 357-2 memory card license **LX** with firmware **LX**
- FM 357-2 memory card license **H** with firmware **H**

Notice

If the memory card has no license or an incorrect license, this is indicated as follows:

- “SF” LED flashes
- “No license or incorrect license” error display
Display via “Parameterize FM 357-2” tool or OP 17.

You can only operate the axes in simulation mode.

Power-up times

The duration of the normal power-up (start-up switch = 0) is approximately 40 s.

The duration of the power-up and start-up with default values (start-up switch = 1) is <1 min.

Status displays (on LEDs) during power-up

Power-up status is displayed by the following indicators:

- CPU:
 - 5 V DC (green) → ON
 - RUN (green) → Flashes
 - STOP (yellow) → ON
 - Further LEDs → OFF
- End of power-up:
 - RUN (green) → ON
 - STOP (yellow) → OFF
- FM 357-2:
 - 5 V DC (green) → ON
 - Further LEDs → OFF
- End of power-up:
 - DIAG (yellow) → Flashes (sign-of-life approx. 3 Hz)

7.3 Procedure for parameterization

Information about parameterization

For information about parameterization, please refer to

- Chapters 5 and 9 of this manual
- The on-line help of “Parameterize FM 357-2”

A communication link to the CPU must be set up to parameterize the FM 357-2 online. The basic program must be running on the CPU (see Chapter 6). The CPU is in RUN mode.

Overview

The parameter data of the FM 357-2 comprise:

- Machine data → for starting up the module
- User data (R parameters, zero offsets, tool offsets, NC programs) → for startup and adaptation to the plant and functionality

The data can be processed online or offline by the parameterization tool and transferred to the module over the MPI.

After transfer to the module, the data are stored there modally.

Checklist

It is the responsibility of the user of the module to ensure that all machine data are correct. It is therefore advisable to perform startup using the following checklist.

Table 7-3 Parameterization checklist

Step	Check	What to do:	Ok ✓
1	Default values	<p>Set-up default values of machine data (first startup)</p> <p>For power-up with default values see Section 7.2 The default values of the machine data are listed in Table 5-2.</p>	
2	Configuration	<p>Definition of the system configuration</p> <p>Here, you define the following important parameters for the module:</p> <ul style="list-style-type: none"> • Internal system of measurement • Number of axes and channels • Axis type (linear or rotary axis) • Drive <p>You must define the internal system of measurement and the axis type at the beginning of the start-up procedure. In Section 9.1 you will find further information about the individual machine data.</p>	
3	Axes	<p>Basic start-up of the axes</p> <p>For basic start-up of an axis, you must define the machine data for:</p> <ul style="list-style-type: none"> • Encoder adjustment • Controller data • Velocities • Monitoring <p>(see Sections 9.2 to 9.6). After loading and activation, you can initiate the test and optimization procedure described in Section 7.4.</p>	
4	Functions	<p>Parameterizing the functions</p> <p>The following functions can be adapted to the needs of your plant using machine data:</p> <ul style="list-style-type: none"> • Reference-point approach • Initial settings • Auxiliary functions • Digital I/Os • Software cams • Motion control • Travel to fixed stop <p>You will find further information in Sections 9.7 to 9.11, 9.16, 9.18</p>	

7.4 Testing and optimization

Information about testing and optimization

After you have installed, wired up, powered up and parameterized the FM 357-2 multi-axis module, you can test and optimize it. The test and start-up user interface and the user program are used for testing and optimization. The user program is stored ready for use in S7 library FM357_2L (see Section 6).

You can also test individual modes and their NC programs, and view and debug them during execution.

You can monitor the interface between the FM and the user program.

The user program uses the “enable test” as an interlock signal to specify that the interface for controlling the FM 357-2 is allowed to use the “Start-up window”. When the user program sets signal “Enable test” = TRUE and the startup user interface sets signal “Test request” = TRUE (“Test” button activated in startup window), the FM 357-2 is controlled via the “Start-up window” (test mode).

The call is performed with the menu **Test > Startup** in the “Parameterize FM 357-2” tool.

When you call up this menu the following screen appears:

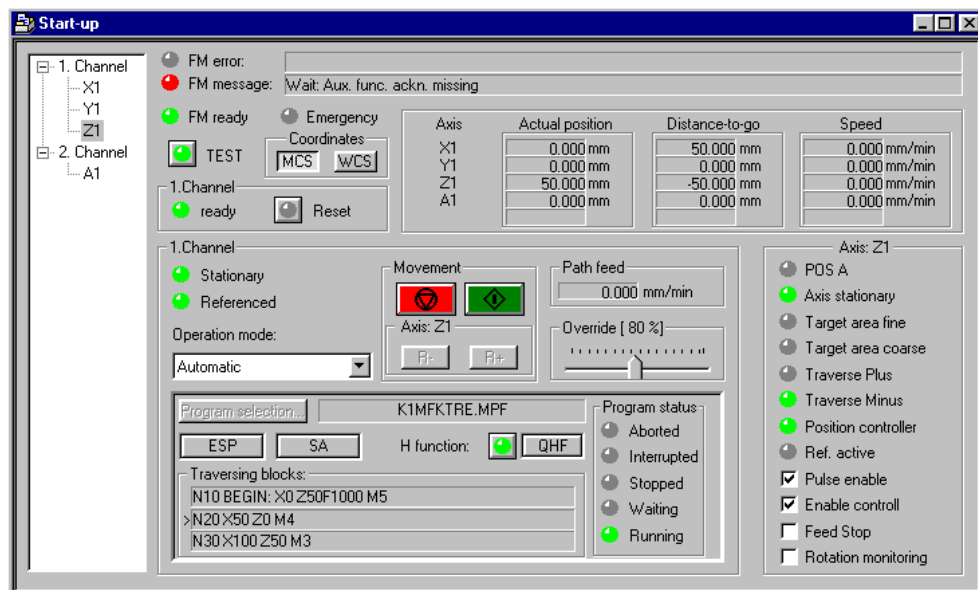


Fig. 7-2 Start-up interface (e.g. for “Automatic” mode)

You can also call up the following screens:

The following display appears when you select **Test > Troubleshooting**:

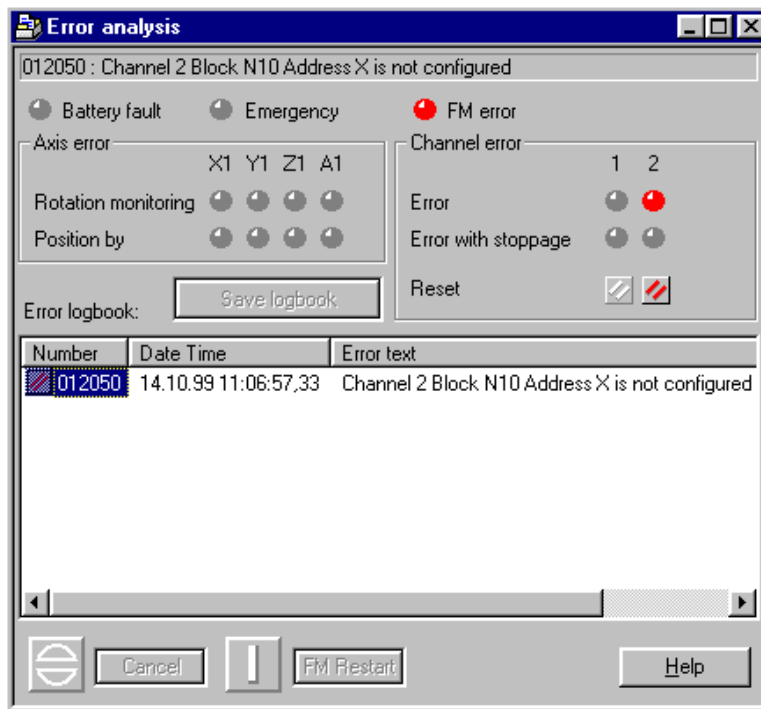


Fig. 7-3 Troubleshooting

The following display appears when you select **Test > Servicing Data**:

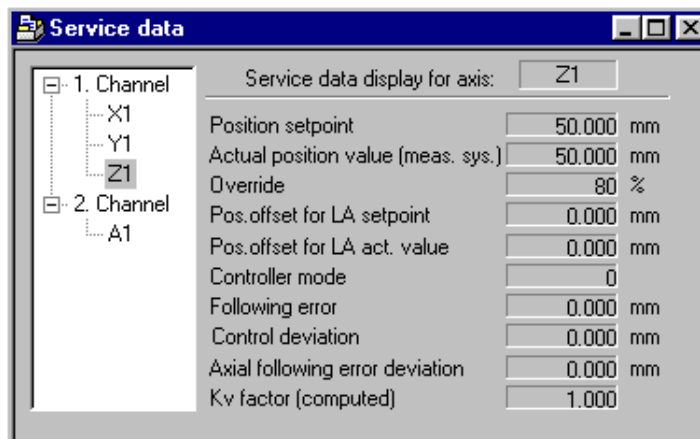


Fig. 7-4 Servicing data

If you call the menu **Test > Digital inputs**, the following screen form will appear:

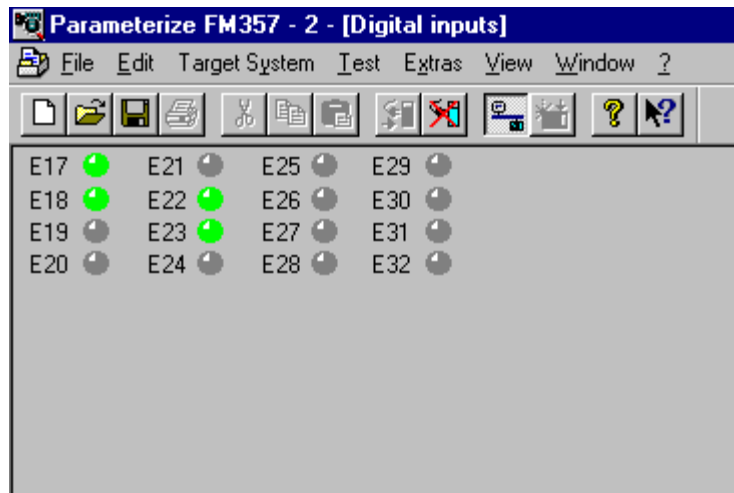


Fig. 7-5 Digital inputs

This screen form displays the status of the digital inputs at the local P bus.

If you call the menu **Test > Digital outputs**, the following screen form will appear:

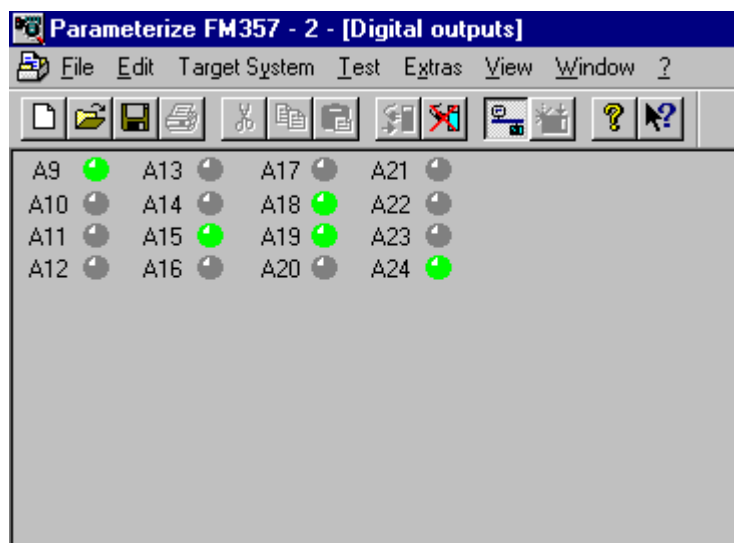


Fig. 7-6 Digital outputs

This screen form displays the status of the digital outputs at the local P bus.

If you call the menu **Test > Synchronized action**, the following screen form will appear:

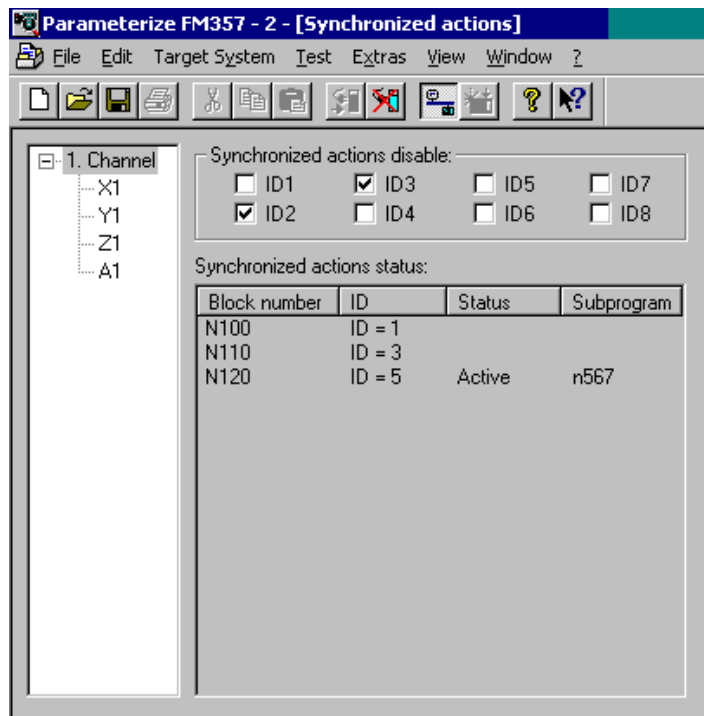


Fig. 7-7 Synchronized action

This screen form displays:

- the mapping of the signals “Disable synchronized action ID1...8” (user DB “FMx”, DBB580, 586, 592, 598); these signals can be written and read in the parameterization user interface.
- the programmed synchronized actions by blocks with block number and validity (in the case of modal/static synchronized actions, with number).
- the status of the synchronized action
 - “ “ (no data)
The condition is checked at the interpolation cycle.
 - “disabled”
For the synchronized action, a “Disable synchronized action ID1...8” (user DB “FMx”, DBB580, 586, 592, 598) has been set by the CPU.
 - active
The action is currently running. If the instruction part of a programmed synchronized action has started a subroutine, the current block number is additionally displayed in the “Subroutine” column.

For the programming of the synchronized action, see Section 10.32.

If you call the menu **Test > Software cams**, the following screen form will appear:

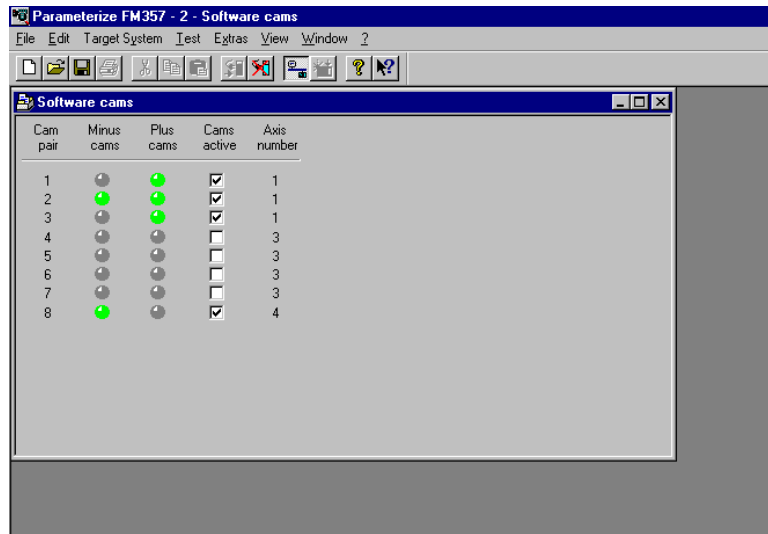


Fig. 7-8 Software cams

This screen form displays the status of the parameterized software cams.

For a description of the software cams, see section 9.1.1.

If you call the menu **Test > Performance display**, the following screen form will appear:

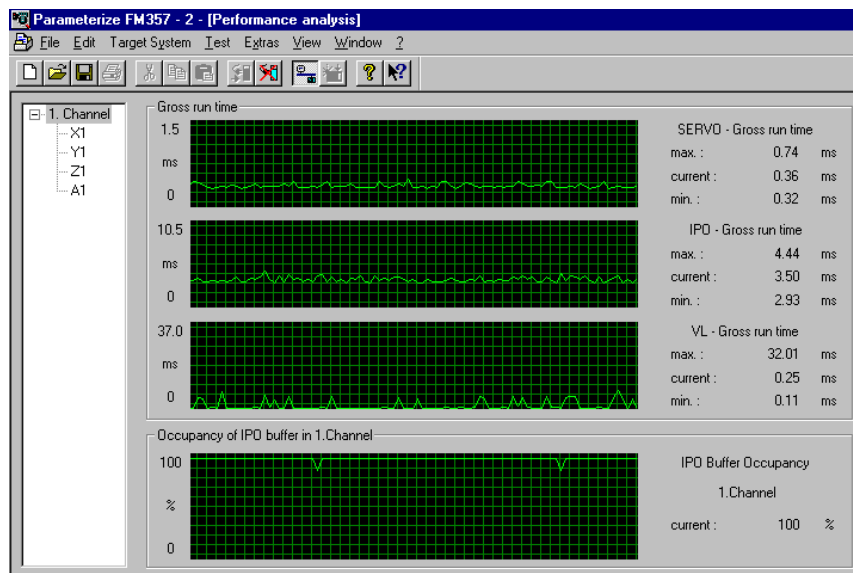


Fig. 7-9 Performance display

see Section 9.1 System clocks

The following screen appears when you select **Test > GUD Viewer**:

GUD-Variable	Bereich	Typ	Wert
VARI1	NCK	INT	148
VARR1[1]	NCK	REAL	1.000000; 2.000000
VARR1[2]	NCK	REAL	2.000000; 3.000000
VARR1[3]	NCK	REAL	3.000000; 4.000000
A_END	1. Kanal	REAL	0.000000
A_MID	1. Kanal	REAL	0.000000
A_START	1. Kanal	REAL	0.000000
VARCI1	1. Kanal	INT	1
VARZ	1. Kanal	REAL	333.110000
X_END	1. Kanal	REAL	0.000000
X_MID	1. Kanal	REAL	0.000000
X_START	1. Kanal	REAL	0.000000
Y_END	1. Kanal	REAL	0.000000
Y_MID	1. Kanal	REAL	0.000000
Y_START	1. Kanal	REAL	0.000000
Z_END	1. Kanal	REAL	0.000000
Z_MID	1. Kanal	REAL	0.000000
Z_START	1. Kanal	REAL	0.000000
A_END	2. Kanal	REAL	0.000000
A_MID	2. Kanal	REAL	0.000000
A_START	2. Kanal	REAL	0.000000
VARCI1	2. Kanal	INT	1
VARZ	2. Kanal	REAL	3.330000
X_END	2. Kanal	REAL	0.000000
X_MID	2. Kanal	REAL	0.000000

Fig. 7-10 GUD-Viewer

The following screen appears when you select the menu **Test > LUD Viewer**:

LUD-Variable	Typ	Wert
BIT_ARRAY[1]	BOOL	TRUE; FALSE; FALSE; FALSE
BIT_ARRAY[2]	BOOL	TRUE; FALSE; FALSE; FALSE
BIT_ARRAY[3]	BOOL	FALSE; FALSE; FALSE; FALSE
BIT_ARRAY[4]	BOOL	FALSE; FALSE; FALSE; FALSE
BIT_ARRAY[5]	BOOL	FALSE; FALSE; FALSE; FALSE
ENDE	INT	11
LVAR1	INT	0
LVAR2	INT	0
LVAR3	INT	0
LVAR4	INT	0
LVAR5	INT	0
POSIT1	REAL	0.000000
START_101	INT	0
TAB_REAL[1]	REAL	1.000000; 2.000000; 3.000000; 4.000000
TAB_REAL[2]	REAL	2.000000; 3.000000; 4.000000; 5.000000
ZEICHEN	CHAR	67 = 'C'
Z_KETTE	STRING	abcdef

Fig. 7-11 LUD Viewer

Notice

Variables of the AXIS type are not displayed in the GUD/LUD Viewer.

To optimize your drive, you can use the “Trace” function.

Choose **Test > Trace** from the menu; the following screen will appear:

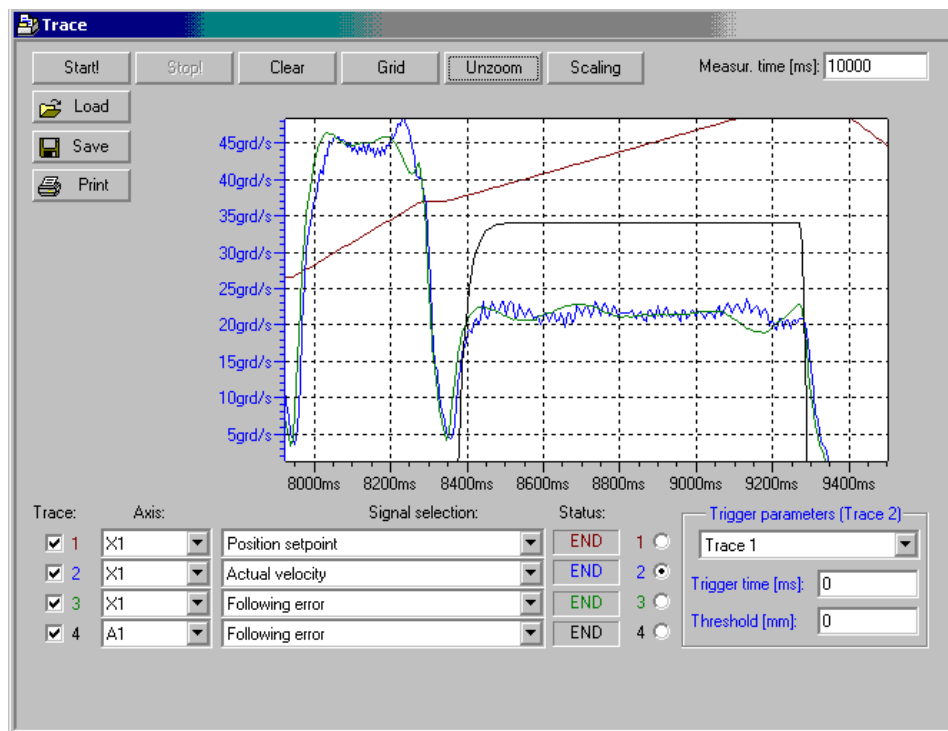


Fig. 7-12 Trace

You have the possibility of recording up to four signal curves in this screen. You can save, load and print these curves.

The selected trace values are determined and saved in each servo trace. At the end of tracing, the data are exported by the parameterization tool for evaluation and display.

The following values / signals are offered for selection:

- Following error
 - ... is the difference between position setpoint and position actual value. Unit: mm, inches or degrees
- Control deviation
 - ... is the difference between position setpoint at the position controller input and the position actual value. Unit: mm, inches or degrees
- Contour deviation
 - ... is the difference between a position actual value calculated in advance from a position setpoint (internal model) and the real position actual value provided by the measuring system. Unit: mm, inches or degrees

- Position actual value provided by the measuring system
... real position actual value of the axis; provided directly from the measuring system.
Unit: mm, inches or degrees
- Position setpoint to the drive
... position setpoint issued by the position control to the drive.
Unit: mm, inches or degrees
- Position setpoint at the controller input
... position setpoint issued by the interpolator to the position control.
Unit: mm, inches or degrees
- Velocity setpoint at the controller input
... velocity setpoint issued by the interpolator to the position control.
Unit: mm/min, in/min or deg/min
- Acceleration setpoint at the controller input
... acceleration setpoint issued by the interpolator to the position control. Unit: mm/s², in/s² or deg/s²
Note:
Up to and including software release 4, this value has only been read in the first servo cycle. A continuous motional sequence has therefore been represented as individual peaks.
- Velocity actual value provided by the measuring system
... real velocity actual value of the axis; provided directly from the measuring system.
Unit: mm/min, in/min or deg/min
- Signal "Interpolation completed"
The interpolator delivers no more setpoints to the position control.
- Signal "Target range fine"
The axis has reached the target range, fine.
- Signal "Target range, coarse"
The axis has reached the target range, coarse.
- Position actual value, summed up
... real position actual value of the axis, provided directly from the measuring system, summed up from the moment when the tracing was started.
Unit: mm, inches or degrees
- Position setpoint summed up
... position setpoint issued by the position control to the drive, summed up from the moment when the tracing was started.
Unit: mm, inches or degrees

Example: Tracing of the following error

To call the Trace window, open the Start-up window via **Test > Start-up**.

Click on the **Start** button in the Trace screen. Switch to the Start-up screen. Select the **R+** button and press the **spacebar**.

After a measuring time of 10 s, the curve is traced and will be displayed.

A trigger parameter can be set for each signal curve:

- No trigger

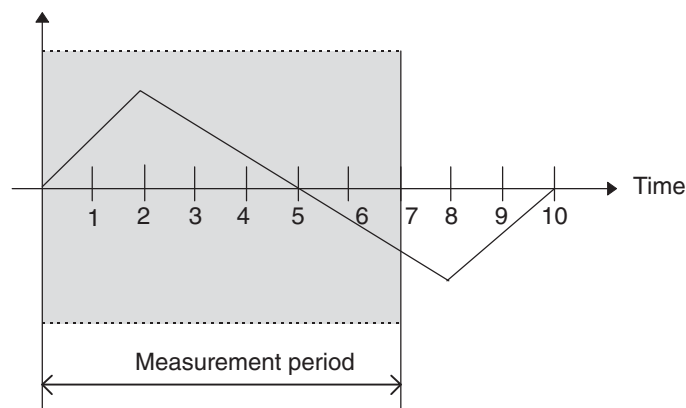
The recording procedure runs until the “Stop” button is pressed. For example, for a measurement period of 10 s, the curve is plotted for the last 10 s before the “Stop” button is pressed.

- Record immediately

Start recording immediately the request is received (on Start). It is not possible to record a preprocessing operation.

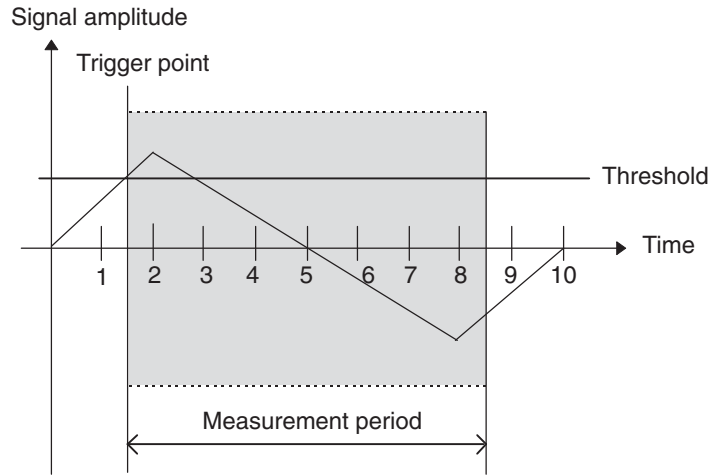
If several signal curves are started simultaneously, a time shift occurs (signals can only be started in succession).

Signal amplitude



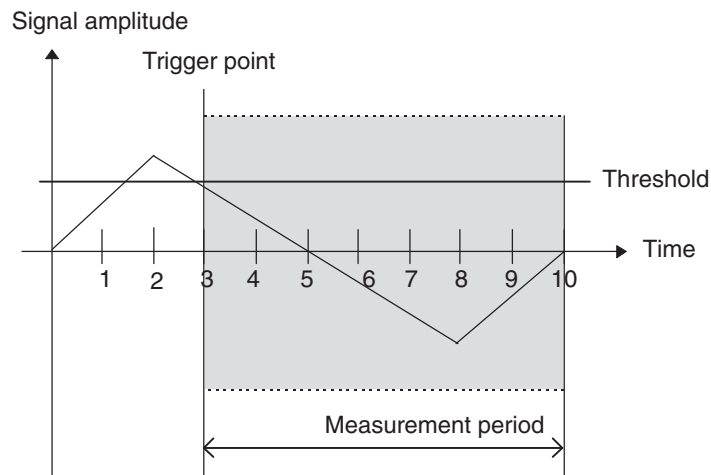
- Positive edge

After the “Start” , the FM monitors the defined threshold. If the last value was less than the threshold and the current value is greater than or equal to the threshold, the trigger condition is “true”.



- Negative edge

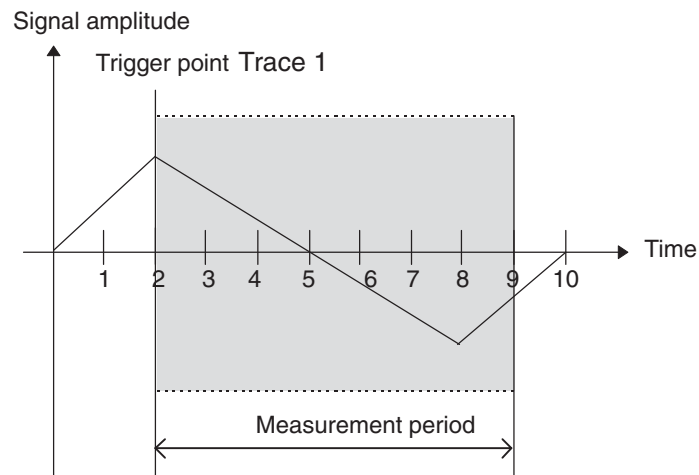
After the “Start” , the FM monitors the defined threshold. If the last value was greater than or equal to the threshold and the current value is less than the threshold, the trigger condition is “true”.



- Trace 1

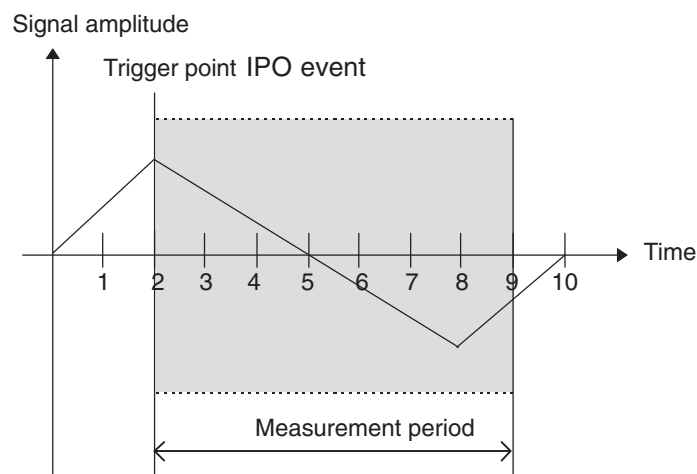
After the “Start”, the FM monitors the status of T trace 1. As soon as the internal status of Trace 1 shows the “gettriggered” event, the trigger condition for the configured trace is “true”.

To synchronize several signal curves simultaneously, the curves can be triggered in response to signal curve 1 (Trace 1).



- IPO event (see Table 10-7 or 10-11)

After the “Start”, the FM monitors the status of the system variable. As soon as the system variable switches from 0 to 1, the trigger condition for the configured trace is “true”.



Enable signals for axes

Before an axis can be traversed from the control system, signals must be supplied at enable terminals on the drive and enable bits set on the interface.

Enables on the drive

The drive connection on the FM 357-2 is implemented over the drive interface (X2). In addition to the analog setpoints and the clock and direction pulses, the "Controller enable" and "Enable signal" (for stepper motor) are output over this interface.

Enables over the CPU interface

The following signal must be routed over the CPU interface for the axis:

Controller enable (user DB "AXy", DBX2.1+m)

The following signals on the interface should **not** be enabled, since these cause the movement to be disabled:

Override (user DB "AXy", DBB0+m) not at 0 %

Delete distance to go (user DB "FMx", DBX107.2+n)

Feed stop (user DB "AXy", DBX4.3+m)

Limit switches

Setting of hardware limit switches and signal check:

- Hardware limit switch plus (user DB "AXy", DBX7.1+m)
- Hardware limit switch minus (user DB "AXy", DBX7.0+m)

Test sequence

To test the axis, please follow the sequence specified in the flowchart overleaf:

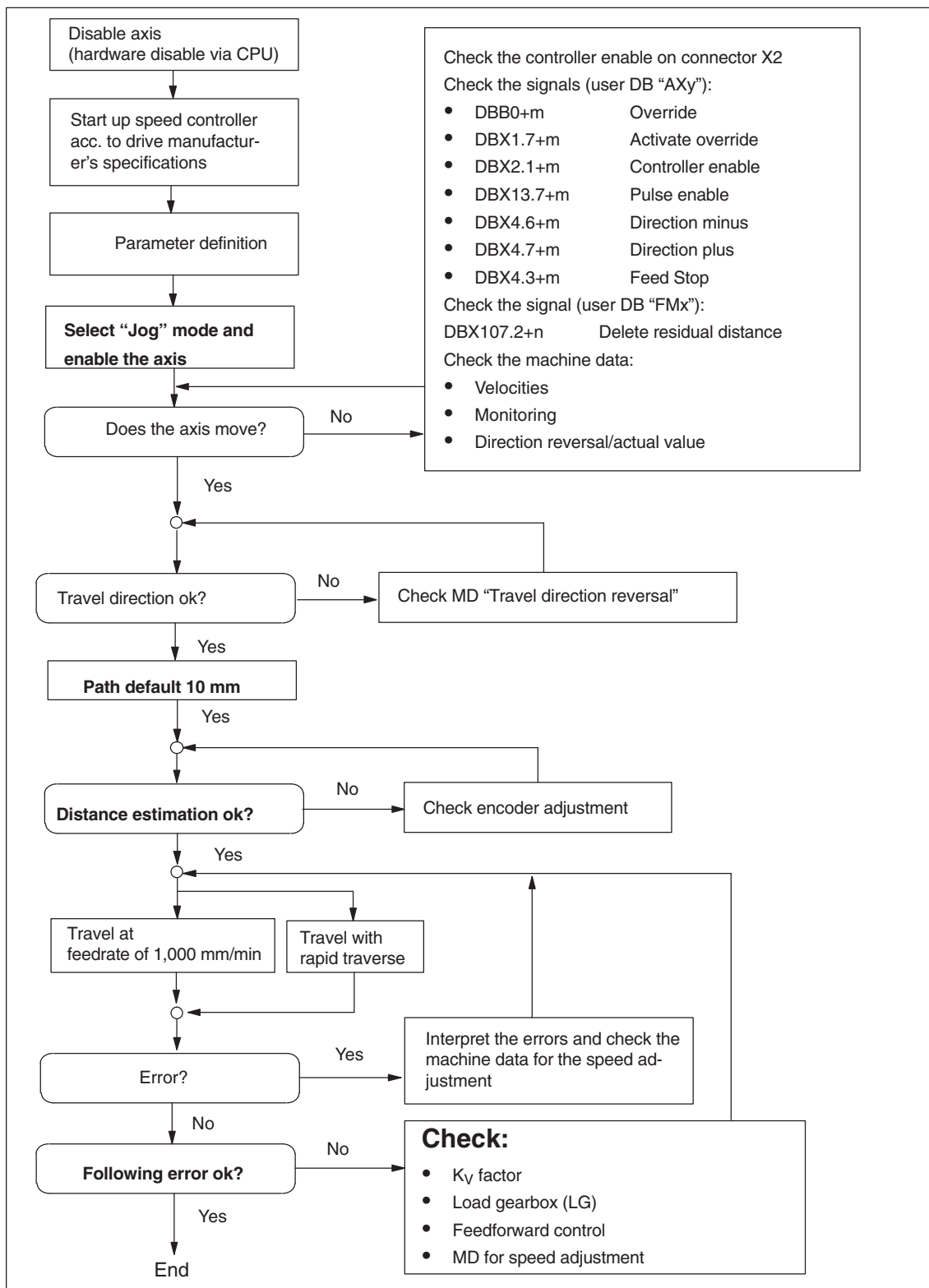


Fig. 7-13 Testing the axis



Human Machine Interface

Section overview

Section	Title	Page
8.1	HMI sample interface for the OP 17	8-3
8.2	Troubleshooting on the OP 17 (configuration example)	8-8
8.3	HMI sample interface for SIMATIC HMI devices	8-10

Overview

This chapter provides you with an overview of the operator control and monitoring features of the FM 357-2.

An operator panel can be connected to the CPU via the MPI interface for operator control and monitoring of the FM 357-2 (see Figure 1-3). The functions of the operator panel (e.g. OP 17) must be configured using the “ProTool” software.

The module uses the SIMATIC interface (backplane bus) to communicate with the control panel.

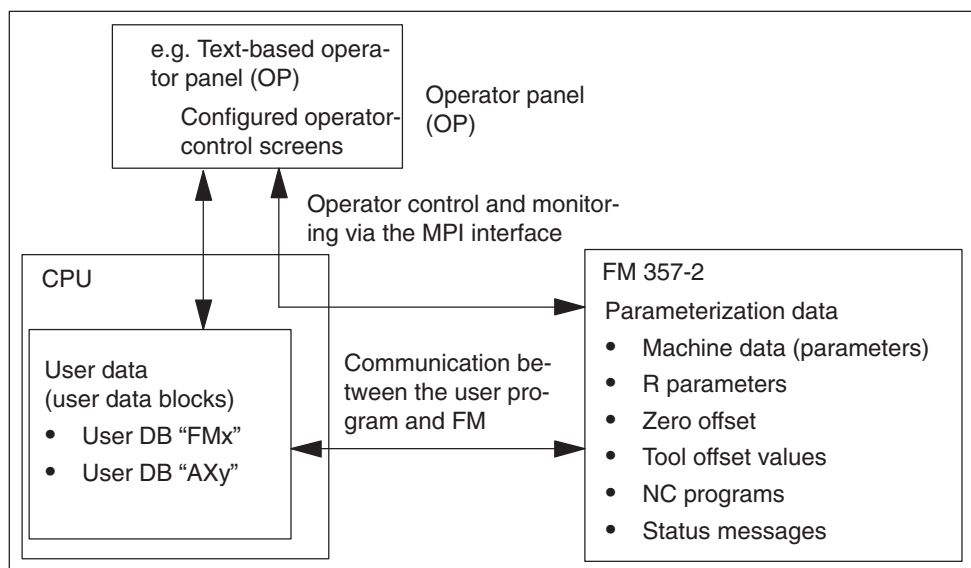


Fig. 8-1 Operator control and monitoring of the FM 357-2

The configuration package contains preconfigured user interfaces for SIMATIC HMI devices (OP 17/27, TP 170 B, MP 270 B).

What can be controlled on the FM 357-2?

The application interface of the CPU (user DBs) can be used to control the FM 357-2 by changing or setting the data/signals:

- User DB “FMx”
 - e.g.
 - Axis selection
 - Start, Stop
 - Operating modes
- User DB “AXy”
 - e.g. Override

Functions/data can be activated/controlled directly on the FM 357-2:

- NC program selection (OP 17 only)
- Tool offset input (OP 17 only)
- R parameters

What can be monitored on the FM 357-2?

The following data/signals can be displayed on the operator panel screen:

- The following data/signals are read directly on the FM:
 - Machine data
 - Error numbers
 - R parameters
 - Zero offsets (OP 17 only)
 - Tool offset data (OP 17 only)
 - NC programs/NC program overview (OP 17 only)
 - Actual NC blocks
 - Actual values
 - Special FM variables (e.g. override)
- Checkback signals and error messages are read from the CPU.

8.1 HMI sample interface for the OP 17

Overview

This chapter described the preconfigured user interfaces (single and multi-channel) which you have to modify in order to meet the needs of your project (e.g. FM addresses, user DB no., axis/channel assignment, number of channels).

The “ProTool/Lite or ProTool” software is used for this purpose. Using this tool, you can change the preconfigured displays, insert new displays or delete existing displays.

The user interface addresses:

- User DBs “FMx” (DB 21) and “AXy” (DB 22) in the CPU (controller: control_CPU; address = 2; slot = 0)
- Variables of the FM 357-2 (controller: control_357; address 3; slot 0).

The OP 17 was assigned an MPI address of 10 in this sample configuration.

You can print out the complete configuration, e.g. using “ProTool/Lite”. This provides you with detailed screen descriptions.

You will find the preconfigured user interface in the following directories:

- **1 channel with 4 axes:** [STEP7 directory]\EXAMPLES\FM357-2\zEn16_01_fm357-2_op_ex\op17_ch1fm357.pdb
- **2 channels with 2 axes each:** [STEP7 directory]\EXAMPLES\FM357-2\zEn16_01_fm357-2_op_ex\op17_ch2fm357.pdb

Monitoring

The data for monitoring can be read and displayed directly from the FM 357-2 or from the CPU via the user DBs.

Operator control

The data and signals (e.g. bit memories and values) are written to the user DBs of the CPU or directly to the FM.

User program

You can use the OP to set bit memories, for example, which the user program evaluates (e.g. select “Automatic” mode on the FM 357-2).

OP 17 operator interface

The figure below shows you an overview of the user interface (menu tree) of the sample OP 17 configuration for the FM 357-2.

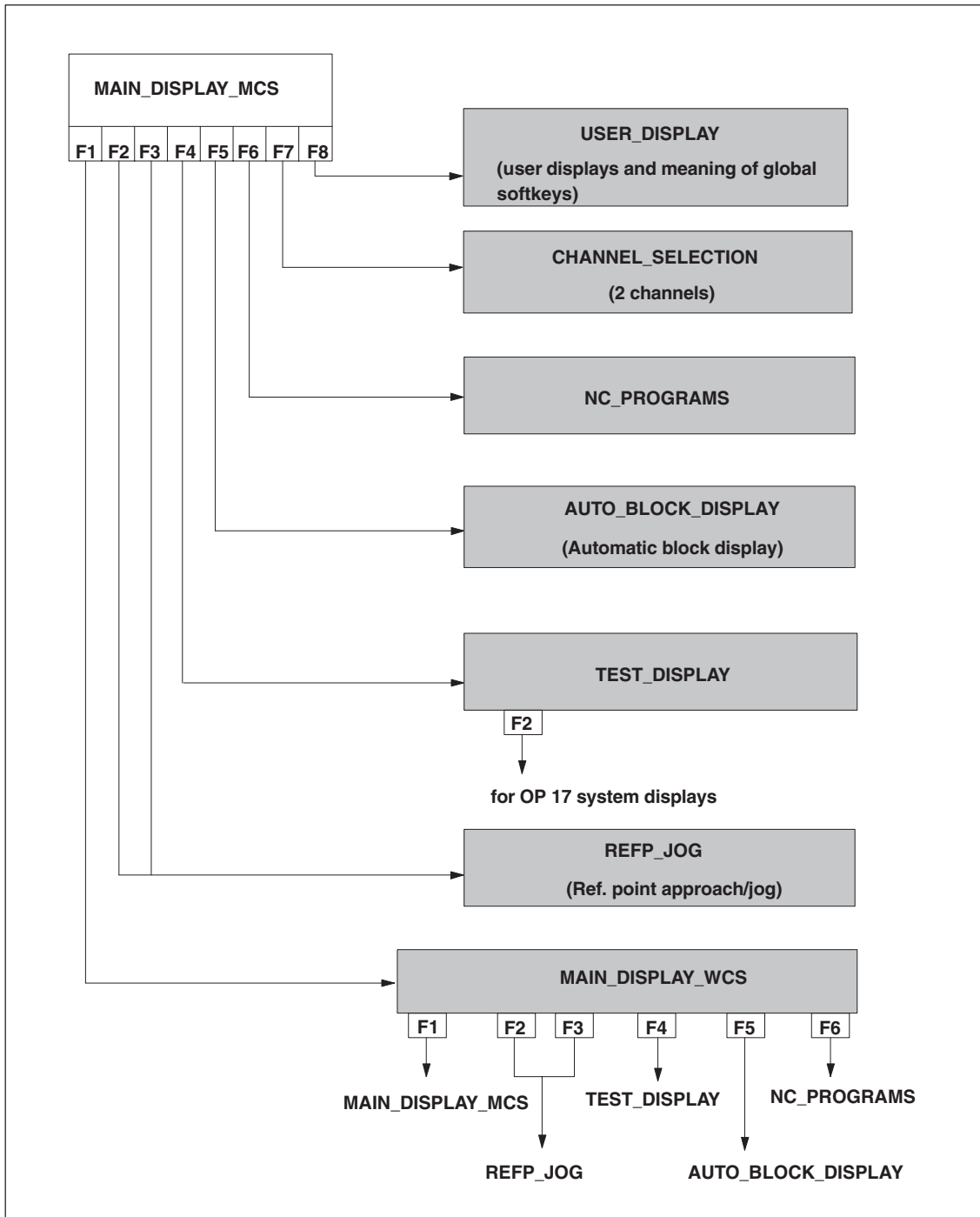


Fig. 8-2 Menu tree of the OP 17 user interface

Configuration example: 1 channel with 4 axes (file: ...\\op17_ch1fm357.pdb)

Meaning of the global function key assignments:

K1 to K4 (4 axes)

K5 – Acknowledge auxiliary function (user DB “FMx”, DBX109.0)

K6 – Start (user DB “FMx”, DBX108.1)

K7 – Stop (user DB “FMx”, DBX108.3)

K8 – Reset (user DB “FMx”, DBX108.7)

K9 – “Jog” mode (user DB “FMx”, DBX120.2) and
“Reference point approach” submode (user DB “FMx”, DBX121.2)

K10 – “Jog” mode (user DB “FMx”, DBX120.2)

K11 – “Automatic” mode (user DB “FMx”, DBX120.0)

K12 – MAIN_DISPLAY_MCS

Shift + K1 = Feed stop (user DB “AXy”, DBX4.3) = FALSE
Controller enable (user DB “AXy”, DBX2.1) = TRUE

Shift + K2 = Feed stop = TRUE
Controller enable = FALSE

Shift + K9 to K15 = Path override 0 % to 120 % (user DB “FMx”, DBX106) and
Override (user DB “AXy”, DBB0)

K14 – OP mode “Offline”

K15 – OP mode “Online”

K16 – OP mode “Transfer”

“REFP_JOG” display:

F1 – Path override “Rapid traverse override active” and “Feedrate override active” (user DB “FMx”, DBX107.6/7) for selected axis = TRUE

F2 – Path override “Rapid traverse override active” and “Feedrate override active” for selected axis = FALSE

F3 – “Rapid traverse override” (user DB “AXy”, DBX4.5) for selected axis = TRUE

F4 – “Rapid traverse override” for selected axis = FALSE

Procedure

To move the axis, press **Shift + K1** (“Feed stop” = FALSE and “Controller enable = TRUE”). Only the currently selected axis (x) is enabled.

You can reset these signals with **Shift + K2**.

The global function key assignment is also shown in the sample configuration in the “USER_DISPLAY” screen.

Notice

If not all four axes are used, the corresponding softkeys (K1 to K4) in the sample project should be cleared.

Configuration example: 2 channels with 2 axes each (file: ...\\op17_ch2fm357.pdb)

Meaning of the global function key assignments:

- K1 and K2 (axis 1 and 2 in channel 1, axis 3 and 4 in channel 2)
- K9 – “Jog” mode (user DB “FMx”, DBX120.2+n) and
“Reference point approach” submode (user DB “FMx”, DBX121.2+n)
- K10 – “Jog” mode (user DB “FMx”, DBX120.2+n)
- K11 – “Automatic” mode (user DB “FMx”, DBX120.0+n)
- K12 – MAIN_DISPLAY_MCS
- Shift + K1 = Feed stop (user DB “AXy”, DBX4.3+m) = FALSE
Controller enable (user DB “AXy”, DBX2.1+m) = TRUE
- Shift + K2 = Feed stop = TRUE
Controller enable = FALSE
- Shift + K9 to K15 = Path override 0 % to 120 % (user DB “FMx”, DBX106+n) and
Override (user DB “AXy”, DBB0+m)
- K14 – OP mode “Offline”
- K15 – OP mode “Online”
- K16 – OP mode “Transfer”

Meaning of the local function key assignments (valid only for the displays in this project):

- K5 – Acknowledge auxiliary function (user DB “FMx”, DBX109.0+n)
- K6 – Start (user DB “FMx”, DBX108.1+n)
- K7 – Stop (user DB “FMx”, DBX108.3+n)
- K8 – Reset (user DB “FMx”, DBX108.7+n)

“Channel selection” display:

- F1 – Channel 1
- F2 – Channel 2

When the channel is selected, the addresses of the variables used are adapted to this channel. The addresses of the axis-specific variables are also defined.

“REFP_JOG” display:

- F1 – Path override “Rapid traverse override active” and “Feedrate override active” (user DB “FMx”, DBX107.6/7) for selected axis = TRUE
- F2 – Path override “Rapid traverse override active” and “Feedrate override active” for selected axis = FALSE
- F3 – “Rapid traverse override” (user DB “AXy”, DBX4.5+m) for selected axis = TRUE
- F4 – “Rapid traverse override” for selected axis = FALSE

Procedure

To move the axis, press **Shift + K1** (“Feed stop” = FALSE and “Controller enable = TRUE). Only the currently selected axis (x) is enabled.

You can reset these signals with **Shift + K2**.

The global and local function key assignments are also shown in the sample configuration in the “USER_DISPLAY” screen.

MAIN_DISPLAY_MCS

The contents of individual displays can be found in the sample configuration.

The figure below shows you an example screen layout for “MAIN_DISPLAY_MCS”.

FM357-2	MAIN_DISPLAY_MCS	Mode: {V7_ba}	Ov: {V_Over_akt1}
Axis	Actual value	Distance to go	NC: {V_stopCond}
{V_Ma_na1}	{ist_pos1}	{ist_rest1}	
{V_Ma_na2}	{ist_pos2}	{ist_rest2}	
{V_Ma_na3}	{ist_pos3}	{ist_rest3}	
{V_Ma_na4}	{ist_pos4}	{ist_rest4}	
Error: {VAR_210}			
WCS	Refp	Jog	Res
		Auto	P
		Sel	User

Fig. 8-3 Main screen MCS PIC_G

The sample configuration is intended as a starting aid for your project. Copy the file **op17_ch1fm357.pdb** or **op17_ch2fm357.pdb**. You can edit the copy to suit your application.

Selection of operating and display variables

The variables which can be read or written by the OP 17 can be found in the following locations:

- User DBs (for description, see Chapter 6)
Target system = Control_CPU
- Symbol list of the NC-VAR Selector
Target system = Control_357

Symbol list

The sample configuration contains a pointer to the symbol list and is available when you select the variables of the FM 357-2. If you want to use a new symbol list (e.g. when making changes to variables or adding new variables), you can copy variables from the NC-VAR Selector and include them your project.

The NC-VAR Selector is supplied with the FM 357-2 configuring package. The installation is described in Section 6.3.4.

You will find the symbol list in the ProTool project. Select the “Controllers” directory in this project. In this directory, select “Control_357”. Select **Edit > Properties > Parameters > Symbol List** and select the path for your symbol list.

8.2 Troubleshooting on the OP 17 (configuration example)

Error display

You can display errors (e.g. read and write errors in NC variables) or error states, which may occur in your user program, on the OP 17.

This Section will use an example (FB 2 troubleshooting, read NC variable) to explain how you can configure troubleshooting on the OP with the “ProTool/Lite or ProTool” configuring tool.

Notice

Function block FB 2 must be supplied with input and output parameters before it is called. If error bit Error = TRUE after the block has been called, you can evaluate parameter State to establish the error cause.

Please proceed as follows:

1. Open your ProTool project by selecting **File > Open**.
2. Select item **Displays** and press button **New**. In the dialog which follows, position the cursor at the point where the error text must be displayed. Select menu commands **Display > Edit/insert field** to activate the **Input/output** dialog.

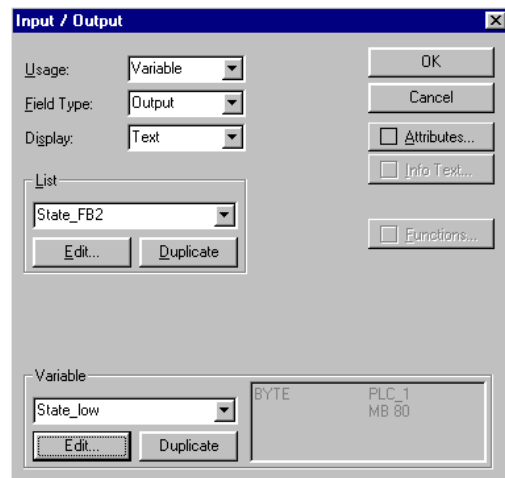


Fig. 8-4 Input/output dialog

3. You must enter/alter the following settings in this screen:

- Use: Select **Variable**
 - Field type: Select **Output**
 - Representation: Select **Text symbol**
- The **List** field is displayed in this dialog
- In dialog field **Variable** , press button **Edit**.

In the dialog which follows, set your variable to the State parameter of the FB. Please remember to enter CPU for “Control of MPI nodes” (default Control_1).

Confirm your inputs with **OK**.

4. In dialog field **List**, press button **Edit**

The **Symbol list/text** dialog is activated

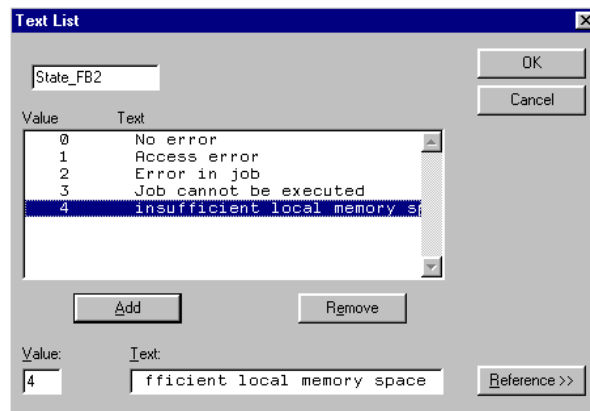


Fig. 8-5 Symbol list/text dialog

5. In dialog field **Value** enter the error number as specified in Table 6-9, and enter the error text in the **Text** field. When you press button **Add**, the entry is transferred to the symbol list field (see Fig. 8-5).
6. After you have entered the error number and error text, press button **OK**.
7. End the **Input/output** dialog by pressing **OK**.

The configured State variable now appears in your display dialog.

Since the error status consists of a high and a low byte (see Table 6-9) you should generate two of these variables to obtain a comprehensive troubleshooting function.

To do so, proceed as described above (starting at item 2. menu command **Display > Edit/insert field**).

When you transfer the project to the OP and start function block FB 2 (FB 2 must be included in the user program, see Section 6), the associated error text appears on the OP (when an error is detected).

8.3 HMI sample interface for SIMATIC HMI devices

You will find the preconfigured user interface and a description in the following directories:

- OP 27
 - [STEP7 directory]\EXAM-
PLES\FM357-2\zdt16_01_fm357-2_op_ex\op27_ch1fm357.pdb
 - [STEP7 directory]\EXAM-
PLES\FM357-2\zdt16_01_fm357-2_op_ex\op27_fm357.doc
- TP 170 B
 - [STEP7 directory]\EXAM-
PLES\FM357-2\zdt16_01_fm357-2_op_ex\tp170_ch1fm357.pdb
 - [STEP7 directory]\EXAM-
PLES\FM357-2\zdt16_01_fm357-2_op_ex\tp170_fm357.doc
- MP 270 B
 - [STEP7 directory]\EXAM-
PLES\FM357-2\zdt16_01_fm357-2_op_ex\mp270_ch1fm357.pdb
 - [STEP7 directory]\EXAM-
PLES\FM357-2\zdt16_01_fm357-2_op_ex\mp270_fm357.doc

The user interface must be adapted to your project/plant (for handling instructions, see also Section 8.1).



9

Description of Functions

Section overview

Section	Title	Page
9.1	Configuration	9-3
9.2	Encoder	9-18
9.3	Position control	9-27
9.4	DSC (Dynamic Servo Control)	9-37
9.5	Velocity and acceleration	9-39
9.6	Monitoring	9-45
9.7	Referencing and alignment	9-56
9.8	Event-controlled program calls	9-68
9.9	Output of M, T and H functions	9-73
9.10	Inputs/outputs	9-76
9.11	Limit switching signals (software cams)	9-85
9.12	Operating modes	9-102
9.13	NC program execution	9-105
9.14	Asynchronous subprograms (ASUB)	9-107
9.15	Protection zones	9-109
9.16	Motion coupling	9-114
9.17	Measurement	9-142
9.18	Travel to fixed stop	9-146
9.19	EMERGENCY STOP	9-155
9.20	Controlling	9-157
9.21	Axis replacement	9-163

General

The following firmware versions are available on memory card for the FM 357-2:

- FM 357-2L
- FM 357-2LX
- FM 357-2H

Table 9-1 Distinction between FM 357-2L, FM 357-2LX and FM 357-2H

FM 357-2 function	L	LX	H ¹⁾	in Section
Gantry	–	x	x	9.16.2
Travel to fixed stop	–	x	x	9.18, 10.11
Oscillation	–	x	x	10.33
Feed interpolation	–	x	x	10.5.2
“Path velocity” system variable	–	x	x	10.32, Tab. 10-11
Overlaid motion in synchronized actions	–	x	x	9.16.6
SPLINE interpolation	–	x	x	10.5.10
Polynomial interpolation	–	x	x	10.5.11
Subroutine as action (synchronized action)	–	x	x	10.32
Static synchronized actions in all operating modes	–	x	x	10.32
Axial measurement	–	x	x	9.17, 10.10.2
Axial measurements in synchronous actions	–	x	x	10.32
Max. number of channels	4	4	1	9.1
REPOS via program	–	x	x	10.22
Tangential control	–	x	x	9.14.4
Electronic gear	–	x	–	9.16.4
Handling transformation	–	–	x	13

1) Not listed as a separate product variant in the description of functions.

The parameterization of the functions described in this Section is supported by the “Parameterize FM 357-2” tool.

Notice

This documentation specifies all units for standard system parameters in the **metric** system of measurement.

9.1 Configuration

Internal scaling system

When you start parameterizing, you must define the internal scaling system. All further value inputs and ranges refer to this setting.

You can set the internal system of measurement for linear axes (see axis type) to the following units:

- metric
- inches

Values are processed in the following basic units in the “Parameterize FM 357-2” tool and in the FM 357-2:

- 0.001 mm
- 0.0001 inches
- 0.001 degrees (rotary axis)

Example

The relationship between the system of measurement and internal values is illustrated on the basis of example values.

Scaling system	Internal values	Input in the interface (example)
mm	10^{-3} mm	10.995 mm
inches	10^{-4} inches	1.0995 inches
degrees	10^{-3} degrees	3600.001 degrees

In addition to the internal system of measurement, you can also switch the programming system of measurement in the NC program (see Section 10.2.6).

Notice

If you change the internal scaling system later, e.g. after you have already input velocity or position values, these values are interpreted in the new system of measurement (i.e. incorrectly). In this case, you will need to enter the values again in accordance with the new system of measurement.

The limit values for velocity or position values are only updated when the scaling system is changed with the next restart.

Override coding

The path override (user DB “FMx”, $DBB106+n$) and the axis override (user DB “AXy”, $DBB0+m$) can be reported from the CPU either as a Gray code or as a binary code. The parameter “Override coding” specifies how the coding is interpreted by the FM.

For further information on the override, please refer to Section 6.6.3.

9.1.1 System rates

(see also Section 6.1.6)

Max. cycle time of a user program

The max. cycle time [ms] informs the FM 357-2 of the duration of an OB 1 cycle. It is evaluated for output of auxiliary functions in G64 mode.

The path feedrate is reduced such that the "Auxiliary function acknowledgement" (user DB "FMx", DBX109.0+n) can be provided within an OB1 cycle.

Servo clock

This parameter determines at which intervals the servo task is started in the FM.

Table 9-2 Parameter for the servo cycle

Parameter	Value/meaning	Unit
Servo cycle	3 (default value, fixed with PROFIBUS-DP drive) 0.5...32	[ms]

Ration of servo cycle to IPO cycle

This parameter specifies after how many servo cycles the IPO cycle is started.

Table 9-3 Parameter ration of servo to IPO cycle

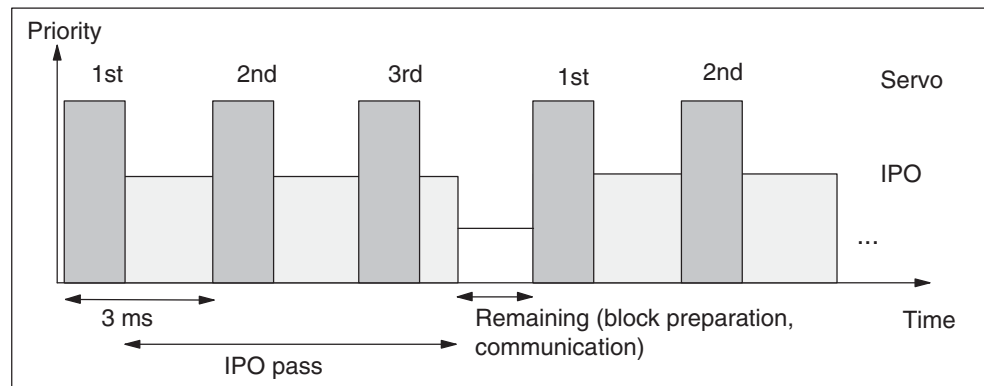
Parameter	Value/meaning	Unit
Ration of servo to IPO cycle	3 (default value) 1...100	–

A reduction of the cycle times is only possible depending on the utilization of the FM (e.g. number of axes, number of the channels or number of the active synchronized actions).

If the permissible utilization is exceeded, error message 4 240 is issued (calculation time on IPO or position controller level exceeded).

Unnecessary increases in the cycle times, on the other hand, reduce the performance of the position control system and/or the response times of the interpolator.

Example: 3 ms servo cycle, ratio of servo cycle to IPO cycle = 3



9.1.2 Axis configuration

Active axis

An axis can be activated or deactivated via its axis number. Four axes are active by default. The order of the axis numbers is fixed (ascending, continuous). You can deactivate axes for test purposes. The associated machine data are retained, and are valid again when you activate the axis.

Note:

The 5th axis can only be operated in simulation. Thus, it is possible, e.g. to create an internal master value for a curve table.

Machine axis name / channel and special axis names / geometry axis name

You can assign different names to the various types of axis of the FM 357-2.

- **Machine axis**

This refers to all axes installed on the machine tool. The machine axis names are used for parameterization, for display of actual values in the machine coordinate system, and for certain error messages.

- **Geometry axis**

These axes represent the workpiece coordinate system. A machine axis is assigned to each geometry axis. The geometry axis name is used for NC programming and workpiece coordinate display. Up to three geometry axes can be parameterized. Geometry axes are always linear axes.

- **Special axis**

Special axis is the name given to all machine axis which are not geometry axes. The name is used analogously to the geometry axis name. Linear or rotary axes can be special axes.

- **Channel axis**

... are all axes in a channel. Machine, channel and geometry axes can be different (e.g. for handling transformation).

Notice

The following cannot be used as an axis name:

- Address letters (D, E, F, G, H, I, J, K, L, M, N, P, R, S, T)
 - Instructions which are used for programming
-

Axis type

You can select the following types of axes:

- Linear axis
- Rotary axis
- Modulo rotary axis

Notice

Select the axis type when you start parameterizing. The internal system of measurement switches from mm (inches) to degrees and vice-versa when you change the axis type. All values which you have already entered for the corresponding axis are therefore interpreted incorrectly.

Linear axes:

Linear axes can be moved within two range limits.

Traversing range: $\pm 999\,999.999$ mm or $\pm 399\,999.999$ inch

Programming range: $\pm 999\,999.999$ mm or $\pm 399\,999.999$ inch

Rotary axes:

Rotary axes are programmed in degrees.

They move between two range limits.

Traversing range: $\pm 999\,999.999^\circ$

Programming range: $\pm 999\,999.999^\circ$

Modulo rotary axes:

On modulo rotary axes, the actual value is reset to "0" after one revolution, thus providing you with an endless traversing range. One revolution is always 360° .

Traversing range: infinite

Programming range: $0\dots359.999^\circ$

Drive

The following options for configuring the drive are available:

- **Simulation**

The speed control loop of an axis is simulated internally. There is no actual value measurement and no setpoint output. Here, the axis “moves” with a following error, like a real axis. The function can be used for test purposes.

Note:

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

During simulation, the axis-specific interface signals are output to the CPU depending on the parameter “VDI output”.

- **Servo drive (analog)**

The axis is operated with a servo drive. The control system for the axis consists of a current and speed control loop in the servo drive (analog) and a higher-level position control loop in the FM 357-2.

- **Stepper motor with/without encoder**

The axis is operated with a stepper motor. The step drive is controlled over a pulse interface.

Where the stepper motor does not have an encoder, the stepper motor pulses are fed back internally as the actual value.

- **PROFIBUS DP drive**

Up to 4 axes can be operated in position control mode via PROFIBUS DP. The transfer of the setpoints and actual values to/from the drive is carried out synchronously to the cycle via the PROFIBUS DP protocol.

The module contains fixed SDBs for the most important configurations (system data blocks). Additionally, you can reload external SDBs via SIMATIC S7 HW Config. The PROFIBUS-DP configuration is determined by the SDBs.

Please note that the PROFIBUS-DP connection of the FM357-2 is **only** intended for PROFIBUS-DP **drives**.

PROFIBUS-DP configuration:

- PROFIBUS-DP drive: “SIMODRIVE 611-U”

Single or double-axis modules are possible. According to the selected axis modules, a fixed SDB is used for SIMODRIVE 611-U.

- PROFIBUS-DP drive: “SIMOVERT MASTERDRIVES MC”

Only single-axis modules are possible. A fixed SDB is used for MASTERDRIVES.

- PROFIBUS-DP drive: “External SDB”

An SDB loaded via *SIMATIC S7 HW Config* will be used. Thus, it is possible to parameterize the drive using a programming device (PG) connected to the MPI or PROFIBUS-DP interface of the CPU (routing). Either MASTERDRIVES or SIMODRIVE 611-U drives may be used.

The assignment of the axes to the PROFIBUS-DB addresses is continued to be carried out in the start-up tool.

Notice

Mixed operation of PROFIBUS-DP drives and servo or stepper motor drives is possible.

Axis module (fixed SDB)

The axes can be assigned the following axis modules.

Parameter	PROFIBUS-DP drive:		Unit
	SIMODRIVE 611-U	MASTERDRIVES	
Axis module	none (default value)	none (default value)	–
	1st single-axis module	1st single-axis module	
	2nd single-axis module	2nd single-axis module	
	3rd single-axis module	3rd single-axis module	
	4th single-axis module	4th single-axis module	
	1st double-axis module		
	2nd double-axis module		

Notice

Certain selected double-axis modules must be assigned 2 axes; otherwise, an error message is output during power-up.

The following addresses must be parameterized in the drive:

Axis module	PROFIBUS-DP address:		Unit
	SIMODRIVE 611-U	MASTERDRIVES	
1st single-axis module	10	30	–
2nd single-axis module	11	31	
3rd single-axis module	20	32	
4th single-axis module	21	33	
1st double-axis module	12	–	
2nd double-axis module	13	–	

The module contains the following fixed SDBs:

Module	SDB No. PROFIBUS-DP:		Unit
	SIMODRIVE 611-U	MASTERDRIVES	
1st double-axis module plus (optional) 2nd double-axis module	SDB1	–	–
1st double-axis module plus (optional) 1st single-axis module plus (optional) 2nd single-axis module or vice versa: 1st single-axis module plus (optional) 2nd single-axis module plus (optional) 1st double-axis module	SDB2	–	–
1st single-axis module plus (optional) 2nd to 4th single-axis modules	SDB3	SDB5	–

Mixed operation of MASTERDRIVES and SIMODRIVE 611-U drives is not possible using the fixed SDBs. The start-up tool will select the right SDB depending on the selected module and the axis assignment.

For the drive MASTERDRIVES, the PROFIBUS-DP cycle is set to the fixed value of 3 ms. In the drive, P340 = 5.3 (kHz) and P770 = 1 must be set.

The PROFIBUS-DP cycle for the drive SIMODRIVE 611-U is set to the fixed value of 1.5 ms.

With double-axis modules, the assignment of an axis to the PROFIBUS-DP drive is carried out in the ascending order. This order can be interrupted by another drive (e.g. analog servo drive).

Example:

Axis number	1	2	3	4
	Servo drive (analog)	PROFIBUS drive – double-axis module (1st PROFIBUS-DP axis)	Servo drive (analog)	PROFIBUS drive – double-axis module (2nd PROFIBUS-DP axis)

(PROFIBUS-DP SDB1, 2nd axis module does **not** exist)

Logic address (external SDB)

If you are using an external SDB, the logic PROFIBUS-DP address must be entered here.

Default values of the logic addresses:

Axis number	1	2	3	4
Logic address	258	290	322	354

Message frame type (external SDB)

When using the external SDB, define the message frame type here.
The following message frame types can be set:

- Standard type 1
- Standard type 2
- Standard type 3
- Standard type 4
- Standard type 5
- 611-U type 101
- 611-U type 102
- 611-U type 103
- 611-U type 104
- 611-U type 105

Use the message frame types specified in Table 9-4.

Note for creating external SDBs

The requirement to create external (reloadable) SDBs and to carry out the routing to the PROFIBUS-DP drive is software version 5.2 of the FM357-2, STEP 7, version 5.2, and DRIVE ES V 5.1 SP2.

Set the following parameters in the FM357-2:

- PROFIBUS-DP drive: "External SDB"
- Frame message type
- Logical address

Some project creation samples are to be found in the Sections 5.2.1 through 5.2.3.

The drive parameterization is possible either via the MPI or the PROFIBUS-DP interface of the CPU. When using the PROFIBUS-DP interface, a routing to the K bus is provided in the CPU. Thus, the communication times are longer, compared with the MPI interface.

The most important PROFIBUS-DP parameters are contained in the table below. Please note that any other parameterization has not been tested (marked with a **) or is not supported by FM357-2 (marked with a *).

Table 9-4 PROFIBUS-DP parameters

PROFIBUS-DP Parameter	PROFIBUS-DP Drive:		Unit
	SIMODRIVE 611-U	MASTERDRIVES	
Data Transfer Rate *	12	12	Mbit/s
Equidistant bus cycle *	yes	yes	–
DP cycle *	1.5	3,0	ms
Master applications cycle *	3.0	3,0	ms
I/O addresses **	258, 290, 322, 356 Logical addresses (external SDB)		–
PROFIBUS-DP address: **			–
• Single-axis modules	10, 11, 20, 21	30, 31, 32, 33	
• Double-axis modules	12, 13	–	
Frame message type: **			–
• without DSC	Message frame 102 PZD-6/10	Standard frame 2 PZD-4/4	
• with DSC	Message frame 105 PZD-10/10	Standard frame 5 PZD-9/9	

Note with regard to the drive parameterization

The drive parameterization for SIMODRIVE 611-U is carried out via:

- Operator and display unit on the SIMODRIVE 611-U
- Parameterization and start-up tool “SimoCom U”

You will need the following documentation: Description of Functions
SIMODRIVE 611 universal, Order No.: 6SN1197-0AB20-0AP[edition]

The drive parameterization for SIMOVERT MASTERDRIVES MC is carried out via:

- Operator and display unit on the SIMOVERT MASTERDRIVES MC
- Parameterization and start-up tool “DRIVE Monitor”

You will find the relevant documentation on the supplied CD ROM under **Compendium** in the directory **MASTERDRIVES_MC**.

Notice

The CD ROM of the parameterization tool contains the script file **stdtlg5_fm_v1.ssc** under MASTERDRIVES. This script file must be run after the basic parameterization, e.g. using the DRIVEMONITOR tool. The most important settings required to run the MASTERDRIVES are carried out on the FM 357-2.

External master value

In conjunction with the function “Master-value link” (see Section 9.16.3), an axis can be defined as an external master. For actual value acquisition, an encoder must be connected to the appropriate measuring system interface. The FM internally creates a “simulated” master value as an input quantity for the curve table.

No position control is active and no setpoints are output.

The interface signal “Follow-up mode” (user DB “AXy”, DBX1.4+m) must be set.

VDI output

If an axis is run in the simulation mode, it is possible to define via the parameter “VDI output” whether the FM signals the interface signals user DB “AXy” to the CPU.

Thus, you can test, e.g. sequences in conjunction with axis motions in the CPU.

Controlling

allows an axis to be traversed in the speed-controlled mode. In addition, the interface signal “Control axis” (user DB “AXy”, DBX8.1+m) must be set.

For further information, see Section 9.20 (“Controlling”).

Global measuring

Using the global measuring function in conjunction with PROFIBUS-DP axes, the sensing probe at the FM 357-2 can be used without any additional wiring to the drive.

For further information, see Section 9.17 (“Measuring”).

DSC (Dynamic Servo Control)

Using the DSC function in conjunction with PROFIBUS-DP drives, the position controller can be relocated into the drive, allowing calculations in the fast speed-controller clock.

The FM provides a certain number of position setpoints per servo clock as a control difference.

Compared with the position controller calculated in the FM, higher K_V factors (loop-gain factors) can be achieved in this way.

For further information, see Section 9.4.

Independent CPU axes

This parameter defines the max. number of independent CPU axes. No indirect axis assignment is carried out .

For further information, see Section 6.5.1 “Axis control from the CPU”.

9.1.3 Channel configuration

Active channels

A maximum of 4 channels can be activated. The channels are independent with respect to:

- NC program execution
- Axis movements
- Operating modes
- Channel-specific interface signals (e.g. Start, Stop, Reset)

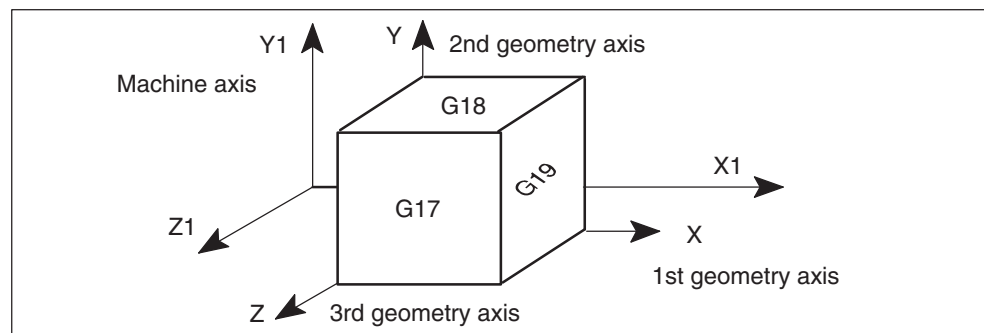
Channel assignment

If several channels are active, you can assign the axes to different channels. By default, all channels are assigned to the 1st channel. If you intend to use the function “Axis replacement” (see Sections 9.21 and 10.36), you must assign the axis to several channels and a master channel.

Geometry axes

The geometry axes must describe a rectangular coordinate system. The machining planes (Section 10.2.7) and the effect of the tool offsets (Section 10.16) are defined by the first, second and third geometry axes.

In the example, we have illustrated the usual assignment.



If an axis is used in several channels (axis replacement), it may have either different or none geometry axis number in the individual channels.

9.1.4 Parameters for configuration

The following parameters are relevant with respect to configuring:

Table 9-5 Configuration parameters

Parameters	Value/Meaning	Unit
Internal scaling system	metric = 10^{-3} (default value) inches = 10^{-4}	[mm] [inches]
Override coding	Gray (default value) The override value delivered from the CPU is interpreted by the FM as a Gray code. Binary The override value delivered from the CPU is interpreted by the FM as a binary code.	–
Max. cycle time of user program	40 (default value) 10 to 200	[ms]
Servo cycle	3 (default value, fixed for PROFIBUS-DP drive) 0.5...32	[ms]
Ratio between servo cycle and IPO cycle	3 (default value) 1...100	–
Number of R parameters (see Section 10.19)	100 (default value) 0 to 10,000	–
Number of FIFO ranges (see Section 10.21)	0 (default values) 0...10	–
Start of FIFO range (see Section 10.21)	0 (default value) 0...max. value ¹⁾ 1) dependent on "Number of R parameters" and "Number of FIFO elements"	–
Number of FIFO elements (see Section 10.21)	0 (default value) 0...max. value ¹⁾ 1) dependent on "Number of R parameters" and "Start of FIFO range"	–
FIFO range sum generation (see Section 10.21)	no (default value) yes \$AC_FIFO[3] contains the total (value) of all data elements	–
Number of curve tables (see Section 9.16.3)	0 (default value) 0 to 100	–
Number of curve segments (see Section 9.16.3)	0 (default value) 0 to 1,800	–
Number of curve table polynomials (see Section 9.16.3)	0 (default value) 0 to 3,600	–
Number of GUD variables globally FMs (see Section 10.22)	10 (default value) 0... ¹⁾ 1) depending on the free SRAM available	–

Table 9-5 Configuration parameters, continued

Parameters	Value/Meaning	Unit
Number of GUD variables globally channels (see Section 10.22)	40 (default value) 0... ¹⁾ 1) depending on the free SRAM available The number of variable names is parameterized. This name will apply to all channels. The value assignment, however, is channel-dependent (for one variable, max 4 values).	—
Reserved memory area (SRAM) for GUD values (see Section 10.22)	12 kB (default value) 0... ¹⁾ 1) depending on the free SRAM available	—
Number of protection zones see Section 9.15	0 (default value) 0 to 4	—
Active axes	4 (default value) 1 to 5 (5th axis only in simulation)	—
Axis name	Machine axis (X1, Y1, Z1, A1 – default value) Geometry axis (X, Y, Z – default value) Special axis (A – default value) (max. 8 characters) (B1, B – default, only possible in simulation mode)	—
Axis type	Linear axis = (10 ⁻³ mm or 10 ⁻⁴ inches) Rotary axis = (10 ⁻³ degrees) Modulo rotary axis = (10 ⁻³ degrees)	—
Drive	Simulation (default value) Servo drive (analog) Stepper motor without encoder Stepper motor with encoder PROFIBUS-DP: SIMODRIVE 611-U PROFIBUS-DP: MASTERDRIVES PROFIBUS-DP: External SDB	—
Axis module	None (default value) 1st ... 4th single-axis module 1st two-axis module 2nd two-axis module	—
Logic address (external SDB)	Default values: 258 290 322 354	—
Message frame type (external SDB)	Standard tape 2 (default value) Standard types 1...5 611U type 101...105	—
External master value	no (default value) The axis cannot be used as an external master value. yes The axis is an external master value.	—

Table 9-5 Configuration parameters, continued

Parameters	Value/Meaning	Unit
VDI output (for simulation)	no (default value) The user DB "AXy" interface signals are not transferred to the CPU. yes The user DB "AXy" interface signals are transferred to the CPU.	–
Controlling (see Section 9.20)	no (default value) Axis traverses in position control yes Axis can traverse controlled (speed-controlled)	–
Global measuring (see Section 9.17)	no (default value) No global measuring yes Global measuring	–
DSC (Dynamic Servo Control) (see Section 9.4)	no (default value) yes The position controller for the PROFIBUS-DP axis is calculated in the drive.	–
Independent CPU axes (see Section 6.5.1)	0 (default value) 0...4	–
Active channels	1 (default value) 1...4	–
Channel assignment	1 (default value) 1...4	–
Geometry axes	(1. X, 2. Y, 3. Z – default value)	–

9.2 Encoders

General

The following encoders can be connected to the measuring system interface of the FM 357-2:

- Incremental encoder
- Absolute encoder (SSI)

Distances and velocities are represented to:

- 0.001 mm or 0.0001 Inch (linear axis)
- 0.001 degrees (rotary axis)

The path resolution achieved by the encoder is calculated on the FM 357-2 from the distance per spindle revolution, the transmission ratio between the encoder and the mechanical system, and the number of increments per encoder revolution.

Selection of the encoder

The prerequisite for achieving a certain positioning accuracy is an nth degree improvement in path resolution by the encoder.

Recommended values for n		
Minimum	Optimum	Maximum
2	4	10

When configuring your system, you should choose an encoder that meets the positioning accuracy required in your application.

With the known design data of the machine axis and a desired resolution R:

$$R = \frac{1}{n} \cdot \text{Positioning accuracy} \quad [\text{mm}], [\text{inches}], [\text{degrees}]$$

yield a calculation of the necessary pulse number per encoder revolution according to the following relationship (taking a metric measuring system as an example):

Incremental encoder	Absolute encoder (SSI)
$I_G = \frac{S \text{ (mm)}}{4 \cdot i_{GS} \cdot A \text{ [mm]}}$	$S_G = \frac{S \text{ (mm)}}{i_{GS} \cdot A \text{ [mm]}}$

The following table gives you an overview of the data used in this calculation, and their meaning.

Symbol	Significance
I_G	Increments per encoder rotation (incremental encoder)
S_G	Increments per encoder rotation (absolute encoder)
S	Displacement per revolution of a spindle or turntable [mm/rev], [inches/rev], [deg/rev]
R	Required resolution [mm], [inch], [degrees]
4	Pulse multiplication (constant)
i_{GS}	Ratio between encoder and mechanism - Number of encoder revolutions $\left[\frac{\text{number of encoder revs}}{\text{spindle revolution}} \right]$ or $\left[\frac{\text{number of encoder revs}}{\text{turntable revolution}} \right]$

Notice

If unusual numbers of pulses or increments result, the encoder with the next-higher number of pulses or increments should be selected.

The general encoder configuration and the machine geometry are defined by the following parameters:

Parameters	Value/Meaning	Unit
Encoder version	Linear: Linear scale Rotary: Rotary encoder (default value)	–
Encoder mounting	Motor: Indirect path encoding (default value) Machine: Direct path encoding	–
Encoder type	Incremental: Incremental encoder (default value) Absolute: Absolute encoder (SSI)	–
Distance per spindle revolution	10 (default value) Value range 0.001 to 100 000	[mm/rev]
Load gearbox	Defines the transmission ratio of the load gearbox $\frac{\text{No. of motor revolutions}}{\text{No. of spindle revolutions}} = \frac{1 \text{ to } 10\,000}{1 \text{ to } 10\,000}$	–
Resolver gearbox	Defines the transmission ratio of the resolver gearbox $\frac{\text{No. of motor revolutions}}{\text{No. of encoder revolutions}} = \frac{1 \text{ to } 10\,000}{1 \text{ to } 10\,000}$	–
2nd encoder (only relevant for PROFIBUS drives)	no (default): no 2nd encoder yes 2nd encoder	–

2nd encoder for PROFIBUS drives

This parameter allows the use of a 2nd encoder at the PROFIBUS drive. The 1st encoder is assigned to the speed-control loop in the drive and has also to be parameterized in the drive. The 2nd encoder must be parameterized in the FM and provides the actual value for the position-control loop.

9.2.1 Incremental encoders

General

The encoders return pulses which are summed in the FM 357-2 to produce an absolute value. When the FM 357-2 is switched on, an unpredictable offset exists between the internal position value and the mechanical position of the axis. To generate the position reference, you must therefore reference the axis.

Variants

The following variants of application are permissible:

- **Rotary incremental encoders on linear axes**

Encoders with one zero pulse per revolution may be used. The number of encoder pulses must be a multiple of ten or a power of two.

- **Rotary incremental encoders on rotary axes**

Encoders with one zero pulse per revolution may be used. The number of encoder pulses must be a multiple of ten or a power of two. When the encoder is mounted indirectly, it must be guaranteed that the revolution of the rotary axis can be divided by the cyclic zero pulse without a remainder.

- **Linear scale on linear axes**

Scales may be used with at least one reference zero pulse, or with a cyclic zero pulse.

In comparison to rotary incremental encoders, instead of the encoder revolution a scale division is used as a basis here, corresponding for example to the segment between two zero-mark pulses.

Parameters for encoder adaptation

The following parameters are provided on the FM 357-2 for adapting incremental encoders:

Parameters	Value/Meaning	Unit
Increments per encoder revolution	2048 (default value) Value range 2...16 384 Number of increments per revolution on a rotary encoder	–
Scale division	0.01 (default value) Value range 0.001 to 100 Specifies the spacing of the marks on a linear scale. With an external pulse shaper electronic circuit (EXE), the multiplication factor must be taken into account (e.g. linear scale with 0.020 mm scale division and 10 times EXE → “Scale division” parameter = 0.002 mm)	[mm]

Example of an encoder adaptation

Linear axis with rotary encoder (5000 increments per revolution) on motor, load gearbox (gear ratio = 2:1), leadscrew (distance per spindle revolution = 10 mm)

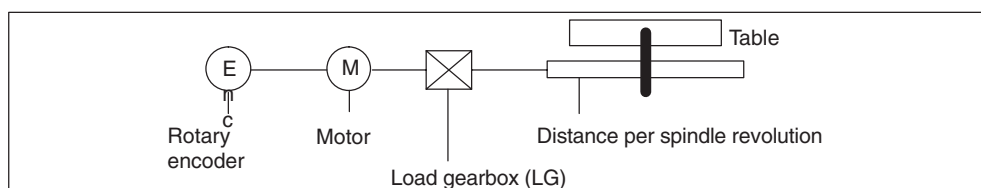


Fig. 9-1 Rotary encoder on motor

Linear axis: Internal precision = 1000 increments per mm

Encoder: Number of increments per revolution = $5000 * 4 = 20\,000$ increments (the encoder increments are multiplied internally).

Machine: Distance per motor revolution = $2 * 10\text{ mm} = 20\text{ mm}$

Calculation: Encoder increments : mm = $20\,000 : 20 = 1\,000$

Result:

The ratio between the internal increments per mm and the encoder increments : mm is 1:1.

Connection of encoders

see Section 4.7

9.2.2 Absolute encoders (SSI)

General

Absolute encoders (SSI) have a number of important advantages over incremental encoders, i.e.

- Longer cable lengths
- Reliable data capture by using a single-step GRAY code
- You don't need to synchronize the encoder when you switch it on

Variants

- **Rotatory absolute encoders (SSI):**

- at linear axes

If the function "Traversing range extension" is not active, it must be guaranteed that the absolute range of the encoder corresponds at least to the distance to be traversed by the axis. Overtraveling of the encoder zero is not permitted.

- at rotary axes

It must be guaranteed that the absolute value registered by the encoder corresponds exactly to one rotary axis revolution.

- **Linear scale absolute (SSI):**

It must be guaranteed that the absolute range of the encoder corresponds at least to the distance to be traversed by the axis. Overtraveling of the encoder zero is not permitted.

Parameters for encoder adaptation

The following parameters are provided on the FM 357-2 for adapting absolute encoders:

Table 9-6 Parameters for absolute encoders

Parameters	Value/Meaning	Unit
Sensing probe connection	Input 4 (default value) Input 5	–
Traversing range extension	ON (default value) OFF	–
Encoder revolutions in the absolute range	4,096 (default value) $2^0 \dots 2^{13}$ Number of encoder revolutions without overtraveling the absolute range	–
Message frame length	25-bit multi-turn (default value) 13-bit single-turn 21-bit multi-turn	–

Table 9-6 Parameters for absolute encoders, continued

Parameters	Value/Meaning	Unit
Increments per encoder revolution	8192 only with 25-bit multi-turn and 13-bit single-turn 4096 2048 ... 2^1	—
Scale division	0.01 (default value) Value range 0.001 to 100 Specifies the spacing of the marks on a linear scale.	[mm]

Encoder revolutions in the absolute range

This parameter definitely specifies the traversing range for rotary absolute encoders (encoder absolute range).

Example:

Message frame length	Increments per encoder revolution	Encoder revolutions in the absolute range
25-bit multi-turn	4,096 (2^{12})	4,096 (2^{12})
21-bit multi-turn	4,096 (2^{12})	256 (2^8)
13-bit single-turn	4,096 (2^{12})	1 (2^0)

Traversing range extension

This function is intended for **rotary absolute encoders** to extend the traversing range of the axis beyond the limits of the absolute range of the encoder.

The FM corrects the offset in the adjustment in cyclic operation by overtraveling the encoder absolute range (encoder zero):

- Positive direction of rotation of the measuring system
Maximum value of absolute range $\rightarrow 0$
Adjustment value = adjustment value + absolute range
- Negative direction of rotation of measuring system $0 \rightarrow$
maximum value of absolute range
Adjustment value = adjustment value – absolute range

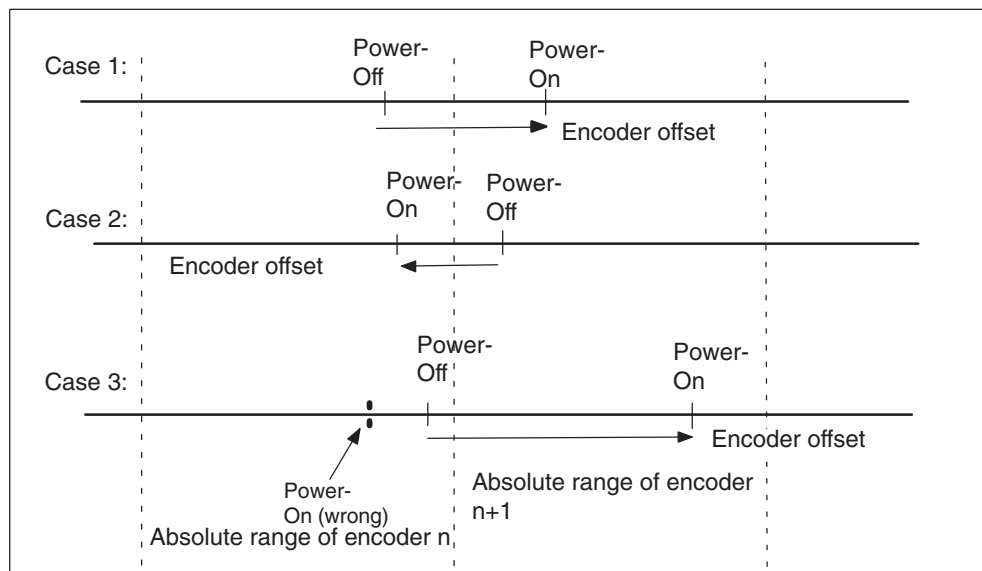
With PowerOn, the encoder position is restored once, i.e. the appropriate absolute range is selected using the “shortest-way-rule”. This, it is possible to correct mechanical offsets over the encoder absolute range in the PowerOff state.

The first encoder position (position after PowerOn) is carried out 0.4 seconds after the cyclic operation of the FM has been reached. At this moment, it is imperative that the encoder provides valid actual values.

Note:

To be able to restore the correct encoder position, the position difference between PowerOff/On may not be greater than the half of the appropriate encoder absolute range.

Example:



In the cases 1 and 2, the FM can assign the position to the appropriate absolute range after PowerOn.

In case 3, the encoder offset after PowerOff is greater than the half of the encoder absolute range. The FM will restore an incorrect position (PowerOn wrong). **No** error message is provided.

You can deactivate the function “Traversing range extension” if:

- the traversing range of the axis is smaller than the absolute range of the encoder
- the encoder zero position is not reached in cyclic operation or in the PowerOff state

Speed limits

The encoder actual value is read once per servo cycle. To acquire the actual value unambiguously, the following condition must be fulfilled:

$$\text{Distance traversed per servo cycle} < 0.5 * \text{absolute range of encoder}$$

This limitation should be observed in conjunction with single-turn encoders!

Example:

Servo cycle	Maximum speed of single-turn encoder (0.5 * rev./servo cycle)	Unit
2 ms	15,000	[r.p.m.]
3 ms	10,000	
4 ms	7,500	
5 ms	6,000	
6 ms	5,000	

Example of an encoder adaptation

Linear axis with absolute encoder (4 096 increments per revolution, 256 revolutions) on motor, load gearbox (gear ratio = 3:5), leadscrew (distance per spindle revolution = 10 mm)

Linear axis: Internal precision = 1 000 increments per mm

Encoder No. of increments per revolution = 4 096 = 2^{12}
No. of revolutions = 256 = 2^8

Machine: Distance per revolution = 3 : 5 * 10 mm = 6 mm

Calculation: Encoder increments per mm = 4 096 : 6 = 682.67

Result:

The ratio between the internal increments per mm and the encoder increments per mm is 1 000 : 682.67.

Notice

The encoder covers an absolute traversing distance of $256 * 6 \text{ mm} = 1\,536 \text{ mm}$.

If no traversing range extension is activated and if the encoder zero is located directly at the beginning of the traversing range, a distance $< 1,536 \text{ mm}$ can be traversed

Connecting the encoders see Section 4.7

9.2.3 Stepper motor

Parameters

When you are using a stepper motor, you must also enter the number of steps per revolution.

Parameters	Value/Meaning	Unit
Increments per motor revolution	1 000 (default value) 2 to 1 000 000 Number of increments per revolution	–

The parameter is required for stepper motors with and without an encoder.

9.3 Position control

General

The control system for an axis consists of the speed control loop of the drive and a higher-level position control loop in the FM 357-2.

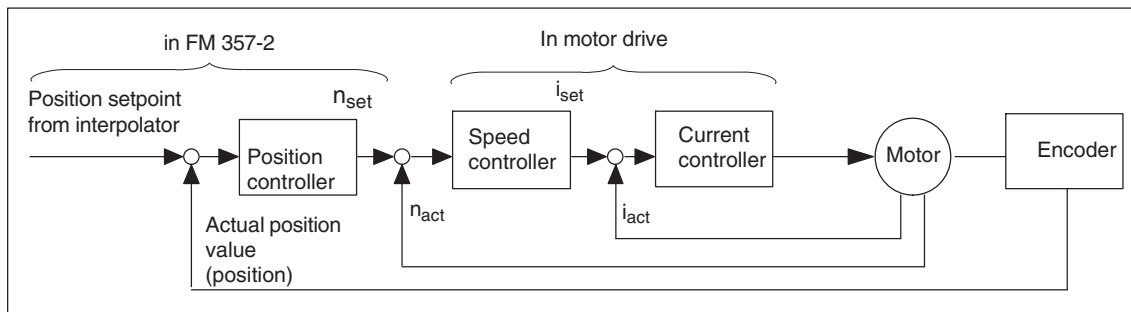


Fig. 9-2 Control loops

The closed-loop position controller performs the following tasks:

- Control of the drive at the right speed while a movement is being performed.
- Accurate approach by axis into programmed target position
- Maintenance of the axis in position in the face of interfering factors.

The position controller is configured as a proportional-action controller. In its environment are a number of function units that provide support by performing special tasks within the complex of movement control, and that can be adapted to axis conditions by means of a variety of parameters.

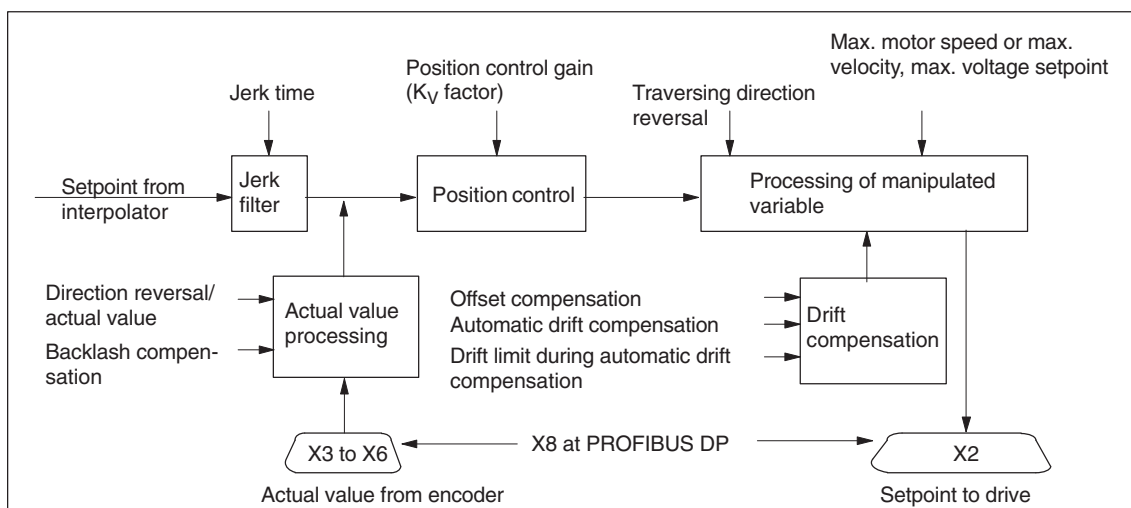


Fig. 9-3 Overview of position controller

Jerk filter

When no jerk limitation is active, acceleration and delay are applied as step changes.

The axis-specific jerk limitation on position controller level can be used during acceleration and deceleration, in order to smooth the sharp discontinuities of the ramp-shaped velocity curve. This yields particularly “soft” (jerk-free) acceleration and braking for certain positioning tasks, such as conveying of fluids.

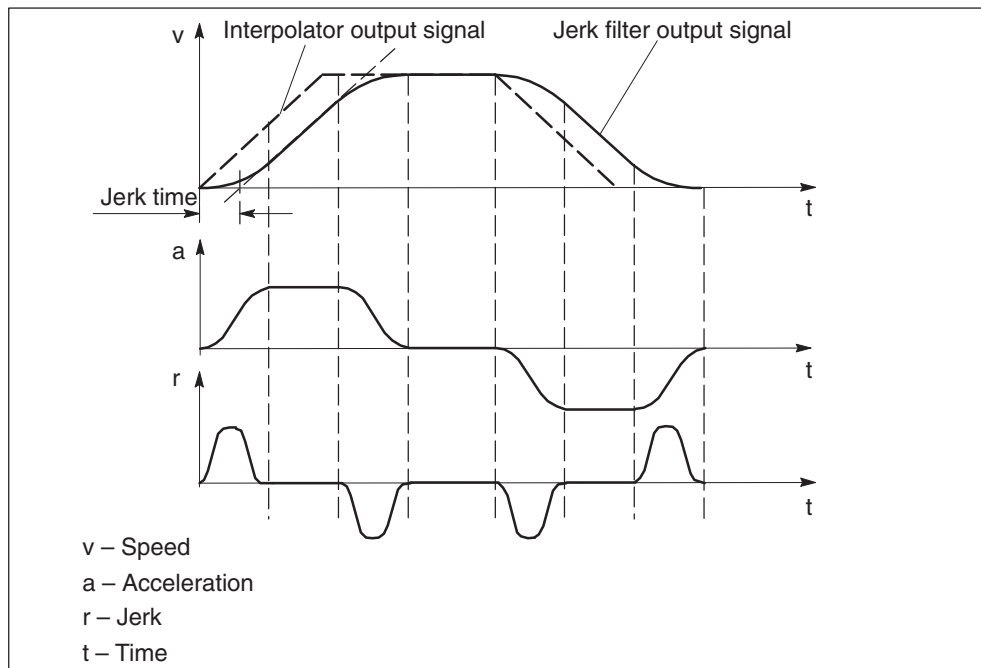


Fig. 9-4 Jerk limitation on position controller level

Parameters	Value/Meaning	Unit
Jerk filter active	no Jerk filter not active (default value) yes Jerk filter active	–
Jerk time	1 (default value) 0 to 100	[ms]

Notice

This jerk limitation acts on every axis movement in all operating modes.

The input of a jerk time reduces the effective K_v factor (contour corruption on interpolation). Allowance should be made for this in axes which are required to have the same K_v factor.

It is generally not advisable to enter values larger than approximately 20 to 30 ms with axis interpolation (because the K_v factor, and thus the contour accuracy, is reduced).

Jerk-limited acceleration (see Section 9.5) should always be used first in jerk limitation.

Direction reversal actual value

If the control direction of the position controller is reversed, you can adapt the system using the parameter “direction reversal/actual value”.

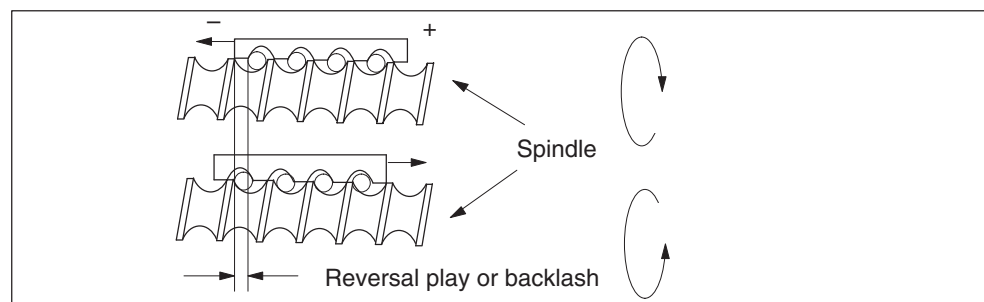
Notice

If the axis does not move in the desired direction, change the “traversing direction reversal” parameter.

Backlash compensation

When power is transmitted between a moving machine part and its drive (e.g. backlash in leadscrew), reversal errors (backlash) generally occur. These must be tolerated since setting the mechanics so as to achieve transmission without backlash would lead to excessive wear.

Backlash can also occur between the machine part and the encoder.



On axes with indirect measuring systems and stepper motors without encoders, mechanical backlash causes the corruption of the traversing path, since the axis travels too far or not far enough (the error being equal to the backlash) when the direction of travel is reversed.

In order to compensate for the backlash, the system corrects the actual value by the amount entered in the “backlash compensation” parameter each time the direction of travel is reversed. After reference point approach, the backlash compensation is active in all operating modes.

Parameters	Value/Meaning	Unit
Direction reversal/actual value	no No reversal (default value) yes Reversal	–
Backlash compensation	0 (default value) –10 000 to +10 000 Positive value: on positive backlash Negative value: on negative backlash	[μm], [10^{-3} degrees]

- **Positive backlash:**

The encoder travels ahead of the machine part (e.g. table). The table does not travel far enough, because the actual position measured by the encoder is ahead of the actual position of the table.

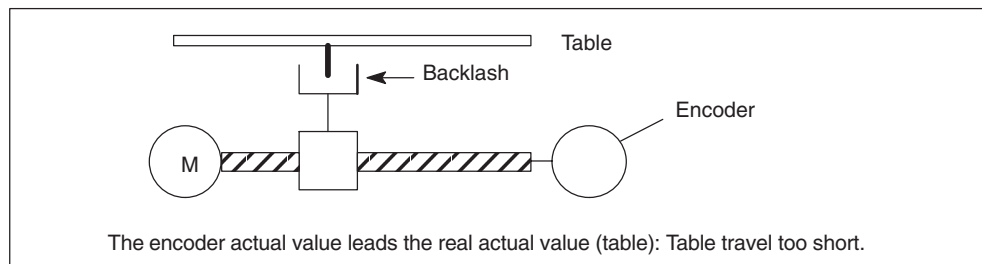


Fig. 9-5 Positive backlash (normal condition)

- **Negative backlash:**

The encoder lags behind the machine part (e.g. table); the table travels too far.

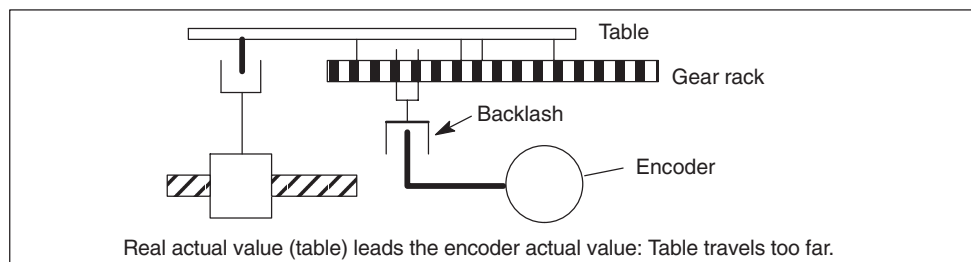


Fig. 9-6 Negative backlash

Position control gain, K_v factor

The position control gain defines the following error at any given axis traversing velocity. The mathematical (proportional) relationship is:

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

The magnitude of the K_v factor affects the following important reference variables of the axis:

- Positioning accuracy and holder control
- Uniformity of movement
- Positioning time

A high loop gain on the position controller is required in order to achieve a high level of positioning accuracy during interpolation. However, an excessive K_v factor leads to oscillations, instability and impermissibly high stress on the machine. The maximum permissible K_v factor depends on the design and dynamic response of the drive and the mechanical quality of the machine.

The following relationship applies for these characteristics:

The better the axis design, the greater the achievable K_v factor, and the better the axis parameters from the technological viewpoint. The size of the K_v factor is especially affected by the time constants, backlash and spring components in the controlled system. In real applications the K_v factor moves within the following bandwidth:

- $K_v = 0.2$ to 0.5 poor axis quality
- $K_v = 0.5$ to 1.5 good axis quality (normal condition)
- $K_v = 1.5$ to 2.5 very good axis quality

Parameters	Value/Meaning	Unit
Position control gain (K_v factor)	1 (default value) 0.1 to 100 The digit 1 must be entered for a K_v factor of 1. The conversion factor is calculated internally.	$[(10^3 \text{mm/min})/\text{mm}]$, $[(10^3 \text{deg/min})/\text{deg}]$

Notice

Axes which interact during interpolation must have the same following error at the same velocity. Allowance should be made for this in axes which are required to have the same K_v factor.

A K_v factor of between 2 and 3 must be selected for stepper motor axes.

Travel direction reversal

If the axis does not traverse in the desired direction, an adjustment can be made via parameter “Travel direction reversal”. The control direction of the position controller is calculated internally.

Parameters	Value/Meaning	Unit
Traversing direction reversal	no	No reversal (default value)
	yes	Reversal

Notice

If the control direction of the position controller is reversed, you can adapt the system using the parameter “direction reversal/actual value”.

Velocity assignment (servo drive)

In order to calculate setpoints, the control must know which maximum setpoint voltage corresponds to which maximum motor speed, and thus to which maximum velocity. This is defined in the parameters “max. voltage setpoint”, “max. motor speed” **and** “maximum velocity”.

Using these parameters, it is possible to adapt the position controller to various speed controllers and various maximum speeds.

Warning!

This assignment **MUST** be identical to the setting on the drive!

When the “max. motor speed” parameter is passed, the “Parameterize FM 357-2” tool calculates the value in the “maximum velocity” parameter according to the encoder settings (distance per spindle revolution, load gearbox) and vice-versa.

As a compromise between the highest possible resolution and adequate CL control reserve, this voltage should lie between 8 V and 9.5 V.

Parameters	Value/Meaning	Unit
Max. motor speed U_{\max} [Motor]	1 000 (default value) 1 to 999 999	[rev/min]
Maximum velocity V_{\max} [Axis]	10 000 (default value) 1 to 999 999	[mm/min], [rev/min]
Set voltage, max.	8 (default value) 0.1 to 10	[V]

Example:

With a voltage of 8 V, the drive reaches a maximum speed of 3000 rev/min. There is no load gearbox (the transmission ratio is 1:1), the distance per spindle revolution is 5 mm.

- Parameter “max. voltage setpoint” = 8 [V] (must be entered)
- Parameter “max. motor speed” = 3 000 [rev/min] (must be entered here)
- Parameter “maximum velocity” = 15 [m/min] (is calculated)

The parameters “max. motor speed” and “max. voltage setpoint” describe the physical properties of the converter and drive, and can therefore only be defined on startup.

Velocity assignment (PROFIBUS-DP drive)

Set the parameter “Max. motor speed” to the motor rated speed parameterized in the drive.

For SIMODRIVE 611-U, the scaling of the manipulated value can be read from the drive. To do so, set the parameter “Max. setpoint voltage” to zero.

If this is not possible (e.g. MASTERDRIVES) or not desired, the parameter “Max. setpoint voltage” must be set to 10 V.

If DSC is active, a path-setpoint specification (control difference) will occur. In this case, the velocity assignment is relevant for the speed feedforward control.

Velocity assignment (stepper motor)

In order to calculate setpoints, the calculate must know the maximum permissible motor speed and thus the maximum velocity. This is defined in the parameters “max. motor speed” and “maximum velocity”.

When the “max. motor speed” parameter is passed, the “Parameterize FM 357-2” tool calculates the value in the “maximum velocity” parameter according to the encoder settings (distance per spindle revolution, load gearbox) and vice-versa.

Parameters	Value/Meaning	Unit
Max. motor speed U_{\max} [Motor]	1 000 (default value) 1 to 999 999	[rev/min]
Maximum velocity V_{\max} [Axis]	10 000 (default value) 1 to 999 999	[mm/min], [rev/min]

The “Parameterize FM 357-2” tool calculates the maximum frequency from the parameter “max. motor speed” or “maximum velocity” according to the encoder settings (distance per spindle revolution, load and resolver gearbox and increments per revolution).

Offset compensation

As a result of the analog modules (FM 357-2 digital-to-analog converter and drive controller module) involved in the position control loop for **servo drives**, a zero point error occurs as a function of operating voltage and component tolerances.

This has the undesired effect that the drive motor turns when the internal speed command in the FM 357-2 is zero. But by setting a voltage offset in the offset compensation parameter, the analog system can be balanced at startup from the FM side.

Offset compensation is not required for stepper motor axes.

Parameters	Value/Meaning	Unit
Offset compensation	0 (default value) -2 000 to +2 000 The entered value is added as an additional speed setpoint and always active.	[mV]

Drift compensation/Drift limit value

Thermal conditions will shift the zero error in the control loop during operation. This behaviour is called drift. In a closed control loop with a proportional-action controller, this results in a temperature-dependent positioning error. With drift compensation, continuous balancing takes place automatically in the positioning control loop.

The value for the drift compensation is limited by the parameter “drift limit”.

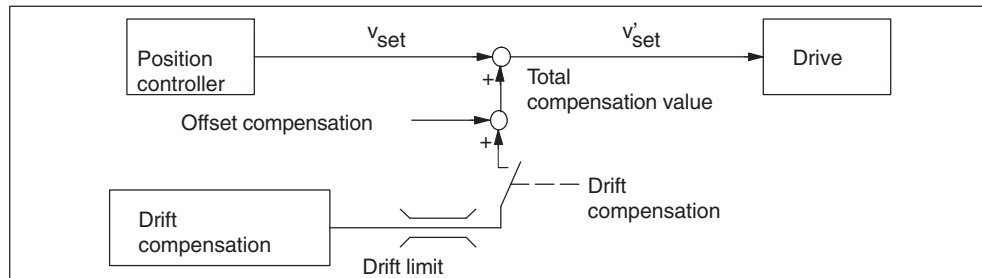


Fig. 9-7 Composition of the total compensation value

Parameters	Value/Meaning	Unit
Drift compensation	no Drift compensation off yes Drift compensation on When the drift compensation is switched on, the controller calculates the drift required to achieve a following error of 0.	—
Drift limit value	100 (default value) 0...500 If the drift value exceeds this parameter setting, an error is signalled and the drift restricted to the parameterized limit.	[mV]

Notice

The effect of drift compensation can be checked with reference to the following error displayed. When the axis is stationary, the following error displayed should be 0.

Drift compensation is not required for stepper motor axes.

Speed feedforward control

The speed feedforward control function can be applied to reduce the axial following error on servo drives to almost zero. The following error causes a velocity-dependent contour error, in particular during acceleration processes at contour curvatures.

With speed feedforward control, an additional speed setpoint is defined at the input of the speed controller.

Parameters	Value/Meaning	Unit
Speed feedforward control active	no yes (default value)	–

Time constant, speed control loop

To obtain a correct speed feedforward control setting, the time constant of the speed control loop must be defined exactly.

This can be achieved by measuring the step response of the closed speed control loop, e.g. using an analog function generator.

Parameters	Value/Meaning	Unit
Time constant, speed control loop	0.5 (default value) 0 to 10	[ms]

Weighting factor

The weighting factor determines the effect of the speed feedforward control.

With an optimally set control loop and a precisely calculated speed control loop time constant, the weighting factor will be approximately 1.

Parameters	Value/Meaning	Unit
Weighting factor	1 (default value) 0 to 10	–

Fine adjustment

By making slight changes to the parameter, it is possible to set the desired response for the relevant axis.

The axis should be traversed at a constant velocity and the control difference checked (service display of the parameterization tool).

Controller difference = 0 Setting is correct

Positive direction of travel:

Controller difference > 0 Time constant or weighting factor too **small**

Controller difference < 0 Time constant or weighting factor too **large**

Notice

A low acceleration and high velocity results in very long acceleration phases. This allows the controller difference to be read off easily.

9.4 DSC (Dynamic Servo Control)

General

For PROFIBUS-DP drives, it is possible to relocate the position controller into the drive. By minimizing the transmission dead times and calculating the position control in the speed control cycle, the dynamic response of the position control loop is enhanced so that higher loop-gain factors can be achieved.

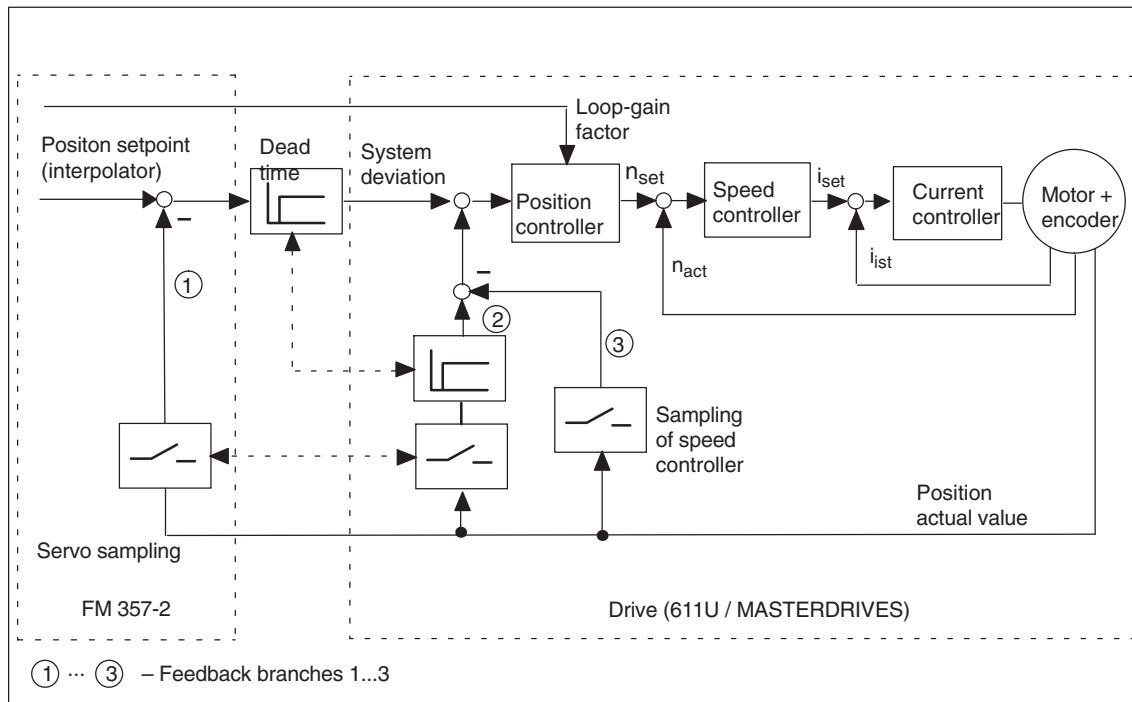


Fig. 9-8 Control-loop structure with DSC

Principle

The loop-gain factor and the calculated system deviation are transferred from the FM357-2.

In total, 3 feedback branches (see Fig. 9-8) are provided for the position actual value. Feedback branch 2 compensates the effect from feedback branch 1 in the FM completely with regard to the actual value delivered from the drive. The dead time in branch 1 has thus no influence on the stability of the position control loop. First, no position control is provided. The feedback branch 3 will close the position control loop with a correspondingly low delay (speed controller cycle); so that higher gains can be achieved.

The absolute reference of the position actual value is provided in the FM357-2. The drive operates relatively with reference to system deviation and position actual value.

Parameterization

For PROFIBUS-DP drives, this function can be activated or deactivated for each axis separately via the parameter “DSC (Dynamic Servo Control)”.

The transfer time between FM357-2 and drive must be entered in the parameter “Transfer dead time”. The DSC function requires this value to correct the actual value in the feedback branch 2 (see Fig. 9-8).

Parameter	Value/meaning	Unit
DSC (Dynamic Servo Control)	no (default value) yes The position controller for the PROFIBUS-DP axis is calculated in the drive.	–
Transfer dead time	0 (default value) – 0.02 ... 0.02 MASTERDRIVES: 0,009 611U : – 0.0015	[s]

With the transfer dead time set correctly, the following error at constant velocity must correspond to the set loop-gain factor.

You can check or determine the transfer dead time with the speed feedforward control deactivated, using the following method:

- Optimize drive and position control without DSC
- Check the following error and the loop-gain factor in the “Service data” screen (menu **Test > Service data**) at constant velocity. With the volume controller and the drive set correctly, the following will apply:
 - Parameterized loop-gain factor = loop-gain factor (calculated)
 - Following error corresponds to the parameterized loop-gain factor
 Example:
 Loop-gain factor = 1 → 1 mm following error with F = 1,000 mm/min
 - Axial following error deviation = 0
- Activate DSC, read following error from the “Service data” screen:
 - Parameterized loop-gain factor ≠ loop-gain factor (calculated)
 - Axial following error deviation ≠ 0
- Setting the transfer dead time:
 - MASTERDRIVES: 3 * DP cycle = 9 ms
 - SIMODRIVE 611U: –1 * DP cycle = – 1.5 ms
- Reading the following error from the “Service data” screen:
 - Parameterized loop-gain factor = loop-gain factor (calculated)
 - Following error corresponds to the parameterized loop-gain factor
 - Axial following error deviation = 0

Note:

Set the message frame 611U type 105 (P922 = 105) for DSC in SIMODRIVE 611-U.

For MASTERDRIVES, standard type 5 is set with execution of the script file **stdtlg5_fm_v1.ssc**.

9.5 Velocities and acceleration rates

Velocities

The following velocities can be set on the FM 357-2 for the different operating modes:

Velocity	Effective in operating mode
Maximum velocity Positioning velocity	Automatic, MDI
Axis velocity Rapid traverse override	Jog and incremental mode, relative
Acceleration (axis related)	On all traversing movements
Path acceleration	On path movements

Maximum velocity

The maximum velocity (see Section 9.3) is a limit velocity up to which an axis may be accelerated. This limitation is effective in all operating modes. The axis moves at this velocity with programmed rapid traverse (G0) in Automatic or MDI mode.

The maximum permissible velocity of an axis depends on the dynamic response of the machine and drive.

Positioning velocity

If a positioning axis is programmed in the NC program without a specific feedrate, then the feed entered in this parameter is automatically applied. This also applies analogously to the CPU axis (see Section 6.5.1).

This feedrate is valid until an axis-specific feedrate is programmed in the NC program.

Parameters	Value/Meaning	Unit
Positioning velocity	10 000 (default value) 0 to 999 999	[mm/min], [rev/min]

If a velocity of zero is entered, the positioning axis does not move without a feedrate.

If the velocity entered is greater than the maximum axis velocity of the axis, the velocity is limited to the maximum axis velocity during traversing.

Axis velocity

The velocity entered in this parameter is applied in “Jog” and “Incremental travel relative” modes.

Parameters	Value/Meaning	Unit
Axis velocity	2 000 (default value) 0 to 999 999	[mm/min], [rev/min]

If the velocity entered in the “axis velocity” parameter is greater than the value in the “maximum velocity” parameter, the maximum velocity applies.

Rapid traverse override

The velocity entered in this parameter is applied in “Jog” and “Incremental travel relative” modes with active rapid traverse.

Parameters	Value/Meaning	Unit
Rapid traverse override	10 000 (default value) 0 to 999 999	[mm/min], [rev/min]

If the value set in parameter “Rapid traverse override” is greater than the setting in parameter “Maximum velocity”, then the maximum velocity setting is valid.

Axis-specific acceleration rate

An acceleration process controlled by the interpolator as well as an acceleration pattern must be parameterized for each individual axis.

The following acceleration patterns are available:

- Brisk acceleration
- Soft acceleration
- Drive acceleration

If no special parameters are entered for the path movement, the path acceleration is made up from the parameters of the axes involved and their proportion of the path vector (geometry).

It is possible to combine axes with different acceleration characteristics.

Initial setting

The acceleration pattern to be activated for positioning movements in modes “Jog, Incremental travel relative, Reference point approach and Automatic” can be programmed for each individual axis.

The acceleration pattern of an axis can also be activated and deactivated in the NC program (see Section 10.6.3):

BRISKA(axis) Brisk acceleration
SOFTA(axis) Soft acceleration
DRIVEA(axis) Drive acceleration

Parameters	Value/Meaning	Unit
Acceleration pattern	Brisk acceleration (default value) Soft acceleration Drive acceleration	–
Initial setting	Ö Brisk acceleration	

Brisk acceleration

The movement is controlled such that there is a step change in the acceleration rate over time. At the beginning of the movement, the axis accelerates to the programmed feedrate at the rate specified in the “acceleration” parameter. Before coming to a standstill, it decelerates at the same rate.

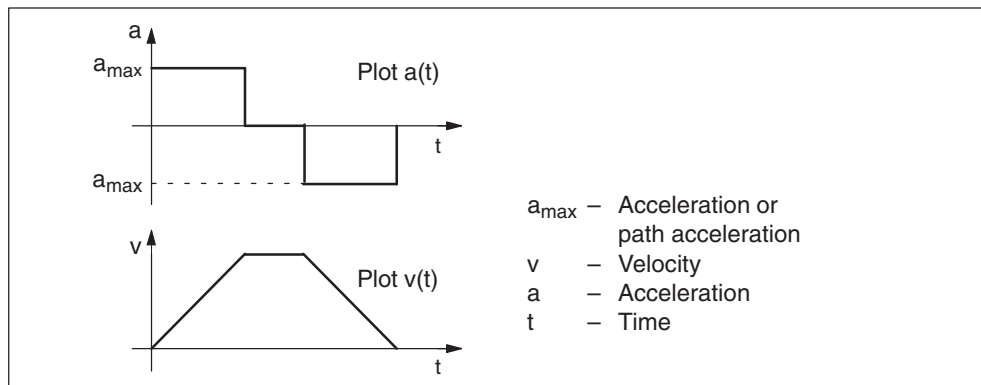


Fig. 9-9 Velocity and acceleration plotted for brisk acceleration

It is not possible to achieve smooth acceleration and deceleration of the axes with brisk acceleration, however it is possible to achieve an optimized velocity/time profile.

Parameters	Value/Meaning	Unit
Acceleration	1 (default value) 0 to 10 000	[m/s ²], [rev/s ²]

Axes can also have different accelerations. The lowest acceleration of the axes involved is used for interpolation.

Soft acceleration

With the soft acceleration pattern, the movement is controlled such that the axis setpoint characteristic remains smooth (without jerks). However, the softer acceleration characteristic increases the traversing time for the same distance, velocity and acceleration rate as brisk acceleration. It may be possible to compensate for this time loss by setting a higher acceleration.

In addition to fully utilizing the acceleration capabilities of the machine, soft acceleration also has the following advantages:

- Reduces wear on the machine's mechanical parts
- Reduces high-frequency oscillations on the machine, which are difficult to control

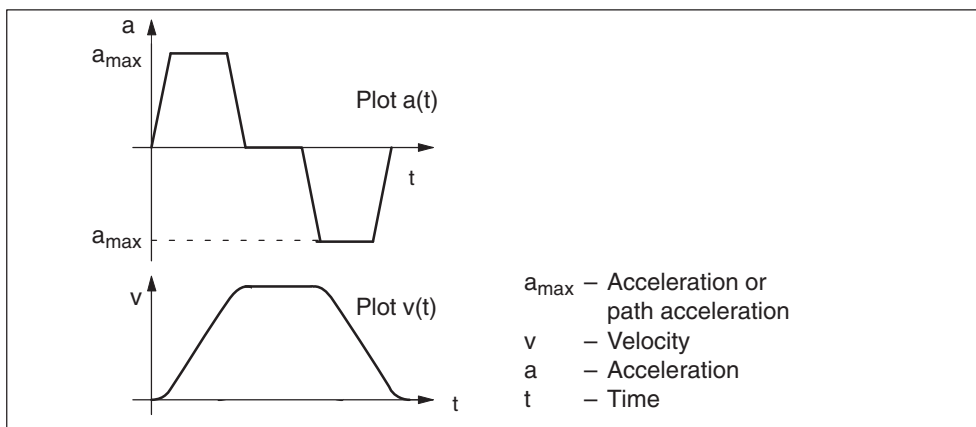


Fig. 9-10 Velocity and acceleration profiles with soft acceleration

Parameters	Value/Meaning	Unit
Acceleration	1 (default value) 0 to 10 000	[m/s ²], [rev/s ²]
Jerk	1 000 (default value) 0 to 100 000	[m/s ³] [rev/s ³]

“Jerk” is the change in acceleration rate per time unit.

Knee-shaped acceleration

A characteristic property of **stepper drives** is the drop in the available torque in the upper speed range.

You can optimize the acceleration profile while providing added protection against overloading by using a speed-dependent acceleration (knee-shaped acceleration).

The creep acceleration is valid above the defined creep velocity, while the “normal” acceleration applies below it.

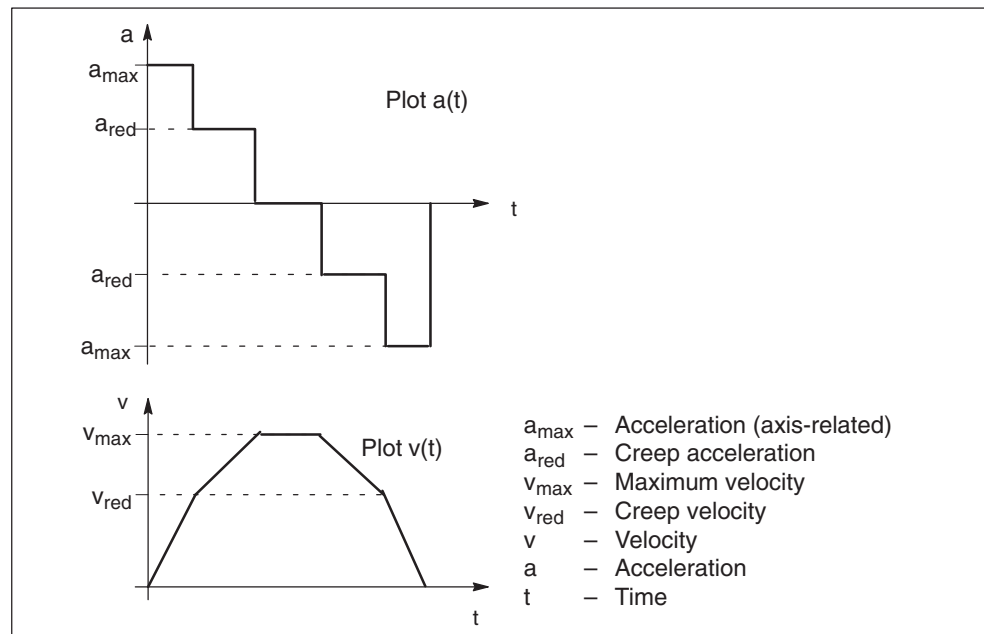


Fig. 9-11 Axial acceleration and velocity characteristic

Parameters	Value/Meaning	Unit
Acceleration	1 (default value) 0 to 10 000	[m/s ²], [rev/s ²]
Creep velocity	10 000 (default value) 0 to 999 999	[mm/min], [rev/min]
Creep acceleration	1 (default value) 0 to 10 000	[m/s ²], [rev/s ²]

Notice

Drive acceleration can only be parameterized in relation to an axis. The path response results from the calculation with the axes involved.

Path response

Axes can interpolate with one another in the “Automatic” or “MDI” modes. An additional path acceleration and path jerk can be entered for this path movement.

If no special parameters are entered for the path movement, the path acceleration is made up from the parameters of the axes involved and their proportion of the path vector (geometry).

Initial setting

The acceleration pattern to be activated with program start can be programmed for the path.

The acceleration pattern of a path can also be activated and deactivated in the NC program (see Section 10.6.3):

BRISK	Brisk acceleration
SOFT	Soft acceleration
DRIVE	Knee-shaped acceleration

These parameters can be used to define an additional limitation of the path acceleration or the path jerk over and above the value derived from the axial limitation values.

Parameters	Value/Meaning	Unit															
Path acceleration	10 (default value) 0 to 1 000	[m/s ²]															
Path jerk	100 (default value) 0 to 100 000 This jerk limits a change in the path acceleration. The path acceleration divided by the jerk limitation value produces a time in which the acceleration change takes place. Examples: <table> <thead> <tr> <th>Jerk</th> <th>Path acceleration</th> <th>Time</th> </tr> </thead> <tbody> <tr> <td>100 m/s³</td> <td>1 m/s²</td> <td>0.01 s</td> </tr> <tr> <td>100 m/s³</td> <td>2 m/s²</td> <td>0.02 s</td> </tr> <tr> <td>200 m/s³</td> <td>2 m/s²</td> <td>0.01 s</td> </tr> <tr> <td>300 m/s³</td> <td>3 m/s²</td> <td>0.01 s</td> </tr> </tbody> </table>	Jerk	Path acceleration	Time	100 m/s ³	1 m/s ²	0.01 s	100 m/s ³	2 m/s ²	0.02 s	200 m/s ³	2 m/s ²	0.01 s	300 m/s ³	3 m/s ²	0.01 s	[m/s ³]
Jerk	Path acceleration	Time															
100 m/s ³	1 m/s ²	0.01 s															
100 m/s ³	2 m/s ²	0.02 s															
200 m/s ³	2 m/s ²	0.01 s															
300 m/s ³	3 m/s ²	0.01 s															

The “Path jerk” is the change in path acceleration rate per time unit.

Notice

The limit value in the “path acceleration” parameter is only applied if the value is lower than the limit value calculated from the axis movement.

There is no path parameter for drive acceleration. The path action is determined by the axial values.

9.6 Monitoring

Overview

In this section, you can find information about:

- Monitoring of movements
- Monitoring of encoders
- Hardware and software limit switches

9.6.1 Monitoring of movements

General

The following table provides an overview of the monitoring systems.

Monitoring function	Active
Move into position	Block finished in accordance with setpoint
Following-error monitoring <ul style="list-style-type: none"> • Axis standstill • Axis movement 	Active position control In "Fine target range" after deceleration
Clamping tolerance	Interface signal "Activate terminals" (user DB, "AXy", DBX2.3+m)
Set speed	Active position control
Actual velocity	Active actual values
Monitoring of the setpoint/actual position	Controlling an axis (speed control active)

Reaction to monitoring function response

Initiation of the corresponding error message.

The affected axis is brought to a standstill with rapid stop (with open position control loop) by means of a speed setpoint ramp.

If the axis is involved in an interpolation relationship to other axes, these are brought to a standstill with rapid stop and following error reduction (definition of a position part setpoint = 0).

Move into position

In order to ensure that an axis is positioned within the specified time, the timer set in parameter “Monitoring time” is started at the end of a motion block (position part setpoint = 0 at the end of motion).

When this time expires, the system checks whether the following error is below the limit value specified in the parameter “target range coarse” (for blocks with target range coarse) or “target range fine” (for blocks with target range fine).

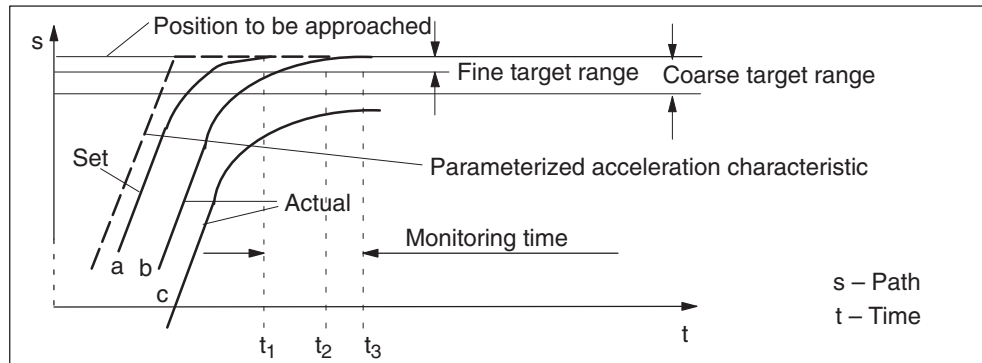


Table 9-7 Position monitoring time

Time	Position monitoring
t ₁ (a)	The monitoring time is started when the target position is reached by the interpolator.
t ₂ (b)	The actual position reaches the target range before the monitoring time expires. Positioning is complete.
t ₃ (c)	The actual position is not within the target range when the monitoring time expires (error).

Parameters	Value/Meaning	Unit
Monitoring timeout	1 (default value) 0 to 100	[s]
Target range coarse	0.04 (default value) 0 to 1 000 Must not be set smaller than the fine target range.	[mm], [degrees]
Target range fine	0.01 (default value) 0 to 1 000	[mm], [degrees]

Notice

The size of the positioning window affects the block change time. The smaller the tolerances selected, the longer the positioning process takes, and the longer it takes until the next statement in the NC program can be executed.

When the “target range fine” positioning window is reached or a new position part setpoint $\neq 0$ is output, the position monitoring system is deactivated and replaced by the zero speed control.

The positioning windows are indicated by the following interface signals:

- Position reached, stop (target range coarse) user DB, “AXy”, DBX20.6+m)
 - Position reached, stop (target range fine) user DB, “AXy”, DBX20.7+m)
-

Following error monitoring**Axis movement**

The monitoring system is intended to ensure that the contour defined by the NC program is machined within a certain tolerance band.

With following error monitoring, the measured following error and the following error calculated in advance from the position setpoint are compared, allowing for a tolerance value entered in the “following error monitoring” parameter.

Notice

The current following error deviation (axial) can be monitored in the servicing display (parameterization tool).

Axis stationary

This monitoring system has the following functionality:

- When a traversing block is completed (position part setpoint = 0 at the end of the movement), the system checks whether the following error has reached the limit specified by the “Zero speed tolerance” parameter after the time configured in the “Zero speed control delay time” parameter.
- On completion of a positioning operation (fine target range reached), the zero speed monitor is activated in place of the positioning monitor. The system checks whether the axis moves out of position by a distance specified in the “Zero speed tolerance” parameter.

The zero speed control system is also activated if:

- “Fine target range” has been reached and
- “Zero speed control delay time” is still running.

Parameters	Value/Meaning	Unit
Following error monitoring system (movement of axis)	1 (default value) 0 to 1 000	[mm], [degrees]
Zero speed control delay	0.4 (default value) 0 to 100	[s]
Zero speed range	0.2 (default value) 0 to 1 000	[mm], [degrees]

Axis stationary

The interface signal “Axis stationary” indicates whether the current velocity of the axis is above or below a limit value programmed in parameter “Threshold velocity axis stationary”.

The monitoring function is active only when setpoint zero has been reached.

Parameters	Value/Meaning	Unit
Threshold velocity for stationary axis	5 (default value) 0 to 10 000	[mm/min], [rev/min]

Clamping monitor

If the axis is to be clamped after completion of the positioning operation, the interface signal “Activate clamp” (user DB, “AXy”, DBX2.3+m) can be used to activate the clamping monitoring system.

This can be necessary, because the axis can be moved out of position, during the clamping operation, by a distance greater than the zero speed tolerance. The tolerance from the set position is specified in the parameter “Clamping tolerance”.

Parameters	Value/Meaning	Unit
Clamping tolerance	0.5 (default value) 0 to 1 000	[mm], [degrees]

Speed setpoint monitoring

The speed setpoint monitoring system is used to check whether the physical limit of the drive of a **servo axis** (10 V maximum voltage for speed setpoint on analog drives) is exceeded.

The speed setpoint is the product of the position controller speed setpoint, the speed feedforward control and the drift value from the drift compensation function.

The system also checks whether the value entered in the “Set speed” parameter is exceeded.

Parameters	Value/Meaning	Unit
Set speed	100 (default value) 0 to 100 % value referred to the maximum motor speed or maximum velocity	[%]

The set speed monitoring system can also be used for test purposes.

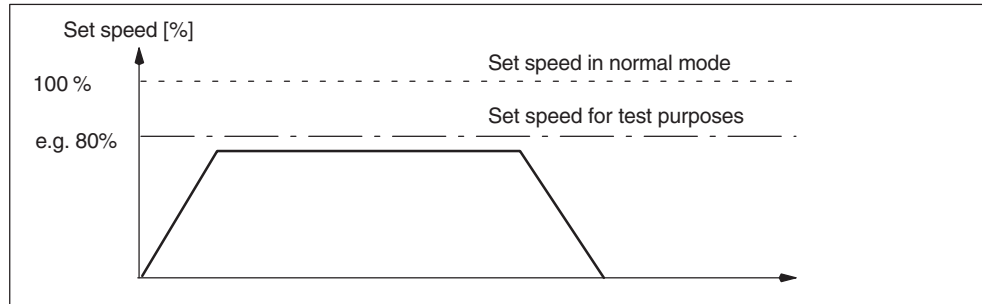


Fig. 9-12 Speed setpoint monitoring

The “Monitoring time” parameter is used to define how long the set speed is allowed to remain within the limitation range before the set speed monitoring system triggers a response.

Parameters	Value/Meaning	Unit
Monitoring time	0 (default value) 0 to 100	[s]

Notice

The limitation of the set speed makes the control loop nonlinear. This generally causes path deviations if an axis remains in the set speed limitation range for too long.

Actual velocity monitoring

This system monitors whether the actual velocity exceeds a permissible limit defined by the “actual velocity” parameter.

The monitoring system always triggers a response if the actual values returned are below the limit frequency.

Monitoring of the setpoint/actual position

Controlled traversing of an axis (traversing in speed-controlled mode) will disable the following error monitoring.

In this state, the monitoring of the setpoint position is effective. If the difference exceeds the value defined in the parameter "Setpoint/actual position tolerance", the axis is stopped and an error is output.

Parameters	Value/Meaning	Unit
Setpoint/actual position tolerance	5.0 (default value) 0...100 000 000	[mm], [deg]

Notice

You should always take into account that all specified positions, contours or limitations can only be approached or monitored with the accuracy specified in the parameter "Setpoint/actual position tolerance".

9.6.2 Encoder monitoring

Overview and features

The following table provides an overview of the monitoring systems and their features.

Monitoring system	Active	Effect on triggering of a response
Encoder limit frequency monitoring	Always	Initiation of the corresponding error message. The affected axis is brought to a standstill with rapid stop (with open position control loop) by means of a speed setpoint ramp. If the axis is involved in an interpolation relationship to other axes, these are brought to a standstill with rapid stop and following error reduction (definition of a position part setpoint = 0).
Zero marker monitoring	if activated with the "Zero marker monitoring" parameter	Initiation of the corresponding error message. The affected axis is brought to a standstill with rapid stop (with open position control loop) by means of a speed setpoint ramp. If the axis is involved in an interpolation relationship to other axes, these are brought to a standstill with rapid stop and following error reduction (definition of a position part setpoint = 0).
Stepper motor rotation monitoring	if interface signal "Rotation monitoring stepper motor" (user DB, "AXy", DBX8.0+m) is set	Interface signal "Rotation monitoring stepper motor" (user DB, "AXy", DBX29.0+m) is set. The monitoring system is deactivated automatically. It is necessary to repeat reference point approach.

Encoder limit frequency monitoring

If the permissible limit frequency of a measuring system entered in parameter "Encoder limit frequency" is exceeded, the machine and control system fall out of synchronization. The affected axis must be referenced again. This status is reported to the CPU by the interface signal "Encoder limit frequency exceeded".

Parameters	Value/Meaning	Unit
Encoder limit frequency	300 000 (default value) 0 to 1 500 000 0...9 * 10 ²⁹⁹ (software version 5 and higher)	[Hz]

Zero marker monitoring

The zero marker monitor checks whether pulses have been lost between two zero marker crossings of the position actual value encoder. The zero marker monitor is activated in parameter “Zero marker monitor”. The number of detected zero mark errors at which the monitor must respond is also specified.

Parameters	Value/Meaning	Unit
Zero marker monitoring	OFF: HW encoder monitoring ON (default value) OFF: HW encoder monitoring OFF ON: 1 to 99 or 101 to 10 000 Number of detected zero marker errors	–

When the monitoring system is activated, counting of the zero markers begins with “0”.

Notice

“OFF: HW encoder monitoring OFF” will skip all encoder errors and error reactions. The measured values will not apply in case of error. Under certain circumstances, this can result in subsequent errors (e.g. error no.: 25050 Following error monitoring) if one tries to traverse the axis.

Rotation monitoring Stepper motor

The proximity switch used for rotation monitoring is connected in the same way as when referencing with a proximity switch (see Section 9.7.2).

The proximity switch used for referencing can also be used for rotation monitoring. In this case, however, the rotation monitoring system must be deactivated during referencing.

The rotation monitoring system is activated/deactivated with the interface signal "Rotation monitoring stepper motor" (user DB, "AXy", DBX8.0+m).

The increments between two proximity switch signal edges are defined in the parameter "Number of increments". The tolerance in the parameter "Increment tolerance" parameter is allowed when comparing increments.

If a change in direction takes place between two BERO edges, the internal increment counter is cleared, i.e. the monitoring system is only active if at least two BERO edges are crossed in the same direction.

Parameters	Value/Meaning	Unit
Number of increments	2 000 (default value) 10 to 1 000 000	–
Increment tolerance	50 (default value) 10 to number of increments	–

Notice

The "Rotation monitor error" also occurs if the stepper motor, for example, is incorrectly started, even if the rotation monitor is not active. The user is responsible for ensuring that the drive is shut down safely.

"Rotation monitoring error" means shut down the drive!

9.6.3 Hardware and software limit switches

General

Possible limit switch monitors:

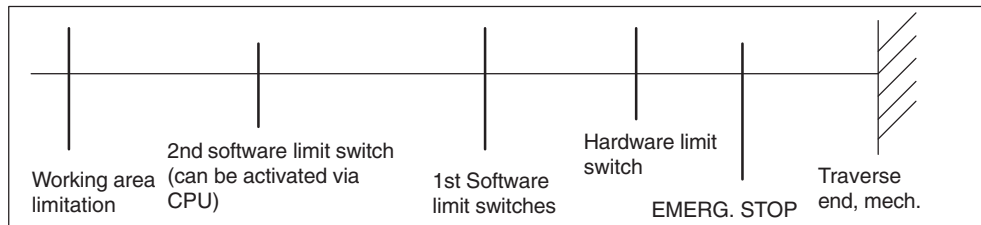


Fig. 9-13 Travel limits

The following table provides an overview of the monitoring systems and their features.

Table 9-8 Features of monitoring systems for static limits

Name	Active	Effect on triggering of a response
Hardware limit switch	After controller power-up in all operating modes	Initiation of the corresponding error message. The axis is brought to a standstill with rapid deceleration (definition of setpoint = 0) and reduction of the following error. If the axis is involved in an interpolation relationship to other axes, these are brought to a standstill in the same way. The direction keys are disabled in the approach direction.
Software limit switch	After referencing in all operating modes	Initiation of the corresponding error message. Automatic mode: The block which would violate the software limit switch is not started. The previous block is not terminated properly. Jog, incremental relative mode: The axis stops at the position of the software limit switch. When the monitoring system kicks in, the axes are decelerated. If an axis is involved in an interpolation relationship to other axes, these are decelerated in the same way. A contour violation may occur. Program execution is interrupted. The direction keys are disabled in the approach direction.

Hardware limit switch

A hardware limit switch is provided for each direction of travel of each axis. If the hardware limit switch is crossed, the CPU sends a corresponding message to the FM 357-2 via interface signal "Hardware limit switch plus/minus" (user DB, "AXy", DBX7.1/0+m) and the motion of all axes is stopped.

Software limit switches

They act as limitations of the maximum traversing range of each individual axis.

Two pairs of software limit switches are provided for each machine axis. These are defined by the following parameters in the machine axis system:

Parameters	Value/Meaning	Unit
1st software limit switch plus	100 000 000 (default value) –100 000 000 to +100 000 000	[mm], [degrees]
1st software limit switch minus	–100 000 000 (default value) –100 000 000 to +100 000 000	[mm], [degrees]
2nd software limit switch plus	100 000 000 (default value) –100 000 000 to +100 000 000	[mm], [degrees]
2nd software limit switch minus	–100 000 000 (default value) –100 000 000 to +100 000 000	[mm], [degrees]

The 2nd software limit switch can be activated from the CPU with the “2nd software limit switch plus/minus” (user DB, “AXy”, DBX7.3/2+m). This enables, e.g. the working area to be reduced. The change is effective immediately. The 1st software limit switch plus/minus is then no longer active.

The software limit switch monitoring system is not active with rotary axes.

Working area limitation

You can define for each axis whether or not the monitoring system is active with working area limitation (WALION).

Parameters	Value/Meaning	Unit
Working area limitation plus	Off: (default value) Working area limitation is not effective for this axis On: Working area limitation is effective for this axis	–
Working area limitation minus	Off: (default value) Working area limitation is not effective for this axis On: Working area limitation is effective for this axis	–

9.7 Referencing and alignment

General

In order to ensure that the control system knows the exact machine zero after power ON, the encoder of the axis must be synchronized with the control. This operation is known as reference point approach for incremental encoders and alignment for absolute encoders.

Notice

The following monitoring systems have no effect on a machine axis which has not been referenced or aligned:

- Working area limitation
 - Software limit switches
 - Protection zones
-

Starting the reference point approach

The reference point approach can be started for each machine axis in “Reference point approach” mode by way of interface signal “Direction plus” or “Direction minus” (user DB, “AXy”, DBX4.7/6+m), depending on the setting in parameter “Reference point approach direction”. All axes can be referenced simultaneously.

If the machine axes are to be referenced in a specific order, the following options are available:

- The user follows the order manually when referencing is started.
- The order is defined by programming the start signal accordingly in the user program.

Start without reference point approach

NC programs are started as a function of the setting in parameter “Start without reference point approach”. All axes must normally be referenced before the program is started. This condition does not have to be met when testing, e.g. simulation.

Parameters	Value/Meaning	Unit
Start without reference point approach	<p>No (default value) All axes which need to be referenced must be referenced/synchronized before an NC program can start.</p> <p>Yes It is possible to start NC programs even if one or more axes which need to be referenced have not yet been referenced/synchronized (e.g. in test mode).</p>	—

Reference point approach necessary

Parameter “Reference point approach necessary” defines for each axis individually whether or not a reference point approach is needed.

An NC program can be executed only if all axes designated as requiring a reference point approach have reached their reference or parameter “NC Start without referencing” has been set.

Parameters	Value/Meaning	Unit
Reference point approach necessary	<p>Yes (default value) This axis needs to be referenced.</p> <p>No This axis does not need to be referenced (e.g. in test mode).</p>	—

Interface signals

Interface signal “Reset” (user DB, “FMx”, DBX108.7+n) aborts the referencing operation. Any axes which have not reached their reference point by this time are not referenced. An appropriate error is displayed.

The following interface signal indicates whether an axis is referenced “Synchronized/referenced” (user DB, “AXy”, DBX20.4+m).

9.7.1 Referencing with incremental encoders

General

In the case of incremental encoders, an unpredictable offset exists between the internal FM position value and the mechanical position of the axis after power ON. To establish the position reference, the value internal to the FM must be synchronized with the real position value of the axis. Synchronization is performed by taking over a position value at a known point of the axis.

Axis with/without reference point switch (RPS)

The following methods can be used to reference incremental encoders:

Parameters	Value/Meaning	Unit
Axis with reference point switch	Yes (default value) Referencing with reference point switch	—
	No Referencing without reference point switch	

Referencing with reference point switch

When an RPS is used for referencing, synchronization is achieved in the following way:

- Travel to reference point switch (RPS)
- Synchronize with zero pulse
- Travel to reference point

Mounting a reference point switch:

The reference point switch (RPS) must be connected to a digital input. The connection of the signal (I...) to the interface signal "Reference point approach delay" (user DB, "AXy", DBX7.7+m) must be programmed in the user program.

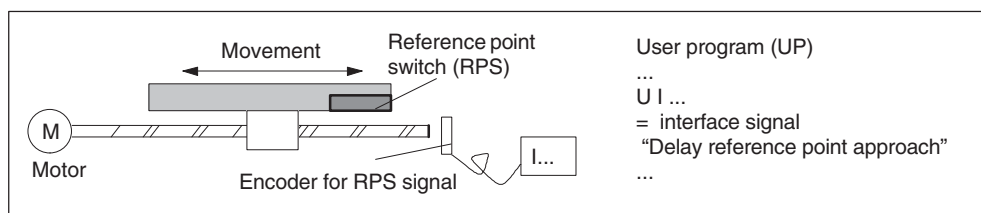


Fig. 9-14 Mounting a reference point switch (RPS)

The reference point switch must be mounted such that it extends to the end of the traversing range.

Reference point switch alignment

If the encoder has several zero pulses which repeat at cyclic intervals (e.g. incremental rotary encoder), then the reference point cam must be aligned exactly.

Practice has shown that the RPS signal edge required for synchronization should be aligned exactly between two zero pulses.

The following factors affect the time required by the controller to detect the reference point switch:

- Accuracy of the reference point switch
- Time delay at the input, cycle time, ...

The parameter "Distance RPS to zero mark/BERO" is redetermined by the FM with each referencing. You can use this value for adjusting the RPS (reference point switch).

Notice

If the RPS is not aligned exactly, an incorrect zero pulse can be evaluated. This causes the controller to assume an incorrect machine zero and move the axes to incorrect positions. All software limits then act on incorrect positions and are thus incapable of protecting the machine.

What is the minimum length of the reference point switch (RPS)?

The RPS must be so long that when the switch is approached at the referencing velocity, the deceleration process brings the axis to a standstill at the switch. The axis must leave the switch again when retracted at the creep velocity (departure at constant velocity).

In order to calculate the minimum length, the larger of the following velocities must be used in the formula:

$$\text{Minimum length} = \frac{(\text{Referencing velocity or creep velocity})^2}{2 * \text{Axis acceleration}}$$

Referencing without reference point switch (RPS)

A machine axis does not require a reference point switch if it has only one zero pulse (e.g. in the case of a rotary axis) over its entire traversing range.

For referencing without an RPS, the synchronization is performed as follows:

- Synchronize with zero pulse
- Travel to reference point

Parameters for referencing

The following table describes all parameters required to reference an incremental encoder:

Table 9-9 Parameters for referencing

Parameters	Value/Meaning	Unit
Referencing in follow-up mode	no (default value) yes The axis is synchronized in the follow-up mode with overtraveling the zero mark.	–
Reference point approach direction	Plus (default value) Minus The reference point approach is started with the travel key for the selected direction. If the axis is before the RPS when you start referencing, it accelerates to the referencing velocity and moves in the defined direction. If it is on the RPS, the axis accelerates to the creep velocity and travels in the opposite direction to the one defined. Axis may not be started after the RPS.	–
Zero marker/BERO	In front of RPS (default value) Behind/on RPS Specifies whether the zero pulse or BERO for synchronization is in front of or behind/on the RPS.	–
Reference point coordinate	0 (default value) –100 000 to +100 000 The controller uses this position as the new reference position after it reaches the reference point.	[mm], [degrees]
Reference-point shift	–2 (default value) –100 000 to +100 000 When the zero pulse has been detected, the axis moves in the defined direction across the distance entered in this parameter. The end position reached corresponds to the reference point (the reference point coordinate is set at this position).	[mm], [degrees]
Maximum distance to RPS	10 000 (default value) 0 to 100 000 If the distance traveled by the axis from the starting position towards the reference point switch is greater than the distance specified in this parameter, the axis stops with an error message.	[mm], [degrees]
Max. distance to zero marker/BERO	20 (default value) 0 to 10 000 The parameter must be smaller than the distance between 2 zero markers or 2 BERO signals, in order to ensure that the first zero marker or the first BERO signal is used for synchronization. If the distance traveled by the axis from the reference point switch is greater than the distance set in this parameter without synchronization taking place, the axis stops with an error message.	[mm], [degrees]

Table 9-9 Parameters for referencing, continued

Parameters	Value/Meaning	Unit
Distance from RPS to zero mark/BERO	0 (default value) 0...10 000 Distance measured by the FM between reference-point switch and zero mark/BERO.	[mm], [deg]
Referencing velocity	5 000 (default value) 0 to 999 999 The axis travels at this velocity towards the reference point switch (RPS).	[mm/min], [rev/min]
Creep velocity	300 (default value) 0 to 999 999 The zero pulse or BERO is approached at this velocity.	[mm/min], [rev/min]
Creep velocity tolerance	10 (default value) 0 to 100 The system monitors whether the creep velocity has been reached. The parameter can be used to specify a tolerance. An excessive discrepancy in the creep velocity reduces the referencing accuracy.	[%]
Approach velocity	1 000 (default value) 0 to 999 999 The axis travels at this velocity between synchronization with the first zero pulse or BERO signal and arrival at the reference point.	[mm/min], [rev/min]

Sequence of motions, axis in position control

The following table lists the sequence of operations involved in referencing with or without a reference point switch.

Type of referencing	Zero pulse	Sequence of movements
Axis with RPS	Zero marker/ BERO before RPS	
	Zero marker/ BERO after/at RPS	
Axis without RPS	—	
V_A – Referencing velocity V_R – Creep velocity V_E – Approach velocity		R_V – Reference point offset R_K – Reference point coordinate

Response during reference point approach, axis in position control

Travel to reference point switch

- The feedrate override and feed hold are active.
- The axis can be stopped/started with Stop/Start.
- If the axis does not stop on the reference point switch, e.g. because the RPS is too short or the referencing velocity too high, then an appropriate error message is displayed.
- If the axis is already positioned on the RPS switch, the zero marker/BERO is approached immediately.

Travel to zero marker/BERO

- The feedrate override is not active. A feedrate override of 100 % is valid. With a feedrate override of 0 %, the operation is canceled.
- The feed hold is active; the axis remains stationary and an appropriate error is output.
- The axis cannot be stopped/started with Stop/Start.

Travel to reference point

- The feedrate override and feed hold are active.
- The axis can be stopped/started with Stop/Start.
- If the reference point offset is smaller than the deceleration path of the axis from the approach velocity to standstill, the reference point is approached from the other direction.

Referencing in the follow-up mode

If the parameter “Referencing in follow-up mode is set” and the axis is in the “Follow-up” state, external referencing is possible.

The time when starting the referencing process is the beginning of motion of the axis. Referencing is only possible without RPS. Without RPS, the next zero pulse after the motion has been started is interpreted as a synchronism mark.

To obtain reproduceable results, the search for the zero mark should always be carried out either at the same velocity or at a lower velocity.

If the “Follow-up” state is canceled during the referencing process, the search for the zero mark is canceled. It is also possible to cancel a referencing process currently running with Reset.

For an axis already referenced, the search for the zero mark is suppressed in the follow-up mode.

9.7.2 Referencing with stepper motors without encoders

General

The reference point approach for stepper motors without encoder and the associated parameter setting options are similar to the referencing of incremental encoders.

Instead of the zero pulse used on incremental encoders, a reference point BERO (proximity switch) is required in this case. This is connected to a digital input of the controller.

Connection of reference point BERO

To allow connection of a reference point BERO for each axis, digital inputs are provided on the FM 357-2 (see Section 4.8):

- X1, pin 22 for BERO of axis 1
- X1, pin 23 for BERO of axis 2
- X1, pin 24 for BERO of axis 3
- X1, pin 25 for BERO of axis 4

Chronological sequence

The chronological sequence of the reference point approach for stepper motors without encoder is divided into the following phases:

- Travel to reference point switch (RPS)
- Synchronization with reference point BERO (simulator of zero marker)
- Travel to reference point

Parameters

To parameterize the reference point approach for stepper motors without encoder, the same parameters are provided as for the referencing of incremental encoders. The following parameters are available additionally:

Parameters	Value/Meaning	Unit
BERO edge evaluation	<p>1-edge evaluation (default value): The positive edge of the BERO is interpreted as a zero pulse.</p> <p>2-edge evaluation: The mean position between the positive and negative edge of the BERO is interpreted as a zero pulse. This evaluation can compensate for a possible drift. The time period between the two edges, including any possible BERO operating delay, must be greater than one position control cycle.</p>	–

Notice

With stepper motors without encoder, EMERGENCY STOP (user DB "FMx", DBX20.1) will reset the reference.

9.7.3 Alignment with absolute encoders

General

On axes with absolute encoders, the offset between the machine zero and encoder zero is measured once during start-up and then entered, i.e. the axis is aligned.

You will need to repeat the alignment:

- if the offset value is lost as a result of a battery failure
- after loading offline machine data or a firmware update
- if the mechanical connection between the encoder and the load was separated and not joined in the exact position.

Notice

The controller cannot detect all cases where it is necessary to repeat the encoder alignment!

When loading offline machine data into the FM, you can choose as from software version 5 whether the current encoder position, the status of the encoder adjustment, the adjustment offset and/or the adjustment actual value (prior to PowerOff) is to be transferred.

Parameters for encoder alignment

The parameters for aligning absolute encoders are described in the following table.

Parameters	Value/Meaning	Unit
Traversing direction key	Minus direction (default value) Plus direction The encoder is aligned with a known position in this direction.	–
Encoder alignment status	Not aligned (default value) Enabled Aligned	–
Actual value (alignment value)	0 (default value) –100 000 to +100 000 This parameter specifies the position at which the axis should be located at a known position.	[mm], [degrees]
Adjustment offset	0 (default value) 0...100 000 000 displays the offset between encoder zero and machine coordinate system zero after encoder adjustment	[mm], [deg]
Current encoder position	0 (default value) 0...100 000 000 displays the current, absolute encoder position in encoder increments With drives connected to PROFIBUS DP with absolute encoder, the internal resolution is accepted by the drive (EnDat encoder: resolution 2^{22}).	–

Encoder alignment and disadjustment procedure

Basic procedure for encoder alignment:

The alignment must be made in online mode.

The axis to be aligned is moved to a defined position and the corresponding actual value for encoder alignment then set.

1. Move the axis in "Jog" mode to a known position. The direction of approach must match the direction specified in the parameter "Traversing direction key".

Notice

This known position must always be approached at low velocity from a defined direction, in order to prevent corruption of the position through existing backlash.

2. Enter the actual value corresponding to the approached position.

The value can be defined by design characteristics (e.g. the position of a fixed stop) or can be measured using instrumentation.

3. Set encoder alignment status to "Enabled".
4. Activate the values you have entered by selecting menu icons.
5. Select "Reference point approach" mode.
6. Press the travel direction key in point 1 (the axis does not move).

The "Encoder alignment status" parameter is set internally to "Aligned". The entered value appears in the actual-value display.

7. Refresh the encoder alignment status display.

Principal procedure for encoder disadjustment:

1. Set operating mode "Reference point approach"
2. Actuating "Cancel encoder adjustment"
3. Activation by "Transferring/activating data"
4. Reset signal "Synchronized, referenced" (user DB "AXy", DBX20.4+m) by FM restart or by FM switching on/off.

9.8 Event-controlled program calls

General

With software version 5 and higher, an NC program can be started and executed implicitly for certain events. It is thus possible, e.g. to carry out initial settings or initializations.

Path and program name are fixed:

- Path: NC programs/**special programs**
- Program name: **PROG_EVENT.SPF**

You can activate the following call events via parameterization:

Parameter	Value/meaning	Unit
Start	no: no call (default value) yes: with start, call of PROG_EVENT.SPF	–
End of program	no: no call (default value) yes: with the end of the program, call of PROG_EVENT.SPF	–
Reset	no: no call (default value) yes: with reset, call of PROG_EVENT.SPF	–
FM restart	no: no call (default value) yes: after FM restart, call of PROG_EVENT.SPF	–

System variable \$P_PROG_EVENT

Since for the different events the same program PROG_EVENT.SPF is called, the call event can be requested via the system variable \$P_PROG_EVENT.

Values for \$P_PROG_EVENT depending on the call event:

Call event	\$P_PROG_EVENT value
Start	1
End of program	2
Reset	3
FM restart	4

Sequence when starting

- Initial state: AUTOMATIC or MDI mode selected; channel is in the Reset status (no program active)
- **Start** of the selected NC program
- The FM carries out the parameterized default settings.
- Implied subroutine call of PROG_EVENT.SPF with \$P_PROG_EVENT = 1
- Call of the selected NC program.

Sequence when ending the program

- Initial state: AUTOMATIC or MDI mode selected; an NC program is executed in the channel
- NC block containing **M30** or **M2** (end of program) is executed.
- The FM carries out the initial settings after end of program or reset
- Implied ASUB call of PROG_EVENT.SPF with \$P_PROG_EVENT = 2 (ASUB = asynchronous subroutine)
- The FM carries out the initial settings after end of program or reset

Sequence in case of reset

- Initial state: any mode, any channel state
- Trigger **reset**
- The FM carries out the initial settings after end of program or reset.
- Implied ASUB call of PROG_EVENT.SPF with \$P_PROG_EVENT = 3
- The FM carries out the initial settings after end of program or reset.

Sequence at FM restart

- Initial state: FM restart
- The FM carries out the initial settings after FM restart.
- Implied ASUB call of PROG_EVENT.SPF with \$P_PROG_EVENT = 4
- The FM carries out the initial settings after end of program or reset.

Subroutine PROG_EVENT.SPF

PROG_EVENT.SPF is executed as a subroutine and must be ended with M30 or RET.

If PROG_EVENT.SPF is started by the call event FM Restart, the program **must** be ended with M30 and the M command be acknowledged.

If after FM restart or Reset an error is present and if this prevent the program execution, PROG_EVENT.SPF is carried out only after the error has been eliminated.

If PROG_EVENT.SPF is not yet created and a call event is activated, the following is carried out:

- The error message 16 941 (channel ... action ... denied, since no program event has been processed) is issued after FM restart or after reset with start.
- The error message 14 011 (channel ... block ... program ... not existing or is edited) is issued after start or end of program.

Blocking (deadlock) in case of the call event RESET

The following status can be eliminated by power-up with default values:

- Reset is parameterized as the call event.
- PROG_EVENT.SPF contains a programming error.
- Reset is required to acknowledge the error.

In this state, the following is carried out:

- The error is acknowledged with reset, but it is recreated once again immediately.
- A program correction is prevented, since no transfer is possible if errors are present.
- A reparameterization of the call events is prevented because no transfer of the machine data is possible if errors are present.

For this reason, to test PROG_EVENT.SPF you should **not** use RESET as the call event.

System variable \$MC_CHAN_NAME

PROG_EVENT.SPF is executed generally in the channel in which the selected event has occurred. Since the same call event can be activated in several channels, it can be detected by requesting the system variable \$MC_CHAN_NAME in which channel the program PROG_EVENT.SPF is currently running.

Example:

```
N10 IF $MC_CHAN_NAME == "CHAN1"  
    ; Program is run in channel 1  
    N20 ...  
    ...  
N100 ENDIF
```

Statement DISPLOF

To switch off the block display for PROG_EVENT.SPF, you can use the statement DISPLOF.

Example:

```
PROC PROG_EVENT DISPLOF  
...  
; No block display is provided up to M30 or RET  
...
```

Example for PROG_EVENT.SPF

There are 3 active channels.

In channel 1, static synchronized actions are to be activated after FM restart.
In channel 2, the Z axis is to traverse to position Z50 at the end of the program.
In channel 3, various default settings are to be carried out when starting.
The appropriate call events are set via parameterization.

```
PROC PROG_EVENT DISPLOF          ; Execution without block display
```

```
N10 IF $P_PROG_EVENT == 1        ; Event: Start ?
```

```
    N1010 IF $MC_CHAN_NAME == "CHAN2"      ; 2nd channel ?
```

```
        ; Make default settings
```

```
        N1011 R10=4 R11=6 R12=8
```

```
        N1012 G18 G1 F2000
```

```
    N1090 ENDIF
```

```
N20 ENDIF
```

```
;
```

```
N30 IF $P_PROG_EVENT == 2        ; Event: End of program ?
```

```
    N3010 IF $MC_CHAN_NAME == "CHAN3"      ; 3rd channel ?
```

```
        ; Traverse Z axis to position Z50
```

```
        N3011 G0 Z50
```

```
    N3090 ENDIF
```

```
N40 ENDIF
```

```
;
```

```
N50 IF $P_PROG_EVENT == 4        ; Event: FM restart ?
```

```
    N5010 IF $MC_CHAN_NAME == "CHAN1"      ; 1st channel ?
```

```
        ; Activate static synchronized actions
```

```
        N5011 IDS=1 EVERY $AA_IW[Y]> 100 DO $A_OUT[1]=1
```

```
        N5012 IDS=2 EVERY $AA_IW[Y]<= 100 DO $A_OUT[1]=0
```

```
    N5090 ENDIF
```

```
N60 ENDIF
```

```
;
```

```
N100 M30 ; RET is not permitted here; call event "FM Restart" is used
```

9.9 Output of M, T and H functions

General

The M, T and H functions programmed in the NC program (see Section 10) are output to the interface. These signals are available in the user program for programming.

M function

With the output of M functions, a variety of switching operations can be executed on the machine via the user program (UP).

Output options

Predefined M functions are output after the movement.

You can parameterize the time of output for the free M functions in blocks with movement.

Parameters	Value/Meaning	Unit
Output options for M functions	Output prior to movement (default value) Output during movement Output after movement	–

Interface signals:

The following interface signals are available for M functions:

- Interface signals as checkback signals
 - Change auxiliary function (user DB, “FMx”, DBX127.0+n)
 - M functions (decoded) (user DB, “FMx”, DBB140...152+n)
M function number 1...5 (user DB, “FMx”, DBB158...162+n)
- Interface signals as control signals
 - Acknowledge auxiliary function (user DB, “FMx”, DBX109.0+n)

Notice

Please note that auxiliary functions which have been output (including M02 and M30) must be acknowledged with the “acknowledge auxiliary function” signal. Auxiliary functions must also be acknowledged on a “Reset” or program abort.

The output of the auxiliary functions is not synchronized with the output of the checkback signals of the axis, e.g. “Target area fine”.

T function

The output of a T function notifies the UP which tool, and thus which tool offset, must be selected.

Output options

T functions are output before the movement.

Interface signals:

- Change auxiliary function (user DB, "FMx", DBX127.0+n)
- T function number (user DB, "FMx", DBW164+n)

H function

H functions can be output to initiate switching functions on the machine or transfer data from the NC program to the UP.

Output options

You can parameterize the time of output for H functions in blocks with movement.

Table 9-10 H function parameters **without** group assignment

Parameters	Value/Meaning	Unit
Output options for H functions	Output prior to movement (default value) Output during movement Output after movement	–

Interface signals:

- Change auxiliary function (user DB, "FMx", DBX127.0+n)
- H function number 1 (user DB, "FMx", DBW166+n)
- H function number 2 (user DB, "FMx", DBW172+n)
- H function number 3 (user DB, "FMx", DBW178+n)
- H function number 1 (user DB, "FMx", DBD168+n)
- H function number 2 (user DB, "FMx", DBD174+n)
- H function number 3 (user DB, "FMx", DBD180+n)

Block change

A block is considered to be ended when the programmed motion has been executed and the auxiliary function acknowledged. NC program execution waits if necessary, in order to ensure that no auxiliary functions are lost from the viewpoint of the user program.

Continuous-path mode

A path motion remains continuous only if the auxiliary function is output during the motion and acknowledged before the path end is reached.

Example of output of M, T and H functions

Parameterized output option:

Unassigned M functions: During the movement

H functions: After the movement

N10 G01 X100 M22 H7=1 T5

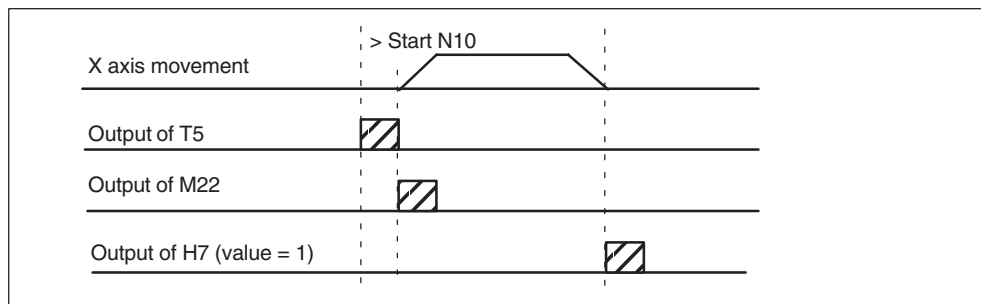


Fig. 9-15 Example for output of M, T and H functions

9.10 Inputs/outputs

General

You can use the following types of I/Os on the FM 357-2:

Table 9-11 I/Os on the FM 357-2

Type	Inputs		Outputs	
	Number	Function	Number	Function
On-board I/Os (see Section 9.10.1)	2	Measurement (probes 1 and 2)	8	used freely
	4	for BERO signal		
	12	used freely		
I/Os over local P bus digital (see Section 9.10.2) analog (see Section 9.8.3)	16	free: implemented with signal modules (SMs) on the local P bus	16	free: implemented with signal modules (SMs) on the local P bus
	8		8	

9.10.1 Digital on-board I/Os

Sensing probe inputs (X1 pins 26 and 27)

Measuring pulse inputs 1 and 2 (see Section 9.17).

In the measuring function, these two inputs are used for the connection of sensing probes.

Note

These inputs are used internally by the FM 357-2.

BERO inputs (X1 pins 22...25)

Use as a BERO input for axis 1 to 4 (see Section 9.7)

On axes which support the use of a stepper motor without encoder, a proximity switch can be connected to this input. The signal is used to reference this axis.

Note

These inputs are used internally by the FM 357-2.

Outputs (X1 pins 2, 4, 6, 8, 13, 15, 17, 19)

The signal state of these outputs can be set or read by the NC program or synchronized actions.

Read/write: \$A_OUT[n] n = Number of output
 X1 Pin 2 = Output 1
 X1 Pin 4 = Output 2
 X1 Pin 6 = Output 3
 X1 Pin 8 = Output 4
 X1 Pin 13 = Output 5
 X1 Pin 15 = Output 6
 X1 Pin 17 = Output 7
 X1 Pin 19 = Output 8

Example:

\$A_OUT[3] = R1 ; The contents of R1 (1 or 0) are
 ; output to output 3.

The signals can be set and read by the user program.

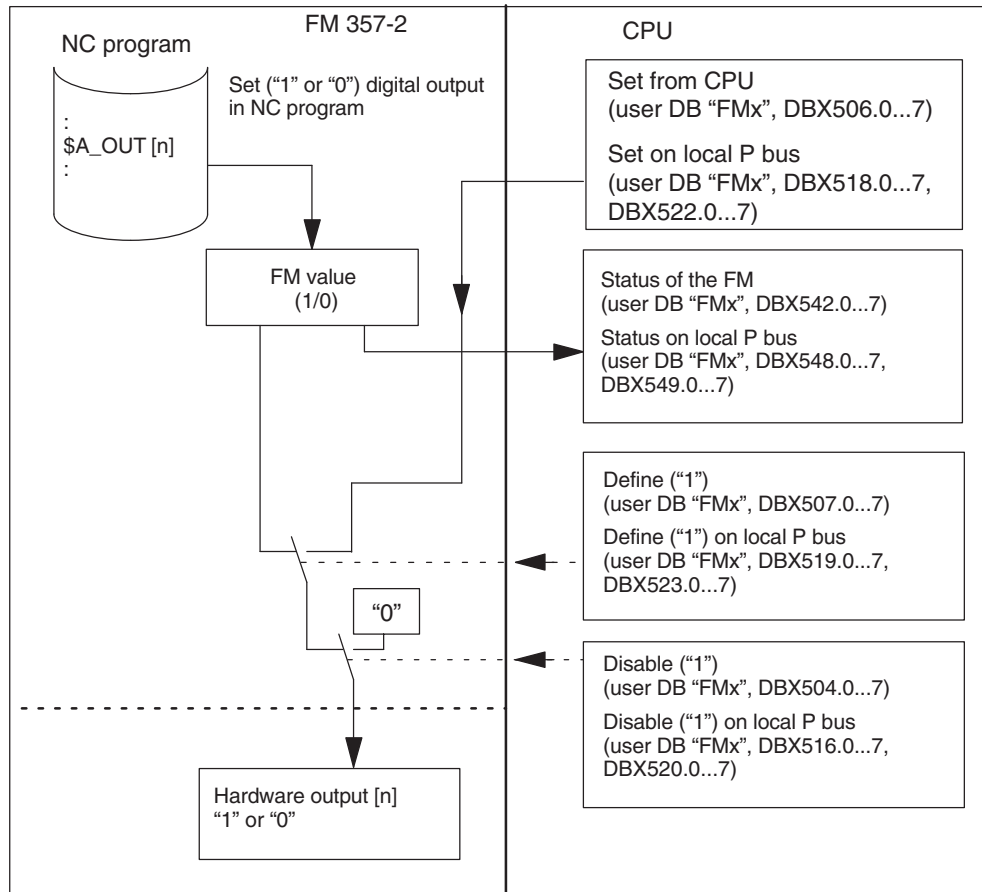


Fig. 9-17 Signal flow for digital outputs

9.10.2 Digital I/Os on local P bus

General

Digital signal modules (SMs) can be plugged into the local P bus of the FM 357-2. Digital I/Os are thus available for free use.

The signals are updated on each interpolator cycle and can be read and disabled by the NC program and the user program.

You must follow the rules for the mechanical configuration when setting up the local P bus. Two slots can be used on the local P bus.

The following digital I/Os can be implemented by plugging signal modules with 8 or 16 signals into the local P bus:

Table 9-12 Digital inputs/outputs on local P bus

Digital inputs	Digital outputs	Description
No. 17 to 32	No. 9 to 24	Implemented with signal modules

S7 configuration

The CPU detects the configuration of the local P bus in exactly the same way as the P bus configuration during FM 357-2 power-up.

When you have configured your project (see Section 5.3), you can call in the S7 Configuration by selecting the module and menu command **Edit > Object Properties the Properties** dialog (see Figure 5-2) .

Activate the local P bus as follows:

1. In the **Properties** dialog, click the **Basic Parameters** tab and select **Local Bus Segment** and confirm with OK.
2. Select **Station > Save and Compile** (the hardware project is now available in the *SIMATIC Manager* in the "System Data" block).
3. You can load this saved project into the CPU by selecting **PLC > Download to Module**.

The local P bus is now activated.

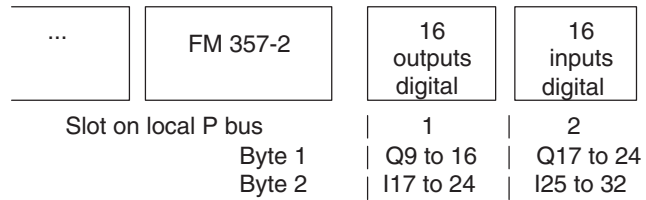
Parameter settings for hardware configuration

The following parameter settings send the slot address of the local P bus in which the inputs/outputs are located to the control system:

Parameters	Value/Meaning	Unit
Slots	None (default value) Slots 1 ... 4	–
Module size	1 byte (default value) 2 bytes	–
Byte 1	Inputs Outputs	–
Byte 2	Inputs Outputs	–
Outputs: 9...16 17...24 Inputs: 17...24 25...32	9...16; 17...24 (default value); 25...32 Assignment of bit numbers	–

Example of a configuration

16 digital inputs/outputs must be implemented on the local P bus. This requires 2 signal modules with 16 input and 16 output signals on the local P bus:



The parameters are passed as follows:

Slots	Slots 1+2	Slot 2
	Slot 1	Slot 2
Module size	Byte 2	Byte 2
Byte 1	Outputs 9 to 16	Inputs 17 to 24
Byte 2	Outputs 17 to 24	Inputs 25 to 32

Use

Reading and writing of digital inputs and outputs via the NC program:

Read: \$A_IN[n] n = Number of input
Write: \$A_OUT[n] n = Number of output

Examples:

- R1 = \$A_IN[9]
 ; The status of input 9 is stored in R1.
- \$A_OUT[9] = R1
 ; The contents of R1 (1 or 0) is output to output 9.
- \$A_OUT[10] = \$A_IN[11]
 ; The status of input 11 is output to output 10.

User program:

Digital inputs/outputs can also be read and disabled from the user program.

- Inputs (see Figure 9-16):
 The status of any input can be read.
- Outputs (see Figure 9-17):
 A disable can be assigned to any output, i.e. the output always has a defined "0" signal regardless of any other modifications (e.g. made by the NC program).
 If the output has not been disabled, it can be controlled from the NC program.
 The status of any output can be read.

9.10.3 Analog I/Os on the local P bus

General

A maximum of two analog signal modules (SMs) can be connected to the local P bus of the FM 357-2. In this way, analog inputs/outputs are realized for free use.

The analog inputs/outputs are updated at the interpolator clock cycle and can be read and written both by the NC program.

When creating the local P bus, observe the specifications with regard to the mechanical design.

Table 9-13 Analog inputs/outputs at the local P bus

Analog inputs	Analog outputs	Description
Inputs 1...8	Outputs 1...8	realized with signal modules

S7 configuration

The configuration of the local P bus is detected by the CPU exactly as that of the P bus during the power-up of the FM 357-2.

After you have configured your project (see Section 5.3), you can call the dialog **Properties** (see Fig. 5-2) via S7 configuration with selecting the module and the menu command **Edit > Object Properties**

The local P bus is enabled as follows:

1. Select the **Local Bus Segment** from the **Properties** dialog box via the **Basic Parameters** tab and then click OK to confirm.
2. Select **Station > Save and Compile** (the hardware object is now existing in the *SIMATIC Manager* in the "System Data" block).
3. Load this saved project into the CPU by selecting **Target System > Download to Module**.

The local P bus is thus enabled.

Parameterization for the hardware configuration

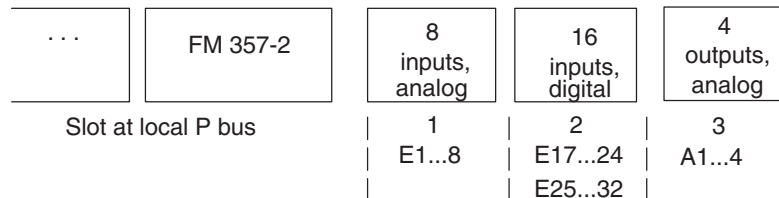
The following parameters tell the control system at which slot of the local P bus the analog inputs/outputs are:

Parameters	Value/Meaning	Unit
Slots	None (default value) Slots 1...4	–
Module size	2 inputs (default value) 4 inputs 8 inputs 4 outputs 4 inputs and 2 outputs 4 inputs and 4 outputs	–
Channel ¹⁾	1...8	–
Evaluation	0...10 V (default value) +/- 10 V	–

1) The “Channel” parameter designates the HW channel on the SM module, providing free assignment to the FM-internal number of the analog inputs/outputs.

Example of a configuration

8 analog inputs, 4 analog outputs and 16 digital inputs are to be realized at the local P bus. To this aim, a signal module with 8 analog inputs, a signal module with 4 analog outputs and a signal module with 16 digital inputs will be plugged onto the local P bus:



The following parameters must be loaded with the following values:

Analog inputs/outputs

Analog inputs/outputs	yes	yes
Slots	Slot 1	Slot 3
Modul size	8 inputs	4 outputs
Channel	1... 8	1... 4
Evaluation	0...10 V	0... 10 V

Digital inputs/outputs

Digital inputs/outputs	yes	
Slots	Slot 2	
Module size	2 bytes	
Byte 1	Inputs 17...24	
Byte 2	Inputs 25...32	

Use

Reading and writing of the analog inputs and outputs via the NC program.
Voltage value unit: mV.

Reading: \$A_INA[n] n = number of input
Writing: \$A_OUTA[n] n = number of output

Examples:

- N15 R3 = \$A_INA[2]
 ; The voltage value of input 2 is stored to R3.
- N20 \$A_OUTA[2] = R3
 ; The contents of R3 is output to output 2 as a voltage value.
- N30 \$A_OUTA[4] = \$A_INA[1]
 ; The voltage value of input 1 is output to output 4.

Example: Voltage output from synchronized action:

```
N40 ID=1 DO $A_OUTA[4] = $AC_VACTB * 0.4711
      ; depending on the current path velocity, a voltage value is output at each
      ; interpolation clock cycle
```

User program

The analog inputs/outputs cannot be read or written directly by the user program. If a voltage value is required in the user program, this can be transmitted, e.g. by an H command.

Example:

```
N10 H10 = $A_INA[2]
      ; The voltage value of input 2 is transferred to the CPU via H10.
```

9.11 Limit switching signals (software cams)

General

This function allows one or several pairs of cams to be assigned to a machine axis. A cam pair consists of a minus and a plus cam.

After activation (user DB "AXy", DBX2.0+m), cam signals are generated for the minus and plus cams when the specified cam positions are crossed and output as an interface signal (user DB "FMx", DBB32 and DBB33).

In addition, the cam signals can be output via digital outputs at the local P bus or on-board.

The "Limit switching signals" function operates in all modes and remains active even after a Reset or EMERGENCY STOP.

9.11.1 Parameterization

Cam pair Plus/Minus cams

The cams are assigned in pairs, each pair consisting of a minus and plus cam, to an axis via a parameter.

One or more cam pairs can be assigned to an axis. However, the same cam pair cannot be assigned to more than one axis.

Parameters	No.	Value/Meaning	Unit
Cam pair Axis number	1	0	not assigned (default value)
	1	1	(1st cam pair to 1st axis)
	2	1	(2nd cam pair to 1st axis)
	3	2	(3rd cam pair to 2nd axis)
	

Cam position

The cam position of the plus and minus cams are defined in the following parameters:

Table 9-14 Cam position parameters

Parameters	Value/Meaning	Unit
Cam position Minus cam	0 (default value) -100 000 000 to +100 000 000	[mm], [degrees]
Cam position Plus cam	0 (default value) -100 000 000 to +100 000 000	[mm], [degrees]

Notice

Cam positions are referred to the selected scaling system (metric or inch). A programmed switchover with G70/G71 has no effect.

The positions are entered in the machine coordinate system. The input is not verified against the maximum traversing range.

Lead time/delay time

To compensate for delay times, a lead and delay time for signal outputs can be assigned to each minus and plus cam. Input resolution: μs

Parameters	Value/Meaning	Unit
Lead time/delay time Minus cam	0 (default value) -100 to +100 Positive value = lead time Negative value = delay time	[s]
Lead time/delay time Plus cam	0 (default value) -100 to +100 Positive value = lead time Negative value = delay time	[s]

Signal level

This parameter allows the output signals for each cam to be inverted. The inversion affects **only** digital outputs.

Parameters	Value/Meaning	Unit
Signal level Minus cam	0 → 1 (default value) 1 → 0 (inverted)	—
Signal level Plus cam	0 → 1 (default value) 1 → 0 (inverted)	—

Assignment to the digital outputs; output response

Here you define the output response and the assignment of the cams to the digital outputs. The assignment is only possible by bytes.

Parameters	Value/Meaning	Unit
Assignment to the digital outputs, minus cam	No assignment (default value) Digital outputs (on-board) 1...8 Digital outputs 9...24	–
Assignment to the digital outputs, plus cam	XOR minus cam (default value) Digital outputs 9...24	–

No assignment

Plus and minus cams are output to the CPU at the IPO cycle (user DB “FMx”, DBB32 and DBB33). For this kind of cam output, timer-controlled limit switching signals or hot-spot measurement can additionally be parameterized.

Digital outputs (on-board) 1...8, digital outputs 9...24, XOR minus cams

Plus and minus cams are output to the parameterized digital outputs at the servo cycle. No timer-controlled limit switching signals are possible.

Plus and minus cams can be provided either to separated output bytes or linked to **one** output byte. For output to an output byte, select “XOR minus cam” for the minus cam.

The signal response for separated and linked output is described in Section 9.11.3 and 9.11.4.

Timer-controlled limit switching signals

The cam signal output is cycle-independent and is controlled by a timer interrupt. This leads to a higher degree of accuracy. The output is now only possible linked to on-board output 1 to 4 (see Section 9.11.4). For the plus cam, select “no assignment”.

Parameters	Value/Meaning	Unit
Timer-controlled position switching signals	No assignment (default value) 1st cam pair to 8th cam pair	–

Assignment of the **timer-controlled** limit switching signals to the digital on-board outputs 1 to 4.

The cams are assigned to the on-board outputs as follows:

1. in ascending order of axis number
2. in ascending order of cam pairs

Example:

Cam pair	Axis number	Assignment to outputs 1 to 4	
1	0		(no assignment)
2	2	4	2 nd Axis, cam pair 2
3	1	1	1 st axis, cam pair 3
4	1	2	1 st axis, cam pair 4
5	1	3	1 st axis, cam pair 5
6...8	0		(no assignment)

Signal output

The output response can be set for timer-controlled limit switching signals using the "Signal output" parameter.

- prioritized

Only one timer-controlled output can take place per IPO cycle. If signal changes are queued for several cam pairs in the same IPO cycle, the output is prioritized as follows:

The cam pair with the lowest number (on-board output with the lowest number) determines the output time for all queued signals, i.e. the signal change on the other cam pairs takes place at the same time.

- independent

The output of each cam signal is accurately timed. There is no output priority.

The cam signals are assigned to the on-board outputs in accordance with the prioritized signal output.

Parameters	Value/Meaning	Unit
Signal output prioritized	ON (default value) OFF	—

Path-time cam

This function provides the timer-controlled output of a switching pulse at a specified position. The pulse duration must be specified in the parameter “Actuation time/delay time of plus cam”.

The parameters “Cam position of minus cam” and “Cam position of plus cam” must be set to the same value; they define the default position of the path-time cam when the control system is turned on. The cam is enabled when this position is crossed. The pulse duration is then independent of the traversing rate, the traversing direction or a direction reversal of the axis.

Crossing the cam position once again with the cam active is not taken into account, i.e. the cam will not be restarted. This response can be noticed, in particular, at modulo rotary axes.

The action time/delay time for the plus cam remains effective and results in an offset of the whole cam.

This function can only be used for timer-controlled output, i.e. for output to the on-board outputs.

Note:

The term 'path-time cam' is referred to the cam function; a position and a time are specified.

The term 'timer-controlled output' is used for the (internal) procedure of cam output.

Parameters	Value/Meaning	Unit
Path-time cam	Off (default value) On Path-time cam for cam pairs with minus cam identical to plus cam position	–

Hot-spot measurement

This function can be used for timer-controlled enabling or disabling of axial measurement. The function is only allowed for a cam pair.

For further information, see Section 9.11.5.

The status measurement enabled or disabled can be output to on-board output 4. The parameter “Adjustment signal” must be set to **On**.

Parameters	Value/Meaning	Unit
Hot-spot measurement	No assignment (default value) 1st cam pair to 8th cam pair	–
Alignment signal	Off (default value) On Measurement enable/disabled on on-board output 4	–

9.11.2 Activation and output of the limit switching signals

Activation of the limit switching signals

The function is activated for every axis via the following interface signal:
“Activate software cam” (user DB, “AXy”, DBX2.0+m)

The successful activation of all cams of an axis is reported by the interface signal:
“Software cam active” (user DB, “AXy”, DBX22.0+m)

Notice

Activation in the user program can be tied to other conditions (e.g. axis referenced).

Output to interface

The status of the minus and plus cam signals is output for all machine axes with active limit switching signals by way of the following signals:

Table 9-15 Software cam minus/plus

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
User DB, “FMx”, DBB32	Software cam minus							
	8	7	6	5	4	3	2	1
User DB, “FMx”, DBB33	Software cam plus							
	8	7	6	5	4	3	2	1

Output to the digital outputs

In addition, it is possible to output the cam signals to digital outputs in the following form:

- either non-timer-controlled output to on-board or local P-bus outputs at the servo cycle, separated to two output bytes or linked to one output byte
- timer-controlled output to on-board outputs via timer interrupt, only linked to an output byte

Status interrogation in NC program

The status of the digital outputs on the local P bus can be read (n = no. of output) from the NC program with variable \$A_OUT[n].

Example:

```
...  
R78 = $A_OUT[5]           ; Read output 5, save in R78  
...
```

9.11.3 Limit switching signals with separated output

The output of minus and plus cams is provided separately to the interface and, if parameterized, to two digital output bytes.

Linear axes

The cam signals (minus and plus cams) are generated and output as a function of the axis traversing direction.

- The minus cam signal switches from 0 to 1 when the axis crosses the minus cam in the negative direction.
- The plus cam signal switches from 0 to 1 when the axis crosses the plus cam in the positive direction.

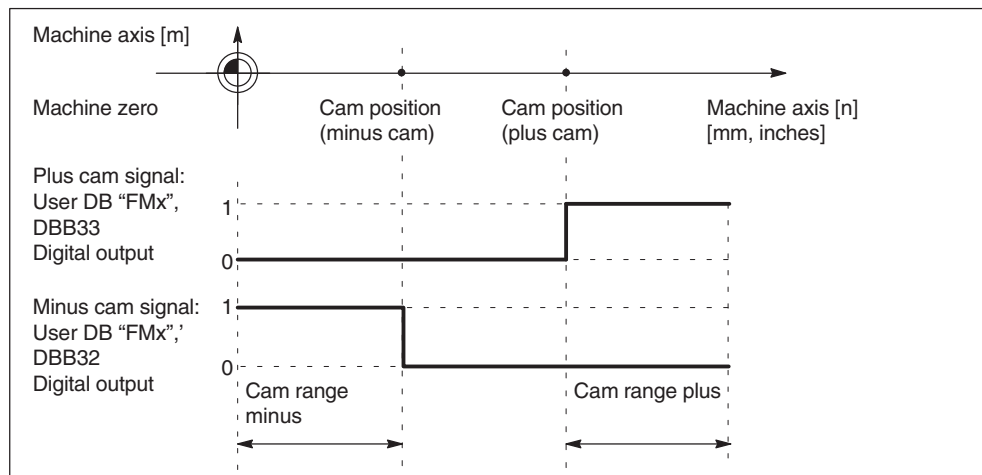


Fig. 9-18 Limit switching signals for linear axis (minus cam < plus cam)

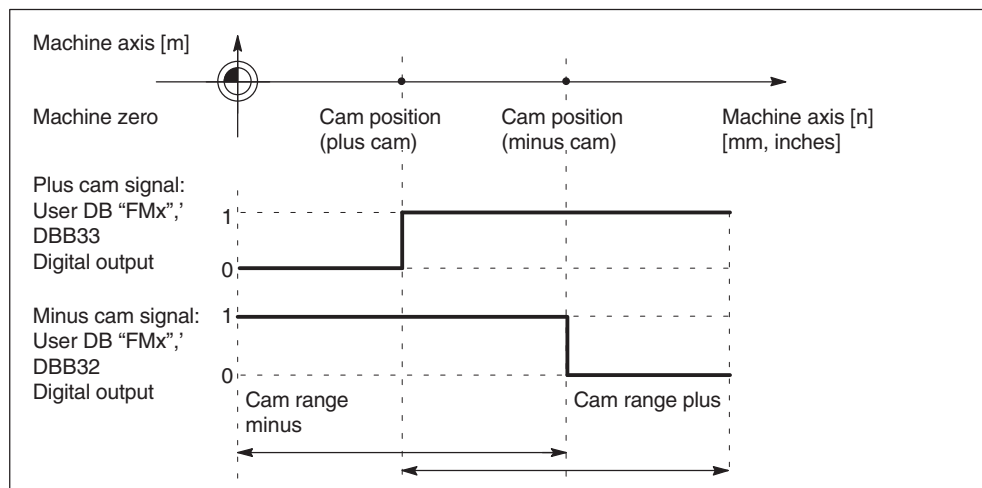


Fig. 9-19 Limit switching signals for linear axis (plus cam < minus cam)

Modulo rotary axes

The switching edges of the cam signals are generated as a function of the traversing direction of the rotary axis:

- The plus cam signal switches from 0 to 1 when the minus cam is crossed in the positive axis direction and switches back from 1 to 0 when the plus cam is crossed.
- The minus cam signal changes level on every positive edge of the plus cam signal.

Notice

The behavior of the plus cam described above is subject to the **condition**:
Plus cam – minus cam < 180°.

If this condition is not fulfilled, or if the minus cam is greater than the plus cam, the response of the plus cam signal is inverted. The response of the minus cam signal remains the same.

The cam crossing can also be detected on the signal change of the minus cam if the cam range has been set so small that the CPU cannot detect it reliably.

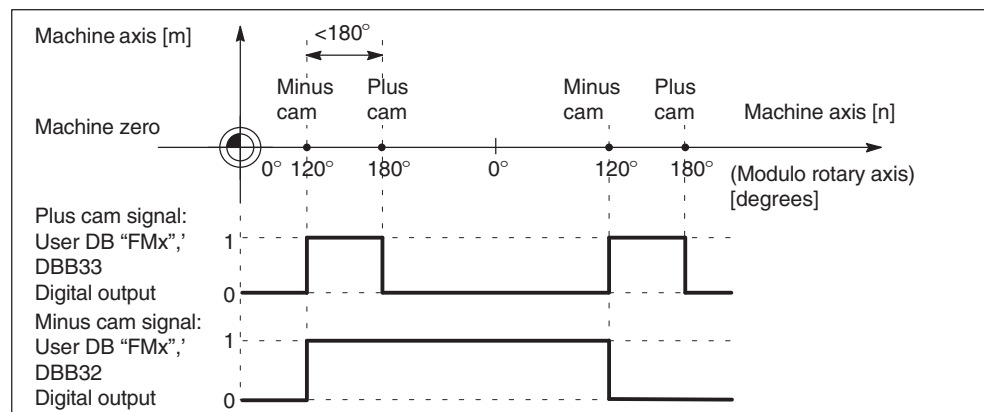


Fig. 9-20 Limit switching signals for modulo rotary axis
(plus cam – minus cam < 180 degrees)

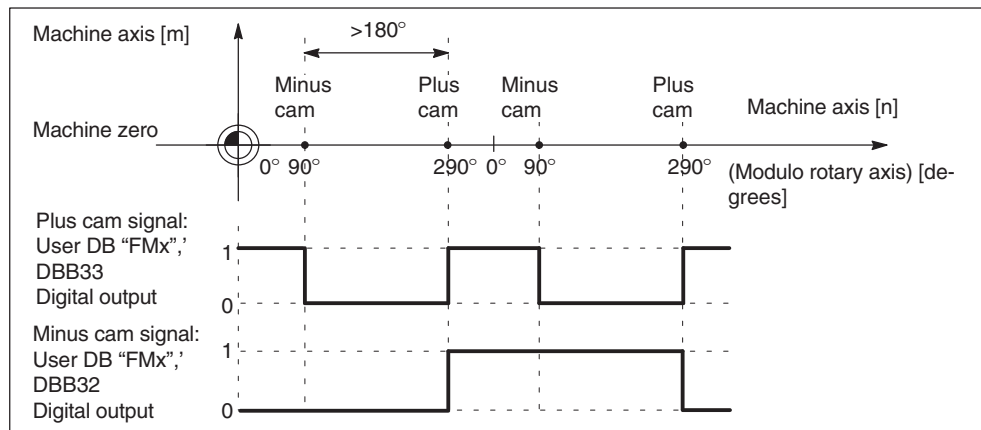


Fig. 9-21 Limit switching signals for modulo rotary axis
(plus cam – minus cam > 180 degrees)

9.11.4 Limit switching signals with linked output

The output of minus and plus cams is carried out linked to **one** digital output byte and also separated to the interface.

Linear axes

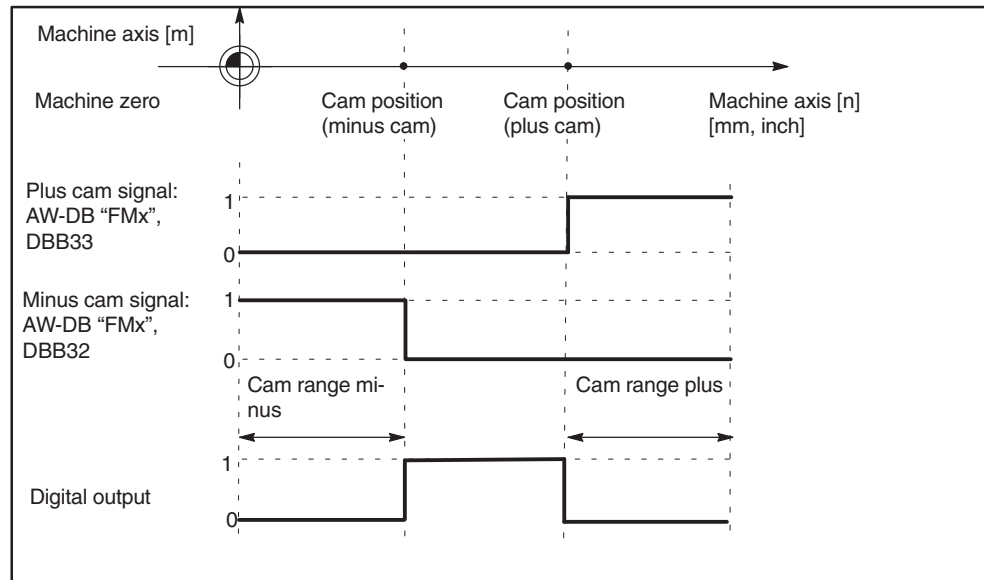


Fig. 9-22 Limit switching signals for linear axis (minus cam < plus cam)

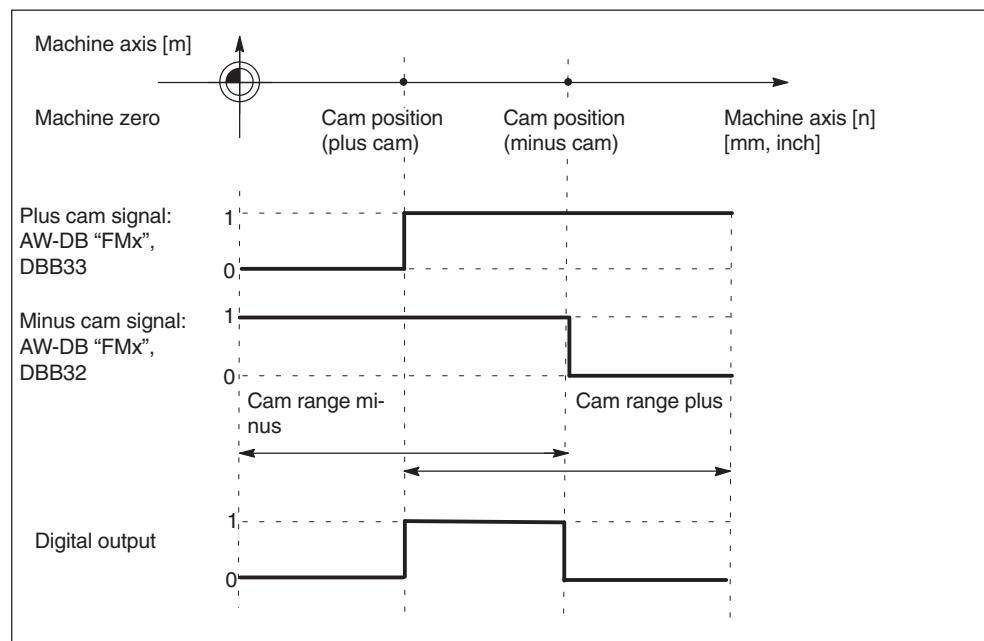


Fig. 9-23 Wertschaltsignale für Linearachse (Plusnocken < Minusnocken)

Modulo rotary axes

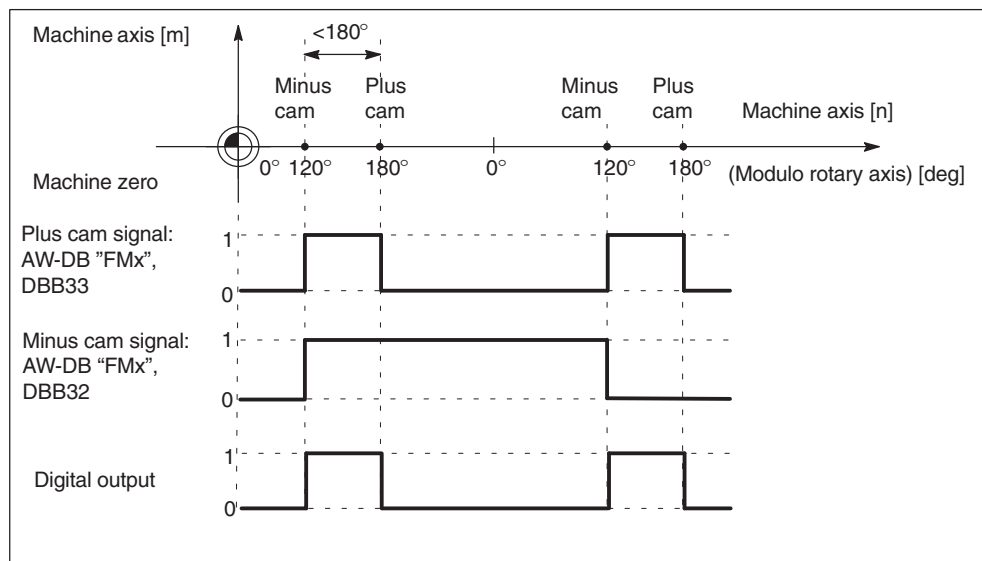


Fig. 9-24 Limit switching signals for modulo rotary axis
(plus cam – minus cam < 180 degrees; minus cam < plus cam)

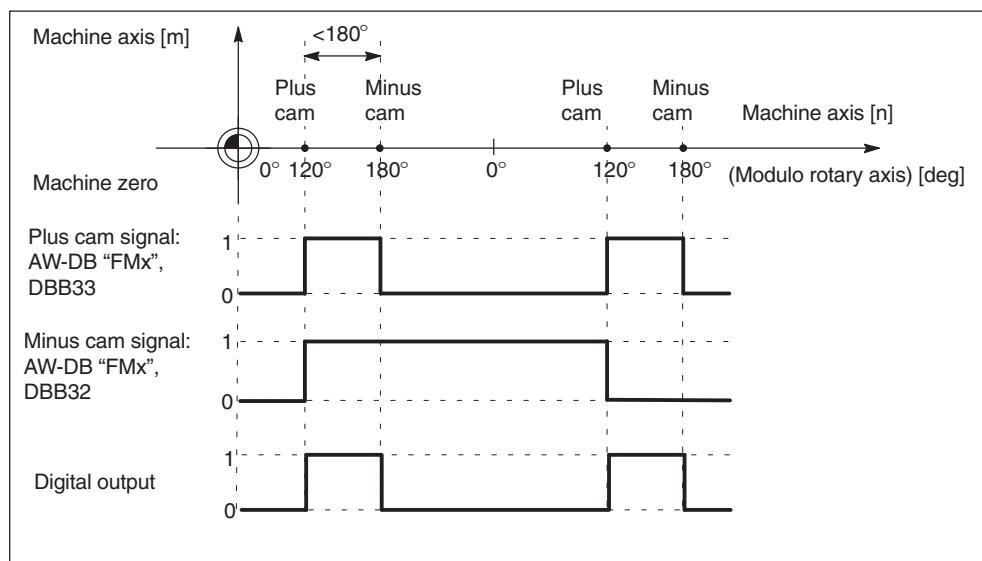


Fig. 9-25 Limit switching signals for modulo rotary axis
(plus cam – minus cam < 180 degrees; minus cam > plus cam)

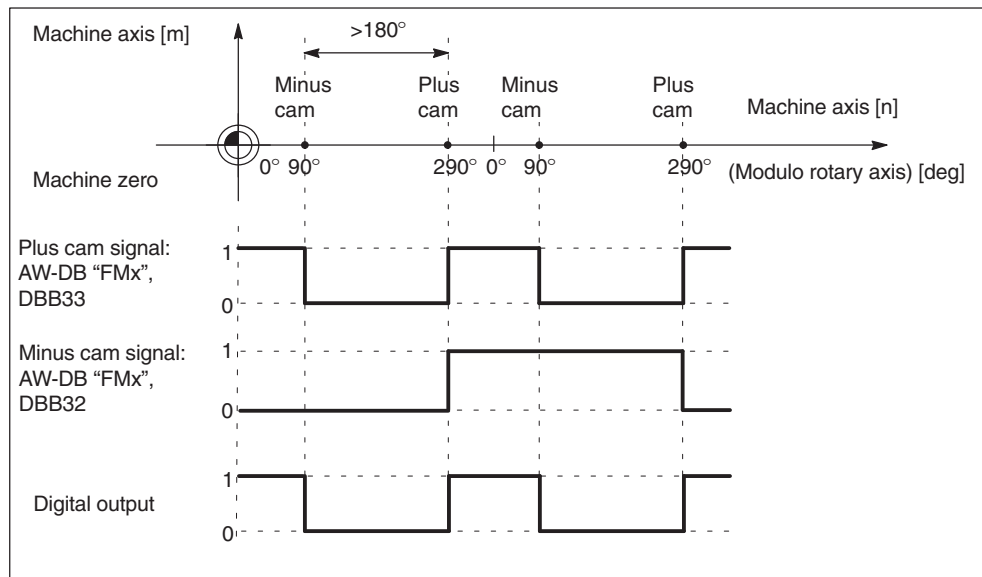


Fig. 9-26 Limit switching signals for modulo rotary axis
(plus cam – minus cam > 180 degrees; plus cam > minus cam)

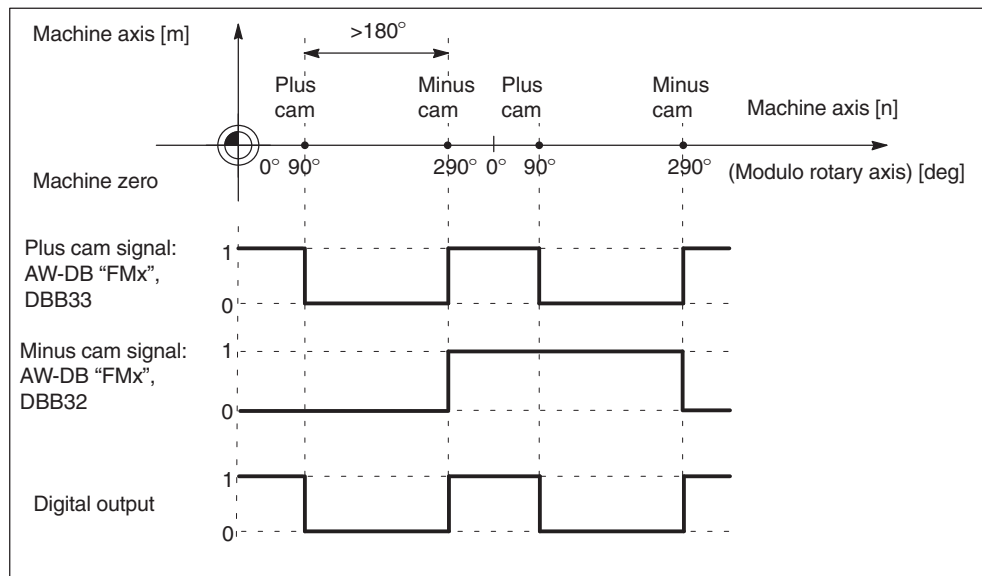


Fig. 9-27 Limit switching signals for modulo rotary axis
(plus cam – minus cam > 180 degrees; plus cam < minus cam)

Signal inversion

The output response to the digital output with modulo rotary axes and plus cam – minus cam > 180 degrees can be controlled using the parameter “Signal inversion”.

Parameter	Value/Meaning	Unit
Signal inversion Cam area > 180 degrees	<p>ON (default value) Signal inversion for plus cam – minus cam > 180 degrees</p> <p>OFF No signal inversion for plus cam – minus cam > 180 degrees (output response is accordingly plus cam – minus cam < 180 degrees)</p>	–

9.11.5 Hot-spot measurement

Measurement can be activated or deactivated as quickly as possible in the IPO cycle in a synchronized action.

The “Hot-spot measurement” function can be used to define a “hot spot” for the detection of probe signal edges.

The switching time of the hot spot is calculated in advance in the IPO cycle, as a function of the current position and velocity, and, if necessary, the probe signal is disabled or enabled via an interrupt function.

The existing functionality of block-oriented and axial measurement is unchanged and there are no dependencies or monitoring relationships between the measurement function and the hot spot.

The software cam functionality is the basis, although no signals are transferred to the hardware outputs for the selected cam pair.

For parameterizing please refer to Section 9.11.1

For hot spots for linear and modulo rotary axes please see Fig. 9-28 to 9-31.

Linear axes

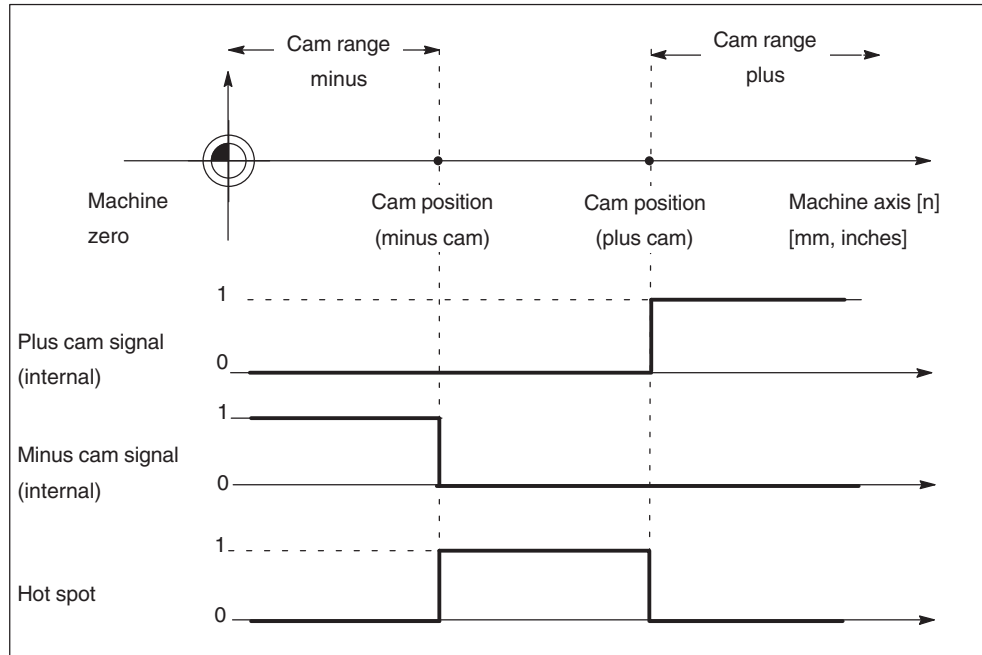


Fig. 9-28 Hot spot for linear axis (minus cam < plus cam)

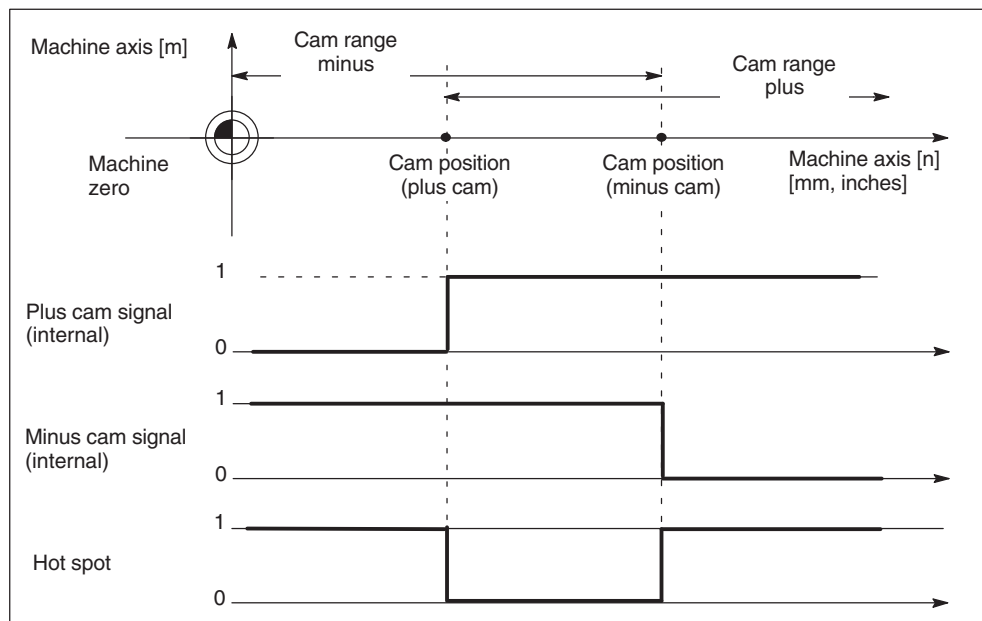


Fig. 9-29 Hot spot for linear axis (plus cam < minus cam)

Modulo axes

The response is valid for plus cam – minus cam < 180 degrees. If this condition is not fulfilled, or if the minus cam is greater than the plus cam, the response is inverted.

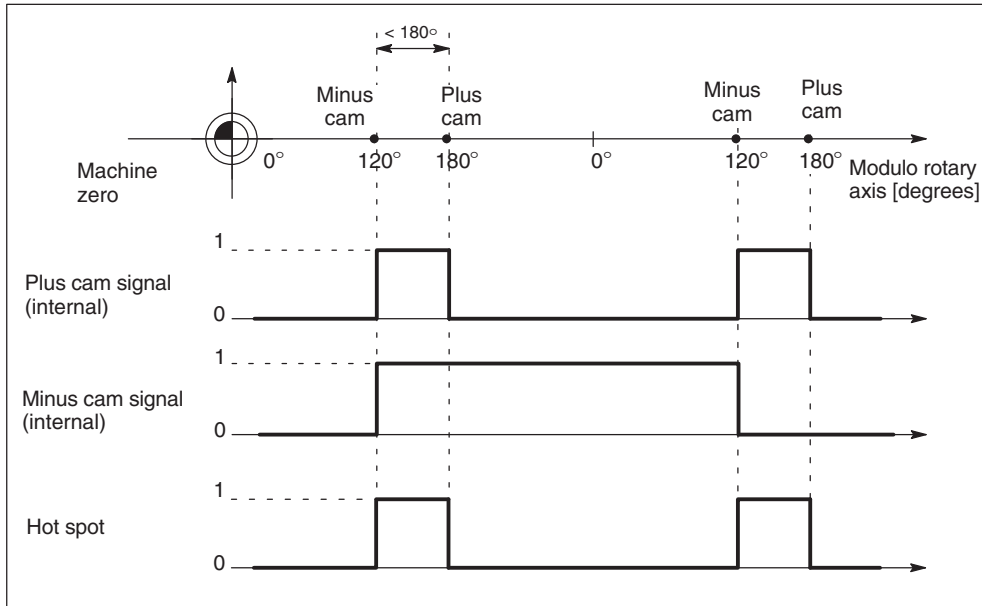


Fig. 9-30 Hot spot for modulo rotary axis (plus cam – minus cam < 180 degrees)

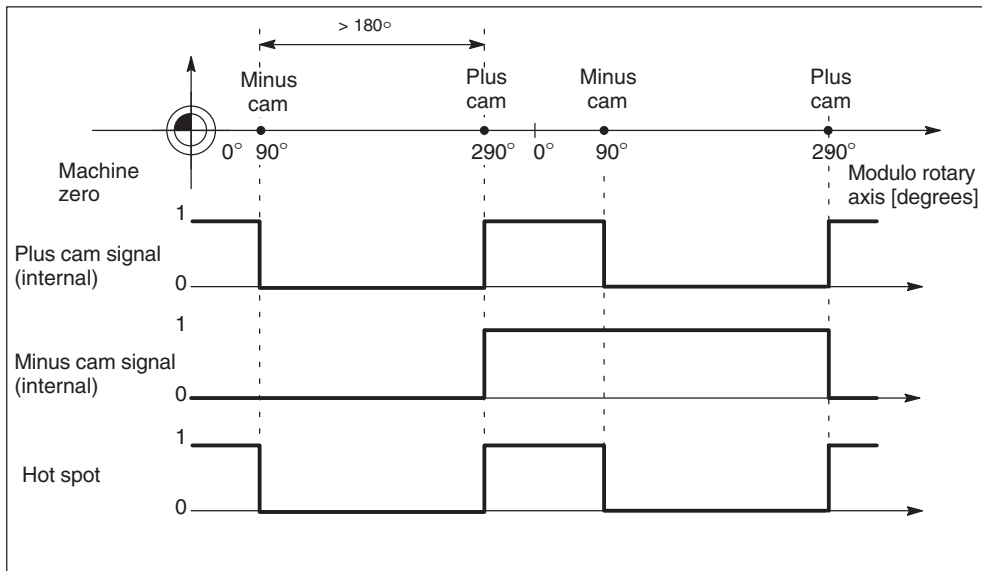


Fig. 9-31 Hot spot for modulo rotary axis (plus cam – minus cam < 180 degrees)

Since the advance calculation of the switching times is performed in the IPO cycle, the difference between the actual position and the position setpoint during the movement must be considered when defining the hot spot or the following error.

The hot spot always applies to both probes and all axes, regardless of the programming instructions.

A signal edge can only be detected within the hot spot. Allowance must be made for missing measurements (if necessary, an average or anticipated value should be used).

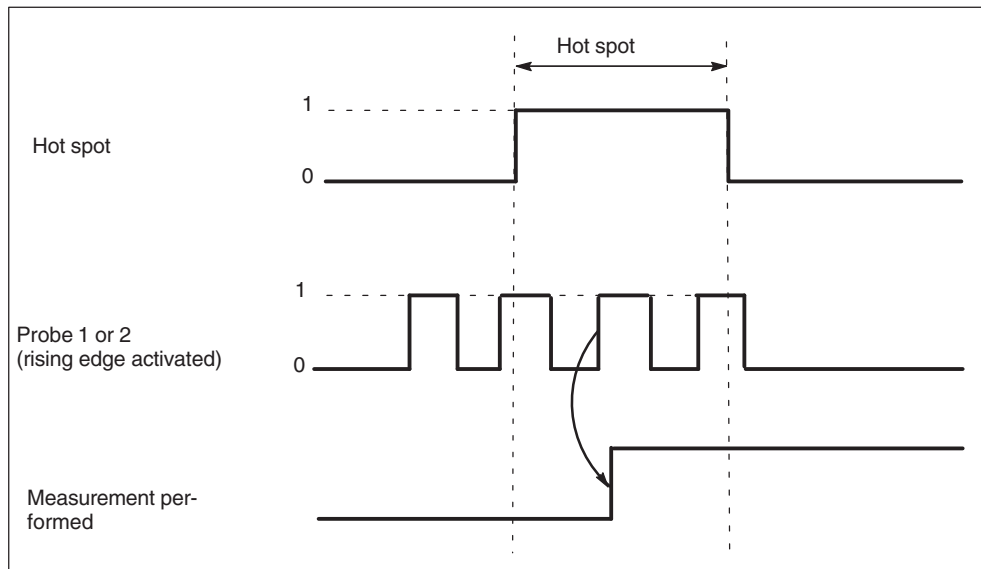


Fig. 9-32 Hot-spot measurement

The status of the cam signals continues to be output to the CPU; the “Activate software cams” signal (user DB “AXy”, DBX2.0+m) must be set by the CPU.

The function is not permitted for PROFIBUS DP axes.

The distance between two measurement events continues to determine the IPO cycle (minimum of 2 IPO cycles).

If measurement is active, the probe status ($\$A_PROBE[n]$) is not indicated outside the hot spot.

9.12 Operating modes

General

Different operating modes can be selected for each active channel.
The following modes are available on the FM 357-2:

Table 9-16 Operating modes and their properties

Mode	Property
<p>Jog (T)</p> <p>Control and checkback signals</p> <p>Parameters</p>	<p>In this mode, the traversing movement of an axis is defined using a direction key (plus or minus direction).</p> <p>The axis travels at the velocity set in the "Axis velocity" parameter. When the rapid traverse override is activated, the axis travels at the velocity entered in the "Rapid traverse" parameter. Allowance is made for the defined override.</p> <p>Select mode: "Jog" control signal (user DB "FMx", DBX100.2+n)</p> <p>Mode checkback: "Jog" checkback signal (user DB "FMx", DBX120.2+n)</p> <p>Start motion: "Direction plus or minus" control signal (user DB "AXy", DBX4.7/6+m)</p> <p>Parameter "Axis velocity" Parameter "Rapid traverse"</p>
<p>Incremental relative (SMR)</p> <p>Submode of "Jog"</p> <p>Control and checkback signals</p> <p>Parameters</p>	<p>In this mode, relative individual positioning operations are executed in response to an incremental dimension input with 1, 10, 100, 1 000 or 10 000 increments.</p> <p>The axis travels at the velocity set in the "Axis velocity" parameter. When the rapid traverse override is activated, the axis travels at the velocity entered in the "Rapid traverse" parameter. Allowance is made for the defined override.</p> <p>The movement is started with the signals (direction plus or minus).</p> <p>The movement is interrupted with Stop and the distance to go is cleared.</p> <p>The movement is interrupted if the "Direction plus or minus" signals are no longer active or if a Stop is initiated when the signals ("Direction plus or minus") are active. If the "direction minus or plus" signals are activated again, the movement is continued and the remaining distance is traversed. Pressing Reset aborts the movement and deletes the distance to go.</p> <p>Select mode: "Jog" control signal (user DB "FMx", DBX100.2+n)</p> <p>Mode checkback: "Jog" checkback signal (user DB "FMx", DBX120.2+n)</p> <p>Increment command: "Incremental dimensions" control signal (user DB "FMx", DBX102.0...4+n)</p> <p>Incremental dimensions active: "Incremental dimensions active" checkback signal (user DB "FMx", DBX122.0...4+n) (user DB "AXy", DBX24.0...4+m)</p> <p>– for geometry axes – for special axes</p> <p>Start motion: "Direction plus or minus" control signal (user DB "AXy", DBX4.7/6+m)</p> <p>Parameter "Axis velocity" Parameter "Rapid traverse"</p>

Table 9-16 Operating modes and their properties, continued

Mode	Property
<p>Reference point approach (REF) Submode of “Jog”</p> <p>Control and checkback signals</p> <p>Parameters</p>	<p>Approach a reference point on axes with incremental encoders. The signals (direction plus or minus) are used to start the reference point approach, and reference the axis in accordance with the parameter definitions.</p> <p>Select mode: “Jog” and “reference point approach” control signal (user DB “FMx”, DBX100.2 and DBX101.2+n)</p> <p>Mode checkback: “Jog” and “reference point approach” checkback signal (user DB “FMx”, DBX120.2 and DBX121.2+n)</p> <p>Start motion: “Direction plus or minus” control signal (user DB “AXy”, DBX4.7/6+m)</p> <p>Referencing: “Reference point approach delay” control signal (user DB “AXy”, DBX7.7+m)</p> <p>(see Section 9.7.1)</p>
<p>MDI (Manual Data Input)</p> <p>Control and checkback signals</p>	<p>Internal operating mode, available only in conjunction with “Parameterize FM 357-2”.</p> <p>Execute an NC program block. Block execution is started with Start.</p> <p>Select mode: “MDI” control signal (user DB “FMx”, DBX100.1+n)</p> <p>Mode checkback: “MDI” checkback signal (user DB “FMx”, DBX120.1+n)</p> <p>Start NC block execution: “Start” control signal (user DB “FMx”, DBX108.1+n)</p>
<p>Automatic (A)</p> <p>Control and checkback signals</p>	<p>In this mode you can execute NC programs automatically (sequential block execution). Once you have selected an NC program, you can start it with Start. Program test functions can be activated while the program is running (see Section 9.13).</p> <p>Select mode: “Automatic” control signal (user DB “FMx”, DBX100.0+n)</p> <p>Start NC block execution: “Start” control signal (user DB “FMx”, DBX108.1+n)</p> <p>Stop NC block execution: “Stop” control signal (user DB “FMx”, DBX108.3+n)</p> <p>Program control: “Read-in disable” control signal (user DB “FMx”, DBX107.1+n) “Skip block” control signal (user DB “FMx”, DBX105.0+n)</p> <p>Mode checkback: “Automatic” checkback signal (user DB “FMx”, DBX120.0+n)</p> <p>Status signals: “Program running” checkback signal (user DB “FMx”, DBX125.0+n) “Program waiting” checkback signal (user DB “FMx”, DBX125.1+n) “Progr. interrupted” checkback signal (user DB “FMx”, DBX125.3+n)</p>

Table 9-16 Operating modes and their properties, continued

Mode	Property
<p>Automatic single-block (AE) Submode of "Automatic"</p> <p>Control and checkback signals</p>	<p>In this mode, execution of the NC program stops after every NC block containing actions (traversing motions, auxiliary function outputs, etc.). The next block is executed with Start. Execution is not stopped after computing blocks, as these do not trigger actions.</p> <p>After an NC block in the NC program has been executed in single block mode, the program status "Program interrupted" is displayed.</p> <p>Select mode: "Automatic" control signal and "Activate single-block" (user DB "FMx", DBX100.0+n and DBX103.4+n)</p> <p>Start NC program execution: "Start" control signal (user DB "FMx", DBX108.1+n)</p> <p>Stop NC block execution: "Stop" control signal (user DB "FMx", DBX108.3+n)</p> <p>Program control: "Read-in disable" control signal (user DB "FMx", DBX107.1+n) "Skip block" control signal (user DB "FMx", DBX105.0+n)</p> <p>Mode checkback: "Automatic" checkback signal (user DB "FMx", DBX120.0+n)</p> <p>Status signals: "Program running" checkback signal (user DB "FMx", DBX125.0+n) "Program waiting" checkback signal (user DB "FMx", DBX125.1+n) "Progr. interrupted" checkback signal (user DB "FMx", DBX125.3+n)</p>

Operating mode switchover

Another operating mode can only be selected in the Reset state or after Stop.

An auxiliary function which has been output must be acknowledged before the mode is changed.

If the system rejects an operating mode changeover, an appropriate error message is output.

Exception:

You can change between the modes "Automatic" and "Automatic single block" at any time without triggering an internal Stop.

9.13 NC program execution

General

In “Automatic” mode, NC programs can be processed independently by the FM 357-2. The NC programs contain instructions for moving axes and controlling the plant.

NC program execution sequence

A typical program run is as follows:

Table 9-17 Typical program run

No.	Command	Comments
1	Write NC program and load on FM 357-2	Using the “Parameterize FM 357-2” tool
2	Selection of “Automatic” mode	see Section 6.3.3
3	Program selection	Only possible in Reset state
4	Set the desired program controls	E.g. “Skip blocks”
5	Start the program	Triggered by signal “Start” (user DB, “FMx”, DBX108.1+n); the program status (program running) is then displayed
6	M02/M30/Reset	The program status (program aborted) is displayed

Select program

You can select an existing program on the FM 357-2 in one of the following ways:

- From user program (UP) with FB 4 (select program)
- Using the “Parameterize FM 357-2” tool
- From the OP 17 (if appropriately configured)

Program states

An NC program can assume the following states while it is running:

Table 9-18 Program states

Program state	Description
Program aborted (user DB, "FMx", DBX125.4+n)	The program is selected but has not been started, or a running program was aborted with Reset.
Program interrupted (user DB, "FMx", DBX125.3+n)	Indicates that the NC program can continue execution on Start. The NC program is interrupted when you change modes, e.g. from "Automatic" to "Jog". In "Automatic" mode, you can continue execution with Start.
Program stopped (user DB, "FMx", DBX125.2+n)	The NC program has been stopped, e.g. by Stop.
Program waiting (user DB, "FMx", DBX125.1+n)	The running NC program has detected a WAIT statement. The condition for the statement has not yet been fulfilled.
Program running (user DB, "FMx", DBX125.0+n)	The NC program was started with Start and is running, or a running program was stopped with read-in disable.

Program test functions

The following test functions are provided for testing and trying out new NC programs. The use of these functions greatly reduces the risk of damage to the machine tool during the test phase and the overall time spent testing the program. It is possible to activate several program test functions at once to enhance the results.

- Program execution without axis motion

The NC program is started and executed when the function is active and Start is selected. Program execution differs from normal execution in the following respect:

- An internal axis disable is valid for all axes, i.e. the machine axes do not move, and the actual values are generated internally from the setpoints which are not output.
- Position control is not interrupted, with the result that the axes do not have to be referenced when the function is deactivated.
- This allows the user to check the programmed axis positions and the auxiliary function outputs of an NC program.
- The function is activated via the interface signal "Activate program test" (user DB "FMx", DBX104.7+n) and the acknowledgment via the interface signal "Program text active" (user DB "FMx", DBX124.7+n).

- Skipping blocks

Blocks which start with the characters “/” in the NC program are masked out during program execution when the function is activated, i.e. they are not executed.

Activation using interface signal “skip block” (user DB, “FMx”, DBX105.0+n)

- Programmed stop

The M01 in the NC program causes a programmed stop during program execution.

Activation via interface signal “Activate M01” (user DB, “FMx”, DBX103.5+n)

9.14 Asynchronous subprogram (ASUB)

General

An asynchronous subroutine (ASUB) is an NC program that can be started in response to an external event.

ASUBs are parameterized in the NC program, and in the ASUB itself, and activated by way of digital inputs or by the user program.

An NC block which is being executed is interrupted immediately. It is possible to resume the NC program at a later stage at the point of interruption.

Different priorities must be assigned to multiple ASUBs so that the system can determine the processing order if events occur simultaneously.

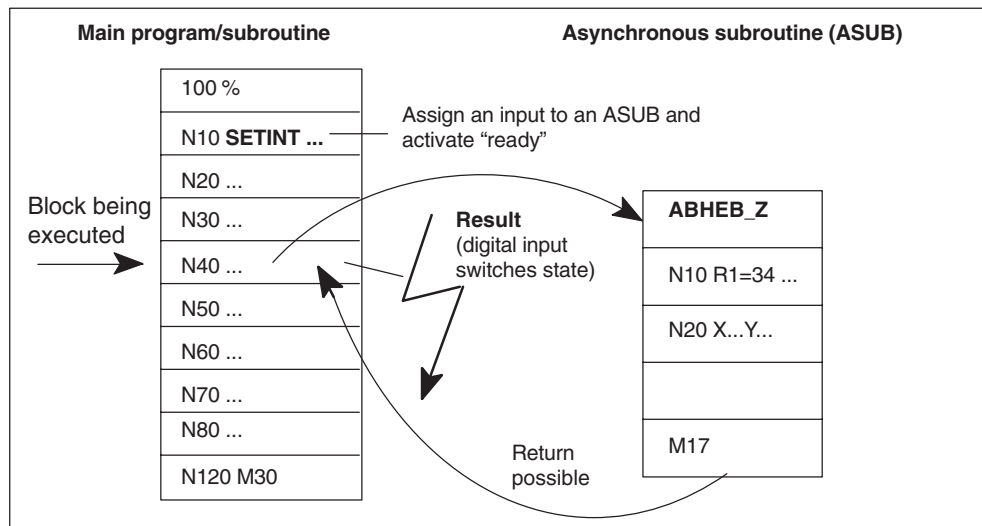


Fig. 9-33 Execution of asynchronous subprograms

Instructions in NC program

The following instructions are available for programming and parameterizing ASUBs in the NC program (see Section 10.30):

Statement in NC program:

SETINT(n) PRIO=1 NAME SAVE

SETINT(n) ; Assignment of a digital input/interrupt no. (n = 1...8/8)
; to an NC program to make the program an ASUB

PRIO = m ; Definition of priority (m = 1 to 128, 1 is highest priority)

NAME ; Name of the ASUB

DISABLE(n) ; Disable ASUB (n = no. of digital input)

ENABLE(n) ; Enable ASUB (n = no. of digital input)

CLRINT(n) ; Clear assignment between digital input and
; NC program

Statements in the ASUB:

SAVE ; Save interruption position and
; processing status

REPOS ; Reposition at interruption point in
; main program/subroutine

Digital inputs for starting ASUBs

The following 8 on-board inputs are available for starting ASUBs (see Section 4.8):

- X1, pin 28 Digital input no. 1
- X1, pin 29 Digital input no. 2
- X1, pin 30 Digital input no. 3
- X1, pin 31 Digital input no. 4
- X1, pin 32 Digital input no. 5
- X1, pin 33 Digital input no. 6
- X1, pin 34 Digital input no. 7
- X1, pin 35 Digital input no. 8

Activation of an ASUB

ASUBs can be activated by two methods:

- 0/1 edge at digital input (inputs 1 to 8)
- “Start ASUB” control signal (user DB “FMx”, DBX110.7+n (interrupt no. 8)

After activation, all machine axes are decelerated to a standstill at the axis acceleration rate, and the axis positions are stored.

Reorganization

In addition to deceleration of the axes, the pre-decoded calculation blocks are re-calculated up to the interruption block and stored again. After the end of the ASUB, the NC program can be continued with the “correct” values.

Exception: Reorganization is not possible with splines.

Processing of interrupt routine

On completion of reorganization, the “Interrupt” routine is started automatically. It is handled in the same way as a normal subroutine.

End of an ASUB

After the end identifier (M02) of the ASUB has been processed, the axis traverses to the end position of the NC program block following the interruption block.

If you want to reposition the axis at the interruption point, you must insert a RE-POS statement at the end of the ASUB (e.g.: REPOSL M02).

9.15 Protection zones

General

Protection zones can be used to protect fixed and moving parts on the machine and the machine equipment (e.g. swivel probes) against collision.

2 or 3-dimensional protection zones are defined in the NC program for the elements to be protected on the machine. The protection zones can be preactivated, activated or deactivated by statements in the NC program.

Preactivated protection zones can be activated/deactivated by the user program.

Notice

During execution in the operating modes “Jogging”, “Incremental relative”, “MDI” or “Automatic”, the system checks whether the tool (or its protection zone) violates the protection zones of the workpiece.

The protection zones are not valid until all participating geometry axes have been referenced.

Number of protection zones

The following parameter defines the maximum number of protection zones:

Parameters	Value/Meaning	Unit
Number of protection zones	<p>0: No protection zones can be defined (default value).</p> <p>1, 2, 3, 4: This number of protection zones can be defined.</p> <p>Note: A change in this parameter causes the reconfiguration of the user memory, and thus the deletion of all user data.</p>	—

Types of protection zone

A distinction is made between the following types of protection zone:

- **Workpiece-related protection zones**

Workpiece-related protection zones refer to the zero point of the workpiece coordinate system (WCS) and are specified by an absolute value.

- **Tool-related protection zones**

Tool-related protection zones refer to the tool carrier reference point F and are specified by an absolute value. These protection zones move with the tool.

Two or three-dimensional protection zones are defined by a maximum of four corner points in the contour description. The protection zones can also contain arcs as contour elements. The contour is defined in a previously specified plane.

Notice

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

If no workpiece-related protection zone is active, no protection zone verification takes place.

Example

The protection zone around the tool carrier reference point F moves with an axis movement. When the axis is moved and the protection zones are active, the system checks whether the protection zones are violated.

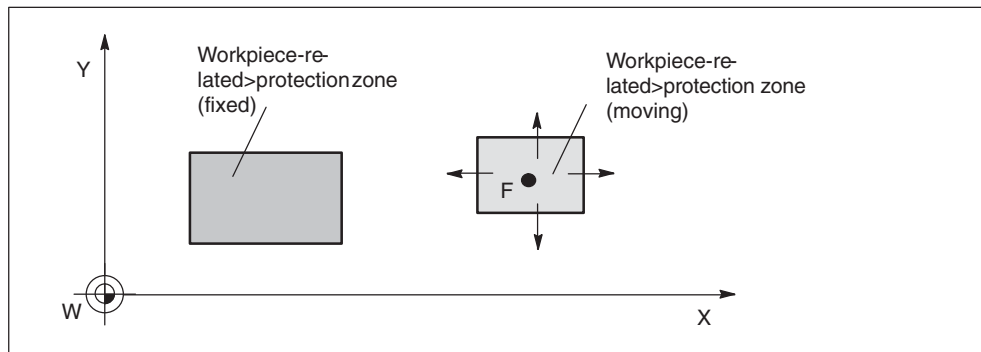


Fig. 9-34 Example of a workpiece/tool-related protection zone

Coordinate system, axis types

Protection zones for geometry axes are defined in the rectangular workpiece coordinate system (WCS).

Only the end point is verified where geometry axes are moved as positioning axes; the error "Protection zone monitoring not guaranteed" is output during the movement.

Orientation

The orientation of the protection zones is determined by defining the plane (abscissa/ordinate) in which the contour is described.

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

Programming protection zones

The following instructions are provided for programming a protection zone (see Chapter 10):

`NPROTDEF()` ; Begin definition of protection zones

`EXECUTE` ; End of definition

`NPROT()` ; Activate protection zone n

Preactivation, activation, deactivation

Protection zones can have the status “preactivated” “activated” and “deactivated”. This status is assigned to a protection zone using the NPROT statement in the NC program.

- A preactivated protection zone can be activated by the user program.
- The protection zone can only be deactivated by the NC program.
- The status of the protection zones (deactivated/preactivated/activated) is retained after the end of the program or a Reset.
- Protection zones activated by the NC program cannot be disabled by the CPU program.

Activation by the user program

Preactivated protection zones can be activated/deactivated by the user program.

Protection zones are preactivated and a positional shift specified by way of NC programming.

The following interface signals indicate which protection zones are preactivated or violated:

- Interface signal “Protection zone 1...4 preactivated” (user DB, “FMx”)
 - DBX622.0...3 for channel 1
 - DBX630.0...3 for channel 2
 - DBX638.0...3 for channel 3
- Interface signal “Protection zone 1...4 violated” (user DB, “FMx”)
 - DBX624.0...3 for channel 1
 - DBX632.0...3 for channel 2
 - DBX640.0...3 for channel 3

The following interface signal is used to activate/deactivate preactivated protection zones:

- Interface signal “Activate protection zone 1...4” (user DB, “FMx”)
 - DBX582.0...3 for channel 1
 - DBX588.0...3 for channel 2
 - DBX594.0...3 for channel 3

Protection zone violation

In modes “Automatic” and MDI”

Protection zones are not crossed in these modes:

- If a block moves into a protection zone from outside, the axis decelerates to the end of the previous block and the movement is stopped.
- If the start point of the block is already inside the protection zone, the movement is not started.

An appropriate error is output on protection zone violation for the workpiece-related protection zone.

In “Jog” and “Incremental relative” modes

The traversing movement of an axis is limited when the axis reaches a protection zone, and an error is displayed.

Crossing workpiece-related protection zones

When a protection zone is violated in “Automatic”, “Jogging” and “Incremental relative” modes, it is possible to temporarily enable crossing of the workpiece-related protection zone with Start.

The error is cleared and the axis moves into the protection zone.

For enhanced safety of a protection zone, you must interlock the Start or program further conditions in the CPU user program and execute them when the error occurs.

If you do not want the protection zone to be crossed, terminate the travel command with Reset.

If more than one protection zone is violated by the movement at the same time, acknowledgement is required for each of these protection zones. You can enable the protection zones individually with Start.

9.16 Motion coupling

Overview

In this section, you can find information about:

- Coupled motion, Section 9.16.1, page 9-114
- Gantry axes, Section 9.16.2, page 9-117
- Master-value coupling, Section 9.16.3, page 9-124
- Electronic gear, Section 9.16.4, page 9-131
- Tangential control, Section 9.16.5, page 9-137
- Overlaid motion in synchronized actions, Section 9.16.6, page 9-140

9.16.1 Coupled motion

General

This function allows you to declare any axis of your choice as a “master axis” and to assign any number of axes as “slaves” to the master. Together, the axes form a coupled-axis grouping.

The leading axis and the slave axis (or axes) are defined and activated/deactivated by means of statements in the NC program.

Position of a slave axis

The position of a slave axis at any given point in time is the product of the dependent motion (motion of master axis allowing for coupling factor) and the independent motion (the motion programmed for the slave axis).

Axis types

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

A simulated axis can also be defined for the leading axis.

Coordinate system

Coupled-axis motion is always implemented in the workpiece coordinate system (WCS).

Programming of a coupled-axis grouping

The following instructions are provided for programming a coupled-axis grouping (see Section 10):

TRAILON(slave axis, leading axis, coupling factor)
; Define and activate a coupled-axis grouping

TRAILOF(slave axis, leading axis)
; Deactivate a coupled-axis grouping

\$AA_COUP_ACT[axis] = 0 ; No coupling active

\$AA_COUP_ACT[axis] = 8 ; Coupled motion active

; Scan the status of the axis coupling using system variables in the NC program

Response in different operating modes

The following responses of a coupled-axis grouping in different modes must be noted:

- **Activation**

An activated coupled-axis grouping is active in the “Automatic”, “MDI”, “Jog” and “Incremental travel relative” modes.

- **Referencing**

The associated couplings are not deactivated while you reference a coupled-motion axis.

- **Delete distance to go**

Deleting the distance to go on a leading axis causes all axes of the associated coupled-axis grouping to come to a standstill.

Deleting the distance to go on a slave axis only stops the independent movement of the axis.

- **Initial setting on power-up**

No coupled-axis groupings are active on power-up.

- **Behavior after reset/end of program**

You can define in parameters whether the active coupled-axis groupings are canceled or retained after reset/end of program.

Parameters	Value/Meaning	Unit
Active coupled-axis groupings remain valid	No: Coupled-axis groupings are dissolved (default value).	–
	Yes: Coupled-axis groupings are retained	

Special features

The following special features of coupled motion must be noted:

- **Dynamic response of control system**

Depending on the application, it can be practical to match the position control parameters of the leading axis and the slave axis (e.g. K_V factor).

- **Acceleration and velocity limits**

The acceleration and velocity limits of the coupled axes are determined by the “slowest axis” in the grouping.

- **Multiple coupling**

If, when a coupling is activated, the system detects that a coupled-axis grouping with a leading axis and slave axis is already active, the activation operation is ignored, and an appropriate error message is generated.

- **Actual-value display**

The display of the actual position and the setpoint/actual value difference is updated for all axes in a coupled-axis grouping.

The setpoint/actual value difference on slave axes refers to the sum of the independent and dependent movement paths.

Effectiveness of interface signals

The effectiveness of interface signals in coupled motion as described below must be noted:

The only interface signals to be effective for a slave axis operating dependently of a master are those which cause a motion stop (e.g. axis-specific feed stop, controller enable, etc.).

When a coupled-axis grouping is activated, the interface signals from the master axis act on the associated slave axis via the axis coupling.

If the master axis is shut down by interface signals (e.g. axis-specific feed stop, controller, enable, etc.), then the associated slave axis is stopped at the same time.

9.16.2 Gantry

General

The Gantry function permits two machine axes to be driven in absolute mutual synchronism, allowing, for example, axes in a rigid mechanical coupling to be traversed without offset. A gantry grouping consists of a leading and a synchronized axis. A maximum of two gantry links may be defined. The axes of a gantry grouping must be in the same channel. In a gantry grouping, only the leading axis may be traversed as a normal NC axis on the basis of programming or operator input. The synchronized axis is moved exclusively by the Gantry function.

The function is available for the FM 357-2LX.

Parameterization

The following table describes all parameters required for the Gantry function.

Table 9-19 Gantry parameters

Parameters	Value/Meaning	Unit
Leading axis	Machine axis name of leading axis	–
Synchronized axis	Machine axis name of synchronized axis A gantry link between a linear and rotary axis or vice versa is not permissible. The synchronized axis must not be a geometry axis of the CPU axis. A synchronized axis cannot be declared as the leading axis in another gantry grouping.	–
Dissolve gantry grouping	No (default value) The gantry link is retained Yes The gantry link is cancelled and the axis synchronization is lost. The gantry axes can now be traversed individually. Important: This may lead to damage on mechanically coupled axes. This setting may be used only to correct a misalignment between the axes.	–
Limit value for warning	0 (default value) 0 to 100 If the difference in the actual position values of the leading and synchronized axes exceeds this value, error message “Gantry limit value for warning exceeded” is output and the interface signal “Gantry limit value for warning exceeded” (user DB, “AXy”, DBX30.3+m) is set. The axis coupling is not cancelled. Important: Value = 0 → Synchronization run is disabled, monitoring functions remain active.	[mm, degrees]

Table 9-19 Gantry parameters, continued

Parameters	Value/Meaning	Unit
Trip limit	0 (default value) 0 to 100 The trip limit must be greater or equal to the limit value for warning. The monitoring function is effective only when the gantry grouping is synchronized . If the difference in the actual position values of the leading and synchronized axes exceeds this value, the error "Gantry trip limit exceeded" is output and interface signal "Gantry trip limit exceeded" (user DB, "AXy", DBX30.2+m) is set. The gantry axes are shut down immediately.	[mm, degrees]
Trip limit for referencing	0 (default value) 0 to 100 The parameter "Trip limit for referencing" must be set higher or equal to the "Trip limit" parameter. The monitoring function is effective only when the gantry grouping is not synchronized . The reaction is analogous to parameter "Trip limit".	[mm, degrees]

Gantry interface signals

The interface signals are axis-specific. Their effect on the leading and synchronized axes is shown in the following table.

Table 9-20 Assignment of gantry interface signals for leading and synchronized axes

Interface signal	User DB, "AXy"	Leading axis	Synchronized axis
Start gantry synchronization run	DBX10.4+m	x	
Gantry trip limit exceeded	DBX30.2+m		x
Gantry limit value for warning exceeded	DBX30.3+m		x
Gantry synchronization run ready to start	DBX30.4+m	x	
Gantry grouping is synchronized	DBX30.5+m	x	
Gantry leading axis	DBX30.6+m	1	0
Gantry axis	DBX30.7+m	1	1

Effect of other interface signals

Axis signals to axis (CPU → FM 357-2):

As a basic rule, the axis signals always act on both axes in the gantry grouping. In this case, each gantry axis has equal priority.

If, for example, the leading axis sets interface signal Controller enable (user DB, "AXy", DBX2.1+m) to FALSE, the synchronized axis is shut down at the same time.

Table 9-21 Effect of individual interface signals on leading and synchronized axes

Interface signal	User DB, "AXy"	Leading axis	Synchroni- zed axis
Enable position controller	DBX2.1+m	on both axes	
Axial deletion of distance to go	DBX2.2+m	axial	axial
Feed Stop	DBX4.2+m	on both axes	
Hardware limit switch minus/plus	DBX7.0/1+m	on both axes (axial error message)	
2nd software limit switch minus/plus	DBX7.2/3+m	axial	axial

Axis signals from axis (FM 357-2 → CPU):

As a basic rule, the axis signals from axis to CPU are set axis-specifically in each case for the synchronized and leading axes.

Exception:

When the leading axis is moving, interface signal Travel minus/plus (user DB, "AXy" DBX23.6/7+m) is also set for the synchronized axis.

Control

The control response of the leading and synchronized axes must be identical, i.e. the following error of both axes must be identical at any given velocity.

The following position control parameters should be set optimally for the leading and synchronized axes (see also Section 9.3, Position control):

- Position loop gain
- Speed feedforward control
- Time constant for current control loop
- Weighting factor

The following position control parameters must be set identically for both axes:

- Jerk filter active
- Jerk time
- Acceleration pattern
- Jerk

Referencing and synchronization of gantry axes

The coupling between the gantry axes must be reliably maintained in all operating modes, including immediately after power ON.

If the leading or synchronized axis has an incremental encoder, the reference point must be approached after power ON with the coupling in tact and the slave axis then synchronized.

A special sequence has been implemented in the FM 357-2 for this purpose.

Misalignment on power ON

An offset may exist between the leading and synchronized axes after power ON. This offset is normally relatively small so that the axes can still be referenced and synchronized.

However, if the offset is excessive, e.g. due to an earlier disturbance, a compensatory motion must be undertaken. The gantry grouping must be dissolved for this purpose (by parameterization) and the axes corrected by the operator.

Notice

If the "Reference point approach" mode is not selected, a traversing motion may only be carried out if the interface signal "Gantry compound is synchronized" (user DB "AXy", DBX30.5+m) is set.

Referencing and synchronization process

Part 1: Referencing of leading axis (incremental encoder)

Referencing must be started in “Reference point approach” mode with interface signal “Direction plus or Direction minus” (user DB, “AXy” DBX4.7/6+m, see also Section 9.7, Referencing and alignment).

During this process, the slave axis travels **in synchronism** with the leading axis.

After the reference point has been recorded, interface signal “Referenced/synchronized” (user DB, “AXy”, DBX20.4+m) is set for the leading axis.

Part 2: Referencing of synchronized axis (incremental encoder)

The synchronized (slave) axis is then **automatically** referenced. The dependency between the leading and synchronized axes is switched over internally. **The leading axis travels in synchronism with the slave axis.**

After the reference point has been recorded, interface signal “Referenced/synchronized” (user DB, “AXy”, DBX20.4+m) is set for the synchronized axis and the correct gantry dependency relationship restored.

If the reference point approach has been interrupted (e.g. by Reset), the leading axis can be restarted to repeat the process.

To ensure that the shortest possible routes are traversed for referencing, parameter “Reference point coordinate” should be set identically for both gantry axes. The difference between the zero marker and reference point coordinate must be entered for each axis specifically in parameter “Reference point offset”.

Part 3: Synchronization (incremental and absolute encoders) in the “Reference-point approach” mode

The axes can be synchronized in two different ways depending on the difference in their actual values:

1. The difference is **less than** the setting in parameter “Limit value for warning”:

The synchronization run is started automatically and error message “Synchronization in progress for gantry grouping” is output. The gantry axes travel **with deactivated coupling** to the reference point coordinate of the leading axis at referencing velocity.

As soon as the gantry axes have reached the target position, interface signal “Gantry grouping is synchronized” (user DB, “AXy”, DBX30.5+m) is set and the gantry coupling reactivated. The synchronization run is thus finished.

2. The difference is **greater than** the setting in parameter “Limit value for warning”:

Interface signal “Gantry synchronization run ready to start” (user DB, “AXy”, DBX30.4+m) is set and the error message “Wait for gantry grouping synchronization start” is output.

The synchronization run must be activated by the CPU by means of interface signal “Start gantry synchronization run” (user DB, “AXy”, DBX10.4+m). The process then continues as described above.

If the synchronization run has been interrupted, it can be restarted with interface signal "Start gantry synchronization run" (user DB, "AXy", DBX10.4+m) if the following conditions are fulfilled:

- "Reference point approach" mode must be active
- Interface signal "Gantry grouping is synchronized" (user DB, "AXy", DBX30.5+m) = 0
- Interface signal "Gantry synchronization run ready to start" (user DB, "AXy", DBX30.4+m) = 1

When synchronization is restarted, the synchronized axis travels to the **current actual position of the leading axis** (not to the reference point coordinate!). .

Synchronization is lost if

- the gantry grouping is dissolved (by parameter "Dissolve gantry grouping")
- a gantry axis loses its reference point
- the gantry axes have been operating in follow-up, interface signal "Follow-up mode" (user DB, "AXy", DBX1.4+m)

In this case, the gantry synchronization run can be restarted directly with interface signal "Start gantry synchronization run" (user DB, "AXy", DBX10.4+m). The synchronized (slave) axis then traverses to the current actual position of the leading axis.

After referencing, the monitoring functions, working area limitation, software limit switches and protection zones are active and the limit values of both gantry axes are applied.

Initial start-up

You should observe the following sequence of operations for initial start-up **(incremental encoder)**:

1. Parameterization of gantry grouping and activation of parameters via parameterization tool. The parameter "Limit value for warning" must initially be set to 0. The synchronization run is then not started automatically. The monitoring functions remain active. Set parameter "Reference point offset" to 0.
2. Set identical positions for the leading and synchronized axes.
3. Select "Reference point approach" mode. Start referencing of leading axis. After error message "Wait for gantry grouping synchronization start" has appeared, determine the actual-value difference (see Figure 7-4, Servicing Data) between the two axes and enter it with negated sign in parameter "Reference point offset" of the synchronized axis and then activate the parameter.
4. Rereference the axes as described in point 3., the actual positions of both gantry axes must now be identical. Then check the dimensional offset of the two axes. You may need to correct parameter "Reference point coordinate" of the synchronized axis.
5. Start the synchronization run with interface signal "Start gantry synchronization-run" (user DB, "AXy", DBX10.4+m).

6. To determine the warning and trip limits, start by setting parameter "Limit value for warning" to a very low value and "Trip limit" and "Trip limit for referencing" to a high value.

Then place a very high dynamic load on the axes and set the limit value for warning such that the gantry can operate just below the trigger limit for error "Gantry limit value for warning exceeded". The calculated setting must, of course, lie within a technically meaningful range. A small safety margin must be added to the settings in parameters "Trip limit" and "Trip limit for referencing".

You should observe the following sequence of operations for initial start-up **(absolute encoder)**:

1. Parameterization of gantry grouping and activation of parameters via parameterization tool. The parameter "Limit value for warning" must initially be set to 0. The synchronization run is then not started automatically. The monitoring functions remain active.
2. Set identical positions for the leading and synchronized axes.
3. Align the following axis as described in Section 9.7.3 steps 1. to 4.
4. Align the leading axis as described in Section 9.7.3 steps 1. to 7.

When you press the travel direction key, the encoder alignment status and the indicator for leading and following axis are set.

5. Start the synchronization run with interface signal "Start gantry synchronization run" (user DB, "AXy", DBX10.4+m).
6. To determine the warning and trip limits, start by setting parameter "Limit value for warning" to a very low value and "Trip limit" and "Trip limit for referencing" to a high value.

Then place a very high dynamic load on the axes and set the limit value for warning such that the gantry can operate just below the trigger limit for error "Gantry limit value for warning exceeded". The calculated setting must, of course, lie within a technically meaningful range. A small safety margin must be added to the settings in parameters "Trip limit" and "Trip limit for referencing".

Automatic synchronization

With software version 5 and higher, an automatic synchronization is also possible if the system is not in the "Reference-point approach" mode.

When switching on the position control, the synchronism of master and synchronized axes are provided automatically if the difference of the actual values is less than the parameter "Standstill range".

During this process, the difference of the actual values is specified for the position controller of the synchronized axis without interpolation. If the parameter "Standstill range" is too large, e.g. error 25060 (Axis ... Setpoint limitation) can occur.

9.16.3 Master value coupling

General

This function uses a curve table to couple the position of a slave axis to the position of a master axis. The functional relationship between the master and slave axes are defined in the curve table.

The coupling can be activated or deactivated directly from the NC program or via synchronized actions.

The slave axis should have the same or better control dynamic response as the master axis.

Master axis and slave axis

Various types of master value coupling are available. You can set the position value of the master axis that is to be applied as the input variable for the curve table in parameter "Type of master value coupling".

Parameters	Value/Meaning	Unit
Type of master value coupling	<p>Actual value Master value is the encoder actual value</p> <p>Setpoint (default value) Master value is the setpoint position calculated by the interpolator.</p> <p>Simulated master value A master value is calculated from an external actual value (encoder input) (see parameter "External master value", Section 9.1)</p>	—

With an actual-value coupling, mechanical disturbances in the master axis can be transmitted to the slave axis. In such cases, a setpoint coupling affords a better synchronization run.

If the master value is to be derived from an external motion (external master), the corresponding actual value must be signalled via an encoder input on the FM. Parameter "External master value" must be set for this axis. "Simulated master value" should be configured as the type of master value coupling in the slave axis, and in the master axis this should be "Actual value".

A coupling may comprise only one master and one slave axis. The slave axis is assigned to the master by programming measures when the master value coupling is activated. When the coupling is active, the slave axis is traversed **solely** via the master value coupling.

Any axis (e.g. CPU axis, positioning axis, path axis) can be declared as the master in either an actual-value or setpoint coupling.

Reaction to interface signals

The only interface signals which affect a coupled slave axis are those with initiate a motion stop:

- Controller enable (user DB "AXy" DBX2.1+m)
- Feed stop (user DB "AXy" DBX4.3+m)
- Stop (user DB "FMx" DBX108.1+n)

The response of the master axis to interface signals does not change when the master value coupling is active.

Parameter "Master value coupling remains active" can be set to control the reaction to Reset (user DB "FMx" DBX108.7+n) and end of program.

Parameters	Value/Meaning	Unit
Master value coupling remains active	<p>No (default value) All active master value couplings are cleared on Reset or end of program.</p> <p>Yes On Reset or end of program, all active master value couplings remain valid (even on changeover to "Jog" or "Incremental travel relative" modes)</p>	—

If the master value coupling has been activated via a static synchronized action (see Section 10.32, Synchronized actions), this parameter has no effect at a program end. The relevant synchronized action, and thus also the master value coupling, remain valid.

Programming

The function must be programmed by means of the following instructions (see Section 10.34):

```

CTABDEF(FA, LA, CTAB No, TYP) ; Start of curve table definition
CTABEND ; End of curve table definition
CTABDEL(CTAB No) ; Deletion of a curve table

as from software version 5:
CTABDEL(CTAB-Nr, CTAB-Nr) ; Deletion of the curve table range
CTABDEL() ; Deletion of all curve tables

LEADON(FA, LA, CTAB No ) ; Activation of coupling
LEADOF(FA, LA ) ; Deactivation of coupling
FA ; Slave axis
LA ; Master axis
CTAB No ; Number of curve table
TYP ; Curve table characteristics
; 0: Curve table is not periodic
; 1: Curve table is periodic (master value)
; 2: Curve table is periodic
; (master and slave value)

```

Curve table

The functional relationship between an input variable, i.e. the master axis position, and an output variable, i.e. the slave axis position, is defined in the curve table.

The relationship is defined in the NC program in the form of motion instructions of the master and slave axes (see Section 10.34). The FM calculates curve segments and curve table polynomials – the internal form of representation of the programmed contour. A curve segment corresponds to an NC block and can contain up to 3 curve table polynomials.

Curve tables are stored in the static memory (program memory) under their number (CTAB No) and retained after power OFF. You must use the following parameters to reserve memory space for this purpose.

Parameters	Value/Meaning	Unit
Number of curve tables	0 (default value) 0... 100 Maximum number of curve tables in FM memory	–
Number of curve segments	0 (default value) 0 to 1 800 Maximum number of all curve segments (see Section 10.34)	–
Number of curve table polynomials	0 (default value) 0 to 3,600 Maximum number of all curve segments; a maximum of 3 polynomials are required for each curve segment.	–

Notice

If this parameter is changed, all user data will be lost due to reorganization of the FM memory. Save your data beforehand if necessary!

When you activate the master value coupling (LEADON), select a curve table by entering a CTAB No. You can activate any curve table for any axis combination of master and slave axes. The axes entered after CTABDEF are required for a syntax check during curve table generation.

The input and output variables of the curve table can be offset and scaled.

Table 9-22 Offset and scaling of master and slave axis positions

Parameters	Value/Meaning	Unit
Master axis position offset	0 (default value) –100 000 000 to + 100 000 000 Sets a master axis position offset (input variable for curve table)	[mm], [degrees]
Scaling of master axis position	1 (default value) –1 000 000 to + 1 000 000 Factor for master axis position (input variable for curve table)	–
Slave axis position offset	0 (default value) –100 000 000 to + 100 000 000 Sets a slave axis position offset (output variable for curve table)	[mm], [degrees]
Scaling of slave axis position	1 (default value) –1 000 000 to + 1 000 000 Factor for slave axis position (output variable for curve table)	–

The following axis position is calculated according to the following equation:

$$\mathbf{FAS} = \mathbf{OFFSET_FA} + \mathbf{SCALE_FA} * \mathbf{CTAB}(\mathbf{OFFSET_LA} + \mathbf{SCALE_LA} * \mathbf{LA})$$

FAS ; Slave axis position (setpoint)
OFFSET_FA ; “Slave axis position offset” parameter
SCALE_FA ; “Scaling of slave axis position” parameter
LA ; Master axis position
OFFSET_LA ; “Master axis position offset” parameter
SCALE_LA ; “Scaling of master axis position” parameter
CTAB ; Curve table

The curve table can only supply new values for the slave axis within the definition range of the master axis.

The following applies: $LA_{\min} \leq (\mathbf{OFFSET_LA} + \mathbf{SCALE_LA} * \mathbf{LA}) \leq LA_{\max}$

Parameters **OFFSET_FA** and **SCALE_FA** can shift the slave axis position out of the definition range of the curve table.

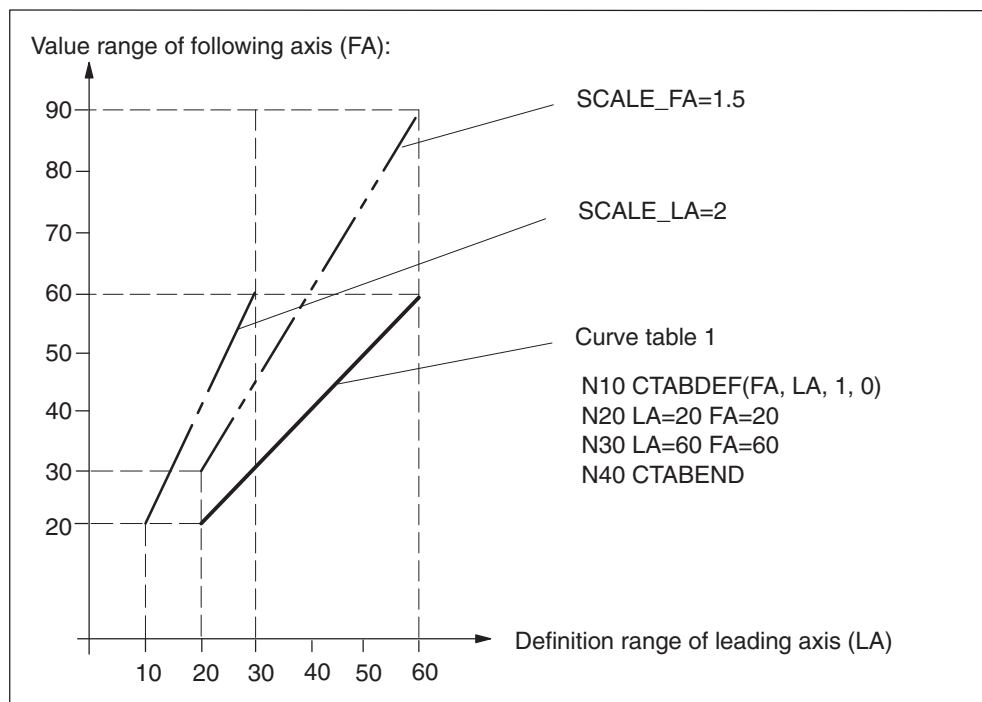


Fig. 9-35 Example of scaling of master and slave axis positions

System variables can be used to modify the parameters for scaling and offset from the NC program. When a coupling is active and synchronized, writing these system variables causes the new slave axis position to be approached immediately.

Assignment of system variables to parameters:

`$SA_LEAD_OFFSET_IN_POS[FA]` ; "Master axis position offset" parameter
; acts on the phase

`$SA_LEAD_SCALE_IN_POS[FA]` ; "Scaling of master axis position" parameter
; acts on the frequency

`$SA_LEAD_OFFSET_OUT_POS[FA]` ; "Slave axis position offset" parameter
; acts as an average offset

`$SA_LEAD_SCALE_OUT_POS[FA]` ; "Scaling of slave axis position" parameter
; acts on the amplitude

Note:

The scale and offset for the master axis must always be configured and programmed for the slave axis.

Parameters overwritten by the NC program do not become effective in the parameterization tool until the machine data have been read out.

Activation and deactivation of master value coupling

The coupling is activated with instruction LEADON(...).

The slave axis is not required to have the position and velocity specified by the curve table at the moment of activation. The coupling is set up by a synchronization run.

The following cases may apply after activation:

Case 1:

The master axis is outside the definition range of the curve table.

The synchronization run does not start until the master axis has entered the definition range.

Case 2:

The master axis is within the definition range of the curve table. The slave axis position calculated from the curve table is approaching the actual position of the slave axis.

The slave axis waits for the “approach” to the position specified by the curve table. It is set in motion as soon as the distance to the specified position can be travelled at the permissible axis velocity (BRISK). The slave axis moves **only** in the direction specified from the curve table.

Case 3:

The master axis is within the definition range. The slave axis position calculated from the curve table is moving away from the actual position of the slave axis.

The synchronization run does not being, the slave axis remains stationary.

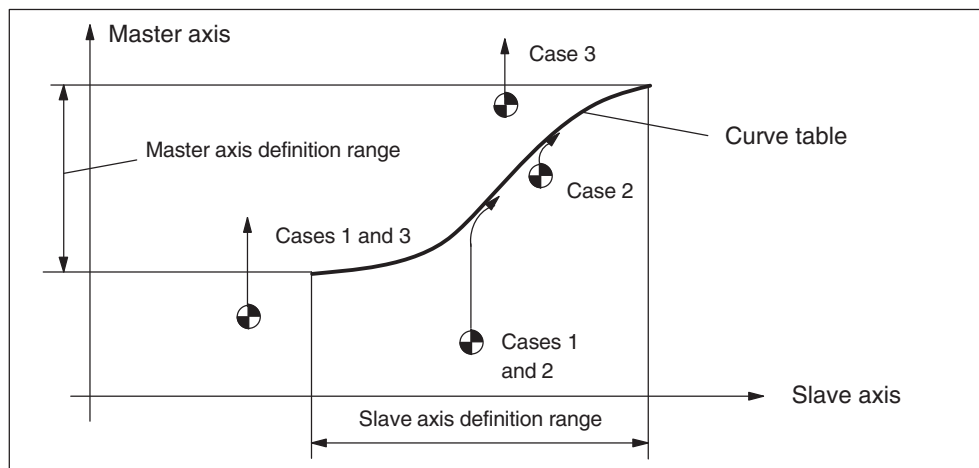


Fig. 9-36 Example of synchronization

In the example, the master axis is traversed by a motion instruction from the NC program (e.g. G01). The slave axis is traversed via the master value coupling.

The “Threshold for coarse and fine synchronism” parameters can be applied to monitor the coupling.

Depending on the current status, the following signals are sent to the CPU:

- Fine synchronism (user DB “AXy”, DBX28.0+m)
- Coarse synchronism (user DB “AXy”, DBX28.1+m)

Parameters	Value/Meaning	Unit
Threshold value for coarse synchronism	1 (default value) 0 to 10 000 Difference between actual values of master and slave axes for coarse synchronism status.	[mm], [de- grees]
Threshold value for fine synchronism	0.5 (default value) 0 to 10 000 Difference between actual values of master and slave axes for fine synchronism status.	[mm], [de- grees]

The status of synchronism can also be read in the NC program from system variable \$AA_SYNC[...]:

- 0: Not synchronized
- 1: Coarse synchronism
- 2: Fine synchronism
- 3: Coarse and fine synchronism

The master value coupling is deactivated by instruction LEADOF (...). The slave axis is stopped if LEADOF is programmed directly in the NC program.

The function can be activated and deactivated “on the fly” by synchronized actions, i.e. while the master and slave axes are moving (see Section 10.32).

9.16.4 Electronic gear

General

The function “Electronic gear” (EG) is intended to couple the motion of a following axis to the motion of four master axes as the maximum. Gear cascading is permitted, i.e. a following axis can be master axis of a series-connected electronic gear. A following axis can continue to be traversed using the NC program; the portion of the EG motion is overlaid in this case.

The functional interrelation between master axis and following axis is defined either by a coupling factor or a curve table. Portions of motions of several master axes act to the following axis motion additively.

The motions of the following axis can be derived either from the setpoint or from the actual value of the master axis.

The function “Electronic gear” **cannot** be programmed using synchronized actions.

The max. number of the active EGs must be defined via parameterization.

This function is only available for the FM 357-2 LX, software version 5 and higher.

Parameter	Value/meaning	Unit
Number of electronic gears	0 (default value) 0...4 Max. number of electronic gears in the channel	—

Programming

EGDEF(FA, LAi, TYPi, ...) ; Define EG group

EGON(FA, SW, LAi, Zi, Ni, ...) ; Activate EG

EGONSYN(FA, SW, SP_FA, LAi, SP_LAi, Zi, Ni, ...) ; Activate EG with specification
; of the synchronized position

EGONSYNE(FA, SW, SP_FA, AM, LAi, SP_LAi, Zi, Ni,...) ; Activate EG with specifcation of
; synchronized position and approach
; mode for modulo master and
; following axes

EGOFS(FA) ; Deactivate all EGs of a
; following axis

EGOFS(FA, LAi, ...) ; Deactivate individual EGs

EGDEL(FA) ; Delete the definition of an
; EG group

Parameters:

i Number of master axes: $i = 1 \dots 4$

Type Setpoint or actual-value coupling:

Type = 0 Actual-value of the master axis

Type = 1 Setpoint of the master axis

FA Following axis

LA Master axis

Z Numerator of coupling factor

N Denominator of coupling factor

SW Block change mode: (string parameters in inverted commas !)

“NOC”: Block change is carried out immediately

“FINE”: Block change is carried out at fine synchronization (default)

“COARE”: Block change is carried out at coarse synchronization

“IPOSTOP”: Block change is carried out at setpoint synchronization

AM Approach mode: (string parameters in inverted commas !)

“ACN”: Absolute dimension, negative approach direction

“ACP”: Absolute dimension, positive approach direction

“DCT”: Absolute dimension, time-optimized approach

“DCP”: Absolute dimension, path-optimized approach

“NTGT”: Next tooth gap, time-optimized approach (default)

“NTGP”: Next tooth gap, path-optimized approach

SP_FA Synchronized position of the following axis

SP_LA Synchronized position of the master axis

Define EG – EGDEF

The statement EGDEF assigns a following axis at least one and a maximum of 4 following axes.

The type of coupling can be different for each master axis.

A coupling may not yet be defined for the following axis.

EGDEF is the precondition to enable the EG.

Delete EG – EGDEL

An EG defined with EGDEF can be deleted with EGDEL.

Activate EG – EGON

With this command, the EG is turned on immediately. The current position of master and following axes are to be considered the synchronized EG positions. This behavior is comparable with the function “Coupled motion”.

The block change module defines when the block that contains the statement EGON or EGONSYN/E is ended. If this parameter is not specified, the default value “FINE” applies.

With execution of the statement EGON or EGONSYN/E, the interface signal “Following axis active” (AW-DB “AXy”, DBX23.1+m) is set.

Activate EG – EGONSYN

The programmed synchronized positions for master and following axes define the point at which the EG reaches the synchronous condition.

If the EG is not yet synchronized at the time when the statement EGONSYN/E is provided, the following axis is traversed via motion overlay to the synchronized position. The difference of the master axis to the programmed synchronized position is taken into account in the synchronized position of the following axis on the basis of the kinematic law of the EG (example, see Section 10.35).

During the motion overlay of the following axis, the interface signal “Overlaid motion” (user DB “AXy”, DBX28.4+m) is output. The FM expects the interface signal “Enable following axis overlay” (“AXy”, DBX8.4+m).

Otherwise, the error message 16 771 “Overlaid motion not enabled” is output and the FM waits for enabling.

The motion overlay of the following axis is independent of the override.

If the EG contains modulo axes, their positions are reduced modulo. This guarantees that in any case the next possible synchronized position is approached, i.e. a “relative” synchronization is provided.

Activate EG – EGONSYNE

This function can be used if master and following axes are modulo axes.

In addition to the synchronized position, an approach mode can be programmed.

For further information, please refer to Section 10.35.

EG via curve table

If you wish the coupling to be carried out using a curve table, the denominator $N = 0$ and the numerator Z must be set to the number of the curve table.

Contrary to the master value coupling (LEADON), the output of the curve table acts **additively** to the synchronized position of the following axis:

$$FAS = SP_FA + (OFFSET_FA + SCALE_FA * CTAB(OFFSET_LA + SCALE_LA * LA))$$

FAS ; Following axis position (setpoint)
SP_FA ; Synchronized position of the following axis

Curve table:

OFFSET_FA ; Parameter "Offset to following axis position"
SCALE_FA ; Parameter "Scaling of the following axis position"
LA ; Master axis position
OFFSET_LA ; Parameter "Offset to the master axis position"
SCALE_LA ; Parameter "Scaling of the master axis position"
CTAB ; Curve table

If an offset to the following axis position (OFFSET_FA) is parameterized, this is added to the synchronized position of the following axis.

$$SP_FA = SP_FA + OFFSET_FA$$

With EGON, this new synchronized position is also approached according to EGONSYN.

Synchronization

When the synchronized position of master and following axes is reached, the following axis is coupled to the master axis via the EG. The FM calculates a setpoint for the following axis from the actual or setpoint position of the master axis/axes on the basis of the kinematic law of the EG.

If the EG is activated specifying synchronized positions (EGONSYN), the following axis must reach its synchronized position simultaneously with the master axis without dynamic overload.

Depending on the **synchronism difference**, the following signals are reported to the CPU:

- Synchronization running (user DB "AXy", DBX23.4+m)
Synchronism difference > threshold value for coarse synchronism
- Coarse synchronism (user DB "AXy", DBX28.1+m)
Synchronism difference < threshold value for coarse synchronism
- Fine synchronism (user DB "AXy", DBX28.0+m)
Synchronism difference < threshold value for fine synchronism

Synchronism difference:

Difference of the actual value of the following axis to the value resulting from the actual values of the master axes according to the kinematic law of the EG.

Parameter	Value/meaning	Unit
Threshold value for coarse synchronism	1 (default value) 0...10,000 Synchronism difference	[mm], [deg]
Threshold value for fine synchronism	0.5 (default value) 0...10,000 Synchronism difference	[mm], [deg]

The synchronism difference can be read in the NC program from the system variable \$VA_EG_SYNCDIFF[...] absolutely or from \$VA_EG_SYNCDIFF_S[...] with sign.

Threshold values for acceleration and maximum velocity

Depending on the parameters “Threshold value: Axis accelerating” and “Velocity and acceleration warning threshold“, the following interface signals are output to the CPU for the following axis:

- “Axis accelerating” (user DB “AXy”, DBX23.3+m)
- “Warning threshold: Velocity reached“ (user DB “AXy”, DBX28.5+m)
- “Warning threshold: Acceleration reached“ (user DB “AXy”, DBX28.6+m)

Parameter	Value/meaning	Unit
Threshold value: Axis accelerating	90 (default value) 0...100 Percentage with regard to the parameter “Acceleration”	%
Velocity and acceleration warning threshold	25 (default value) 0...100 Percentage value with regard to the parameter “Maximum velocity” or “Acceleration”	%

Gear cascading

EGs can be connected in series. The following axis of an EG can be master axis of another EG. A following axis, however, cannot be master axis of its own EG (feedback).

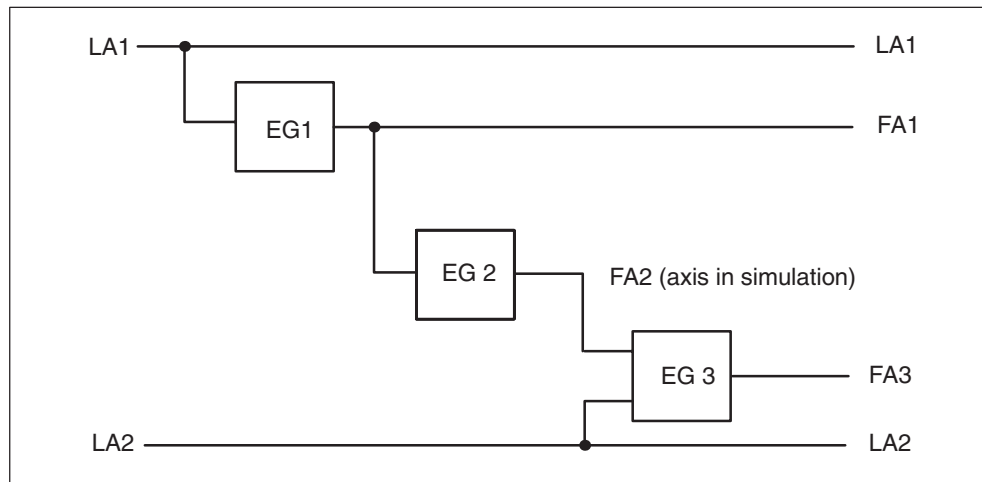


Fig. 9-37 Gear cascading

The synchronism difference always refers to the input and output of **one** EG.

Example: EG3:

Synchronism difference = actual value FA3 – EG3 (actual value FA2, actual value LA2)

Deactivate EG – EGOFS

An EG can be switched off partially or completely.

EGOFS(FA)

The EG is switched off completely; the following axis decelerates up to standstill.

EGOFS(FA, LA1, ...)

Only the specified master axes are switched off. The following axis continues traversing, depending on the remaining master axes.

Behavior in case of Power On, Reset and mode change

After Power On, no EG is active.

An active EG remains active even after Reset and mode change.

9.16.5 Tangential control

General

A rotary axis is coupled to the tangent of a contour generated from two geometry axes.

The rotary axis is the slave axis of the coupled axes and traverses depending on the current position of the contour tangent of the master axes (geometry axes).

The group of coupled axes can be turned on / turned off by appropriate program statements and an angle between contour tangent and rotary axis be specified.

Tangential control is available for the FM 357-2LX version.

Axis types and axis motion

Both master axes must be geometry axes and thus linear axes. The master axes have to be programmed as path axes (G1, G2 ...); any motions as a positioning axis (POS) will not be taken into account.

The slave axis must be a rotary axis and thus a supplementary axis. This can be positioned in addition to the group of coupled axes (POS[...]). The position acts as an additional offset angle.

Slave and master axes can be simulated and/or real axes.

The coupled axes have to belong to one and the same channel.

Each rotary axis can be assigned a tangential control.

Parameters

Parameters	Value/Meaning	Unit
Tangential control remains active	no: Tangential control is canceled with program end/reset (default value) yes: Tangential control is kept after program end/reset	–
Intermediate block limit angle	5.0 (default value) 0...360	[deg]

Programming

Statements with regard to programming the tangential control (see Section 10.9):

TANG (slave axis, master axis1, master axis2, coupling factor)

; Definition of the tangential control

TANGON(slave axis, angle)

; Turning on of the tangential control

TANGOF(slave axis)

; Turning off of the tangential control

TLIFT(slave axis)

; Permit interm. block at contour corners

Coupling factor Interrelation between change of the angle of the contour tangent and the slave axis; default value is 1.

Angle Offset angle between contour tangent in the direction of movement and the slave axis

System variable for the status of the coupling:

$\$AA_COUP_ACT[axis]=0$

; no coupling active

$\$AA_COUP_ACT[axis]=1, 2, 3$

; tangential control active

Coordinate system

The tangential control refers to the workpiece coordinate system (WCS).

The two master axes define a rectangular coordinate system.

Angular position of the slave axis

The angle of the slave axis is at any time composed of the angle of the contour tangent, the offset angle programmed with TANGON and – in some cases – of an angle programmed via POS[FA].

The angles of the contour tangent and of the slave axis are generated in the mathematically positive direction.

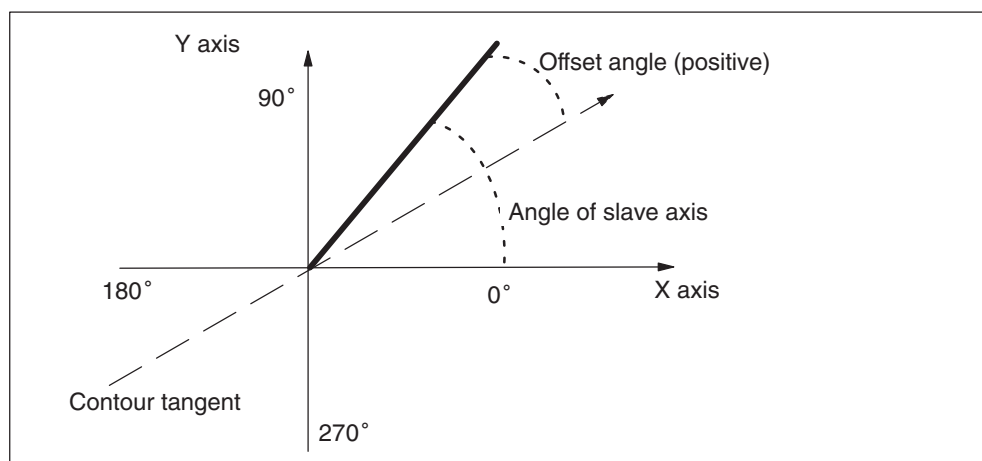


Fig. 9-38 Tangential control, angular position of the slave axis

Behavior in the modes

The following behavior can be noticed in the individual modes:

- **Activation**
Tangential control is only active in the modes "Automatic" and "MDI".
- **Referencing**
When referencing the slave axis, the couplings are disabled.
- **Default position after power-up**
After power-up, no tangential control is active.
- **Behavior after reset/end of program**
It is possible to set via parameterization whether the active tangential couplings are canceled or kept after reset/end of program.

Behavior at contour corners

The following behavior can be selected at contour corners:

- **No TLIFT(FA) programmed after TANG(...)**
The path velocity of the master axes is reduced such that the slave axis reaches its target position synchronously with the two master axes.
- **TLIFT(FA) programmed after TANG(...)**
An intermediate block is inserted at the corners whose angle is greater than the parameter "Intermediate block limit angle". The intermediate block will traverse the slave axis as fast as possible to the position corresponding to the contour tangent after the corner, observing all limit values.
This behavior can be switched off by programming TANG without TLIFT again.

Example

Offset angle 90 degrees, TLIFT programmed.

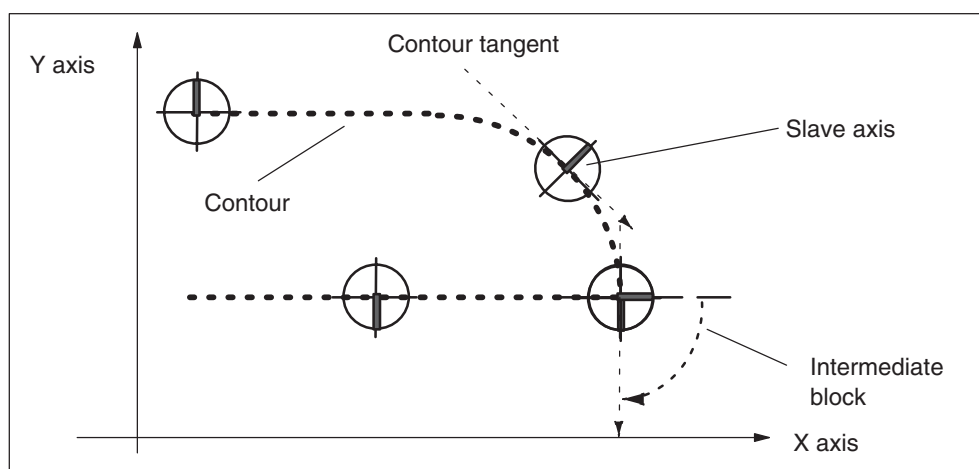


Fig. 9-39 Tangential control with intermediate block

Behavior in the case of direction reversal of the master axis motion

A direction reversal of the master axis motion results in a direction reversal of the contour tangent and thus in a “180 degrees rotation” of the slave axis.

This behavior is not always reasonable in practice and can be prevented by using the work area limitation (G25, G26, WALIMON) at the time of the direction reversal.

If the offset angle of the slave axis is outside the work area limitation after the direction reversal, it is tried with the opposite offset angle to come into the permissible work area.

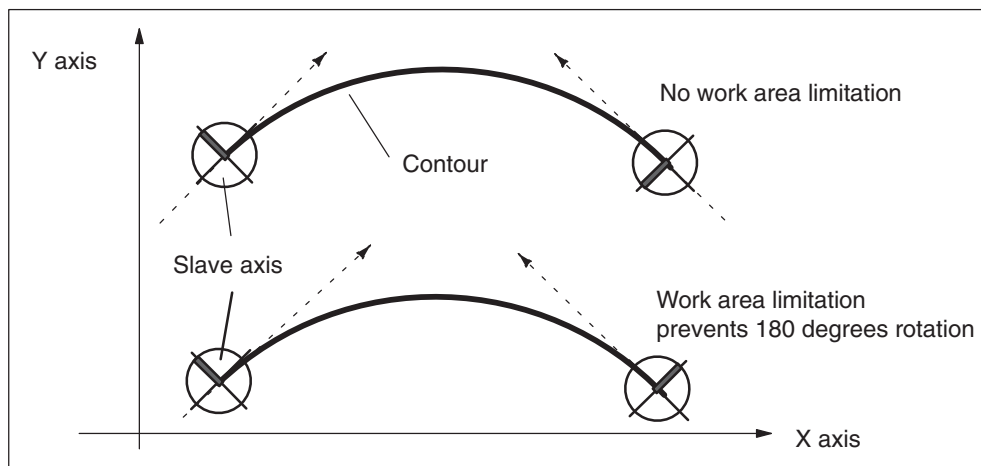


Fig. 9-40 Tangential control; direction reversal with and without work area limitation of the slave axis

To suppress the 180 degrees rotation over several motion blocks, the upper and lower limit values may have to be adapted accordingly. With circles, a division into individual segments may be required.

9.16.6 Overlaid motion in synchronized actions

General

You can start an overlaid motion in the action part of synchronized actions (see Section 10.32) by programming system variable `$AA_OFF[axis]`. The motion acts internally as a compensation value in the machine coordinate system. The overlaid motion commences immediately, irrespective of whether the relevant axis is traversing in response to programming or not.

You can use this function, for example, to implement a clearance control.

The function is available for the FM 357-2LX version.

Parameters

Velocity, upper limit and calculation method must be set in the following parameters.

Parameters	Value/Meaning	Unit
Calculation of compensation value	<p>Absolute (default value) The \$AA_OFF value is calculated as an approach position.</p> <p>Integral The \$AA_OFF value is calculated as a path section, further \$AA_OFF values are added up to obtain a total compensation value.</p>	–
Upper limit of compensation value	<p>100 000 000 (default value) 0 to 100 000 000</p> <p>This parameter limits the compensation value to be traversed. Applied to total compensation value with integral calculation.</p>	[mm], [degrees]
Velocity of compensation value	<p>1 000 (default value) 0...axis velocity</p> <p>The velocity at which the compensation value is applied.</p> <p>If parameter setting "Axis velocity" is altered, the compensation value velocity is altered accordingly as a percentage.</p>	[mm/min], [rev/min]

Programming

\$AA_OFF[axis] ; System variable for overlaid motion
 \$AA_OFF_LIMIT[axis] ; Limit for overlaid motion
 ; 0: Not reached
 ; 1: Reached in positive direction
 ; 2: Reached in negative direction

The \$AA_OFF setting is ignored in the NC program, i.e. the value offsets all positions in the machine coordinate system for the relevant axis.

System variable \$AA_OFF_LIMIT can be used for troubleshooting purposes.

Example:

```

N05 POS[X]=0 POS[Y]=20
N10 ID=1 WHENEVER ($AA_IW[X] >= 50) AND ($A_IN[9]== TRUE)
      DO $AA_OFF[Y]= - R11
N20 POS[X]=100 FA[X]=500
  
```

From position X50, the Y axis is traversed by way of an overlaid motion by an amount from R11 if the sensor input switches to TRUE. The offset value must be integrally calculated, i.e. as long as the condition is fulfilled, a new value is preset cyclically (in the IPO cycle) until the probe switches to FALSE. The compensation is visible only in the machine coordinate system.

9.17 Measurement

General

For directly connected axes (servo or stepper motor drive), the two sensing probe inputs of the FM357-2 must be used. When a sensor probe switches, the axis position is acquired in the hardware through readout of an actual-value counter and stored in a system variable.

There are delays of 15 ms on the rising edge and 150 ms on the falling edge of the probe (when connecting to the front connector X1). The measurement uncertainty depends on this delay time and the approach speed to the sensor probe.

This measurement method is coupled to the appropriate axis and is therefore also called local measuring.

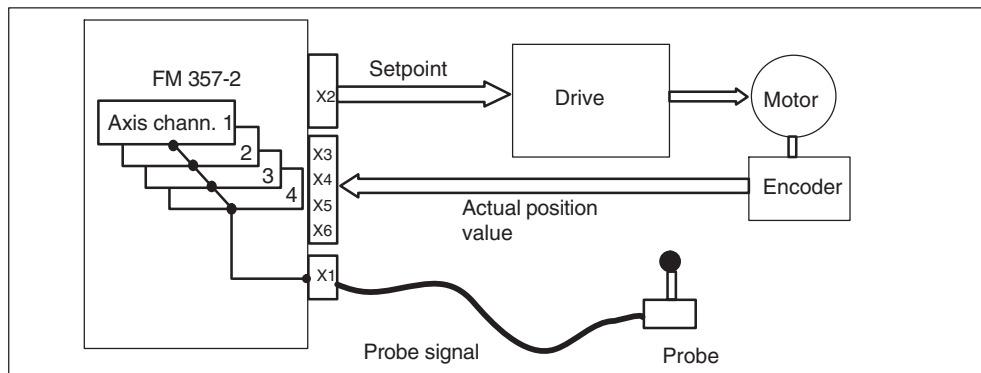


Fig. 9-41 Connection of the sensing probe to the FM357-2; local measuring

With PROFIBUS axes, the sensing probe can be connected directly to the drive (Fig. 9-42). The measured value is generated in the drive and is transferred to the FM357-2. This procedure is also coupled to the axis and thus constitutes a local measuring method.

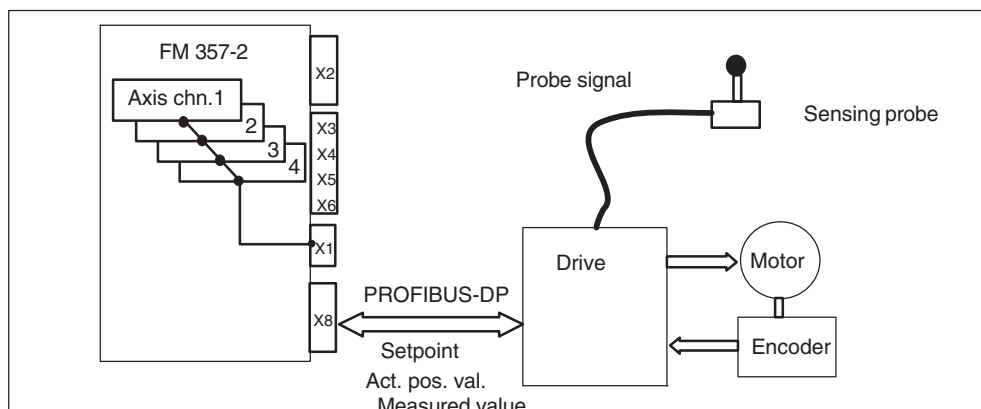


Fig. 9-42 Connection of the sensing probe to the PROFIBUS drive; local measuring

If you wish to carry out measuring with a probe in several axes, additional wiring expenditure to the next PROFIBUS drive or to the FM357-2 for servo or stepper motor axes (Fig. 9-43) is required.

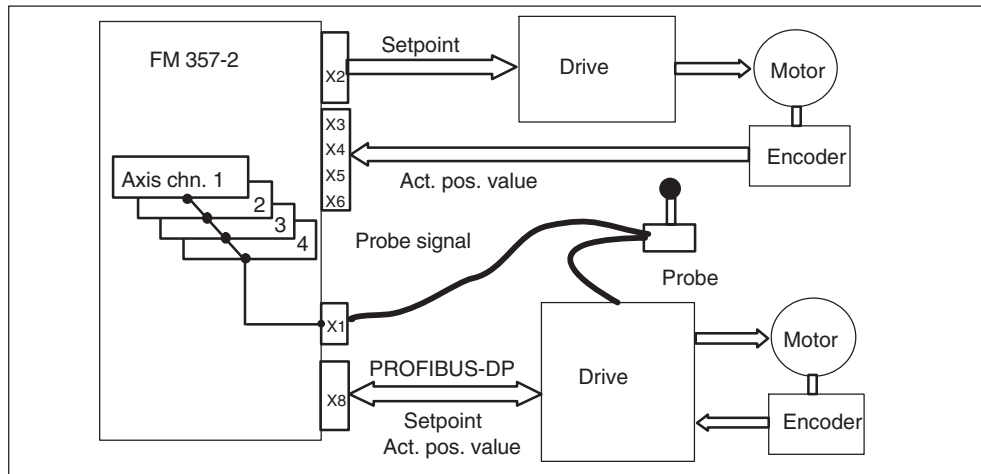


Fig. 9-43 Connection of the sensing probe to the PROFIBUS-DP drive and to the FM357-2; local measuring

The expenditure of additional sensing probe wiring is eliminated by another measuring method, the global measuring. The probe input of the FM357-2 is used exclusively.

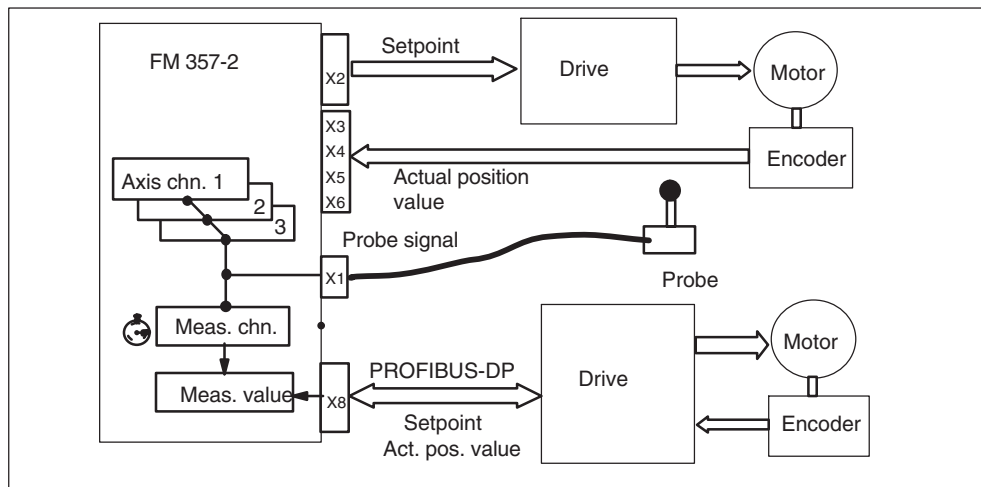


Fig. 9-44 Connection of the sensing probe to the FM357-2; global measuring for PROFIBUS axes

Global measuring

Global measuring uses an axis channel of the FM357-2 virtually as a stop watch. At the moment of the measurement event, the measuring time and the measuring position resulting from interpolation is determined for all axes involved in the global measurement.

Since only one measuring channel is used, only one measuring process can take place at a time for all axes involved in the global measurement. Local and global measuring in one measuring process are possible.

The accuracy of this measuring method is comparable with the direct measuring method. The measuring error is dependent on the acceleration at the time of measuring.

Global measuring occupies the 4th axis channel, i.e. only 3 of 4 possible axes can be used as direct axes (servo or stepper motor). A PROFIBUS-DP axis can also be assigned to the 4th axis channel if global measuring is active.

The programming of the measuring function is independent of the measuring method used. In global measuring, the above mentioned restrictions must be observed.

Parameters	Value/Meaning	Unit
Global measuring	OFF (default value) No global measuring ON Global measuring	–

Connecting the probes

The FM 357-2 is equipped with on-board inputs for connection of the probe (see Section 4.8):

	Connection
Measuring pulse input 1	X1, pin 26
Measuring pulse input 2	X1, pin 27
Switching response of inputs:	0 V (non-deflected state) 24 V (deflected state)

The measured value is recorded for all directly linked axes.

When using servo drive via PROFIBUS DP, the measuring probe has to be connected directly to the drive.

Programming the measuring function

The measurement function is programmed in the NC program by means of the following instructions (see Section 10.10):

Block-related measurement:

MEAS= ± 1 (± 2) ; Measure and delete distance to go
 MEAW= ± 1 (± 2) ; Measure and do not delete distance to go
 \$AA_MM[axis] ; System variable for measurement result in MCS
 \$AA_MW[axis] ; System variable for measurement result in WCS
 \$AA_MEA[n] ; Measuring job status, n = number of measuring input

Axial measurement (the function is available for the FM 357-2LX):

MEASA[axis]=(mode,TE_1,..., TE_4) ; Axial measurement with deletion of distance to go
 MEAWA[axis]=(mode,TE_1,..., TE_4) ; Axial measurement without deletion of distance to go
 MEAC[axis]=(mode,memory,TE1,...,TE_4) ; axial measuring continuously (SW 5.2 and higher)
 \$AA_MM1...4 [axis] ; Measured value of trigger event 1...4 in machine coordinate system
 \$AA_MW1...4 [axis] ; Measured value of trigger event 1...4 in workpiece coordinate system
 \$A_PROBE[n] ; Probe status,
 n = Number of measuring input
 \$AA_MEAAct[axis] ; Status of axial measurement

Interface signals

The status of the probe inputs is displayed via signals:

- Interface signal "Probe 1 actuated" (user DB, "FMx", DBX30.0)
- Interface signal "Probe 2 actuated" (user DB, "FMx", DBX30.1)
- Interface signal "Measurement active" (user DB, "AXy", DBX22.3+m)

Sequence of operations

To take measurements, the following sequence of operations must be programmed:

- Program the measuring function (with MEAS, MEAW). The measuring function is now activated.
- Program the traversing movement. The probe must operate within the movement.
- Program STOPRE to obtain the current measurement values (see Section 10.12 and 10.10).
- Process the measured value.

Hot-spot measurement

A "hot-spot" can be defined for the detection of the probe signal edge (see Section 9.11.5).

9.18 Travel to fixed stop

General

The “Travel to fixed stop” function (FXS = Fixed Stop) can be used to produce defined clamping forces, e.g. such as those required to grip parts.

The fixed stop can be approached by a path or positioning motion. When the stop is reached, the motion is aborted and the FM maintains the specified clamping torque until the function is terminated with FXS=0.

The function is available for the FM 357-2LX.

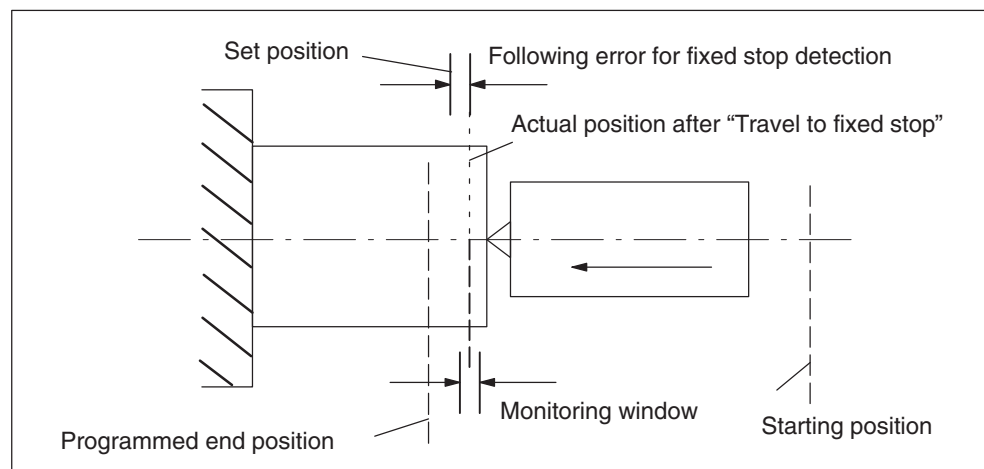


Fig. 9-45 Example of travel to fixed stop

Requirements of the drive

The “Travel to fixed stop” function can be used **only** for axes with analog or PROFIBUS-DB drives.

Demands placed on analog drives:

- Drives with torque limitation
- Drives for programmable pressure forces or torques which can switch between speed and torque control without sign reversal.

e.g.: SIMODRIVE 611-A

Travel to fixed stop **cannot be used** on:

- Vertical axes without weight compensation
- Gantry axes
- Positioning axes controlled from the CPU.

Table 9-23 Parameters for travel to fixed stop, continued

Parameters	Value/Meaning	Unit
Clamping torque	5 (default value) 0 to 100 % value of max. motor torque (% of max. current setpoint on FDD) This parameter takes effect when the fixed stop has been reached or acknowledged. For "Travel to fixed stop" with e.g. SIMODRIVE 611-A and a fixed , the torque limit setting in the drive should be identical to the setting in parameter "Torque limit for fixed stop approach".	[%]
Torque limit for approach to fixed stop (with analog drives)	5 (default value) 0 to 100 % value of max. motor torque This value is applied during the approach motion to the fixed stop. It must be the same as the torque limit set on the drive. The control uses it to limit the acceleration rate as well as the torque for switchover between speed-, current- and torque-controlled modes.	[%]
Error messages: <ul style="list-style-type: none"> • Axis has not reached the fixed stop • Travel to fixed stop aborted 	yes (default value) Error message is output no Error message is not output	–

9.18.2 Analog drive

General

The following section describes special features associated with the “Travel to fixed stop” function using the example of an analog SIMODRIVE 611-A drive.

For an exact description of how to start up the drive, please refer to the following documentation:

Installation and Start-Up Guide *SIMODRIVE 611-A*

Order No.: 6SN 1197-0AA60-0BP4

Fixed clamping torque

A fixed current limitation is preset in the drive via a resistor circuit (or via R12). This limitation is activated by the CPU via an output that is applied to terminal 96 on the drive. It is therefore possible to ensure that a fixed clamping torque is produced by the axis.

Setpoints can be injected via terminals 56/14 or 24/20 on the drive.

Programmable clamping torque

In this case, the CPU switches the drive from speed- to current-controlled operation as soon as the fixed stop is reached. When terminal 22 is activated, the voltage level applied at terminals 20/24 is no longer interpreted as a speed setpoint, but as a current setpoint.

The FM 357-2 is therefore capable of presetting a variable clamping torque.

Setpoints must be injected via terminals 24/20.

Hardware connection

Figure 9-46 shows the hardware connections between the FM357-2, signal module (SM) and SIMODRIVE 611-A (FDD).

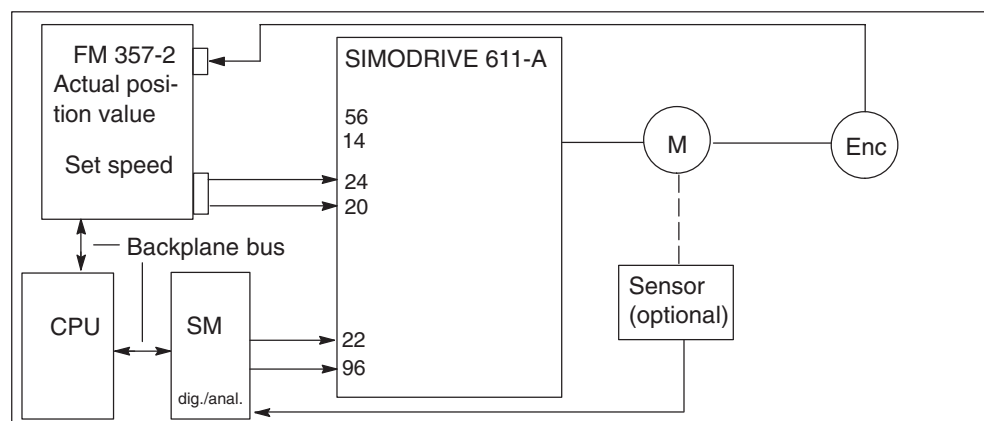


Fig. 9-46 Hardware connections between FM 357-2, signal module and SIMODRIVE 611-A (FDD)

9.18.3 Functional sequence

Selection

The function is activated by instruction FXS[axis]=1. The FM 357-2 sends interface signal "Travel to fixed stop active" (user DB, "AXy", DBX22.4+m) to the CPU.

The CPU must then activate the current limitation on the drive (with analog drives, terminal 96) and send acknowledgement "Enable travel to fixed stop" (user DB, "AXy", DBX3.1+m) to the FM module.

The FM 357-2 now activates the function, the torque limit is set internally to the parameterized value, the acceleration rate reduced accordingly and the axis traversed towards the target position.

Fixed stop is reached

As soon as the axis reaches the fixed stop, the axial following error increases. The control registers that the fixed stop has been reached if the setting in parameter "Following error for fixed stop detection" is exceeded or interface signal "Sensor fixed stop" (user DB, "AXy", DBX41.2+m) is set.

In the case of analog drives, the position controller then outputs a speed setpoint corresponding to the value entered in parameter "Torque limit for fixed stop approach". By virtue of this continuously applied setpoint, the speed controller, whose output is limited by terminal 96, forces the drive to remain at the current limit.

The FM 357-2 then deletes the remaining distance to go and adjusts the position setpoint. The controller enable remains active.

The FM then sends interface signal "Fixed stop reached" (user DB, "AXy", DBX22.5+m) to the CPU.

If with analog drives the FM is to preset a **programmable clamping torque**, then the CPU must switch the drive from speed-controlled to current-controlled operation. To do so, it activates terminal 22 and deactivates the current limitation (terminal 96) after a period of >10 ms. As a result, the current limit is now active in the drive.

The CPU now outputs interface signal "Acknowledge fixed stop reached" (user DB, "AXy", DBX1.1+m). The FM 357-2 reacts to the acknowledgement and outputs the desired clamping torque as a step change to the drive.

A block change is then executed. The clamping torque remains active.

Fixed stop is not reached

If the axis reaches the programmed end position without the “Fixed stop reached” status being detected, the internal torque limitation is cancelled and interface signal “Travel to fixed stop active” (user DB, “AXy”, DBX22.4+m) is reset.

The CPU must then deactivate the current limitation (with analog drives, terminal 96).

It then acknowledges by way of interface signal “Enable travel to fixed stop” (user DB, “AXy”, DBX3.1+m). The block is terminated in the FM 357-2 and NC block processing continues provided that error message “Axis has not reached fixed stop” has not been parameterized.

Deselection

The function is deselected with FXS[...]=0. The FM 357-2 presets a “0” speed or current setpoint, i.e. cancels the clamping torque.

It then resets the following interface signals:

- “Travel to fixed stop active” (user DB, “AXy”, DBX22.4+m)
- “Fixed stop reached” (user DB, “AXy”, DBX22.5+m)

If current-controlled operation is activated in the case of analog drives, the CPU must first activate the current limitation (terminal 96) and switch the drive over into speed-controlled operation (terminal 22) (the FM is applying a speed setpoint of “0”).

The current limitation must then be deactivated (terminal 96).

The CPU then acknowledges by resetting interface signals:

- “Travel to fixed stop enabled” (user DB, “AXy”, DBX3.1+m)
- “Acknowledge fixed stop reached” (user DB, “AXy”, DBX1.1+m)

The FM then takes over the axis in position control (follow-up is ended) and synchronizes it with the new actual position. The travel motion programmed in the block is executed. Logically, this motion must lead away from the fixed stop.

A block change is executed when the target position is reached.

Clock pulse diagrams

The following diagram shows the sequence of events for selection block with FXS[...]=1 and "Fixed stop reached".

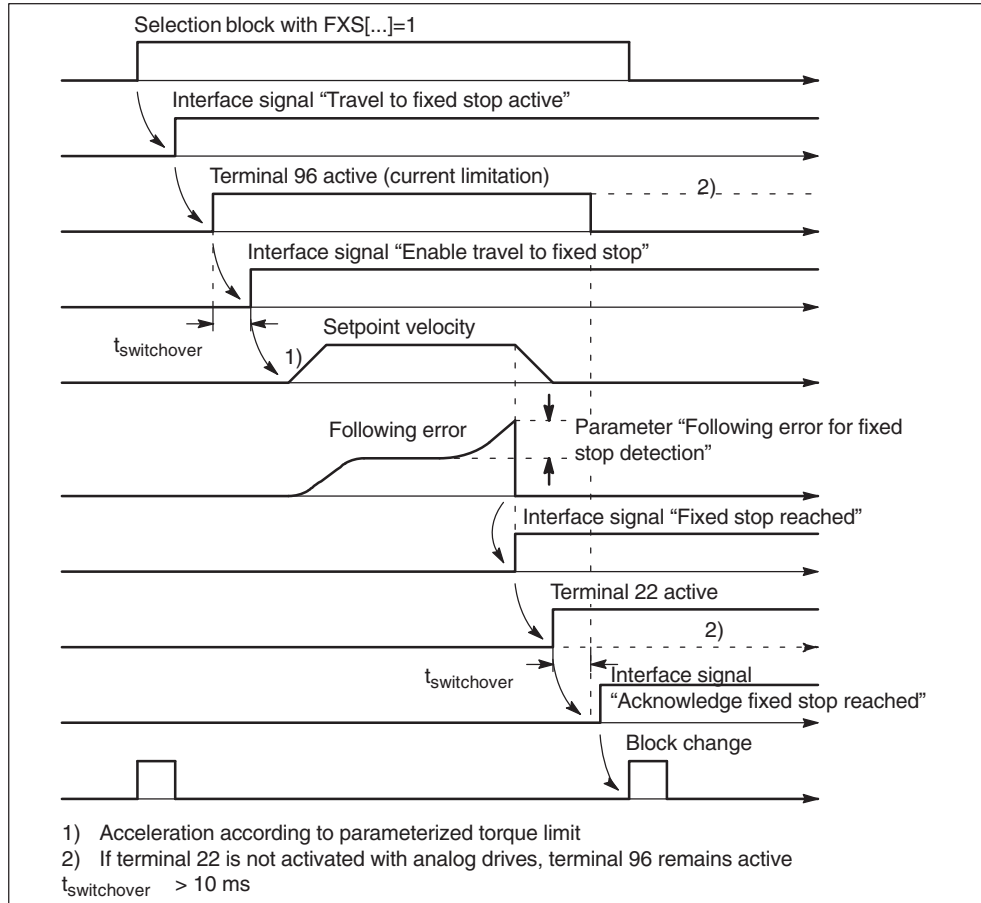


Fig. 9-47 Diagram for "Fixed stop reached"

The following diagram shows the sequence of events for selection block with $\text{FXS}[\dots]=1$ and “Fixed stop not reached”.

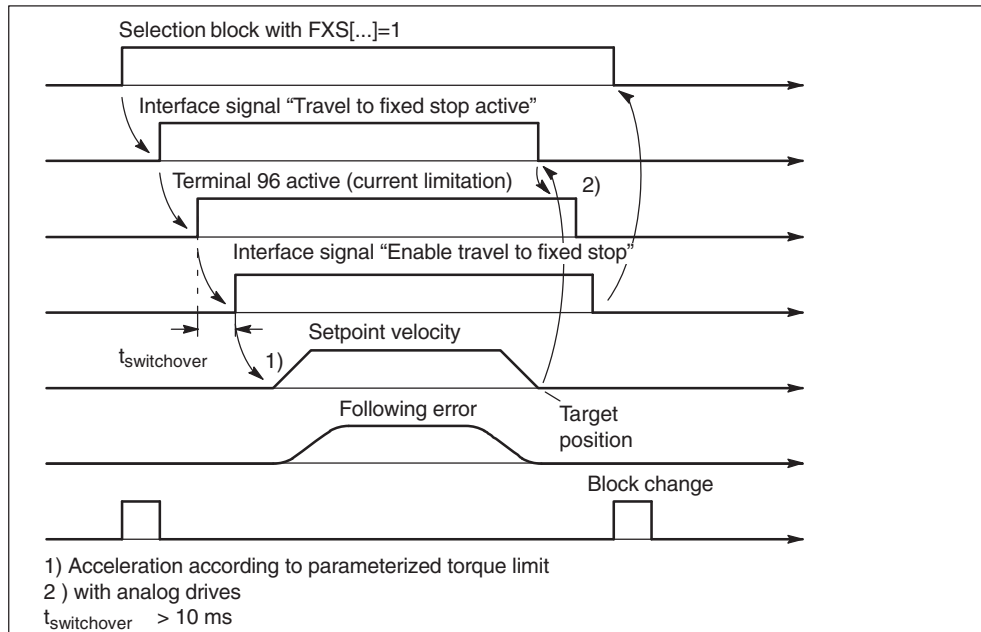


Fig. 9-48 Diagram showing “Fixed stop not reached”

The following diagrams shows function deselection with $\text{FXS}[\dots]=0$.

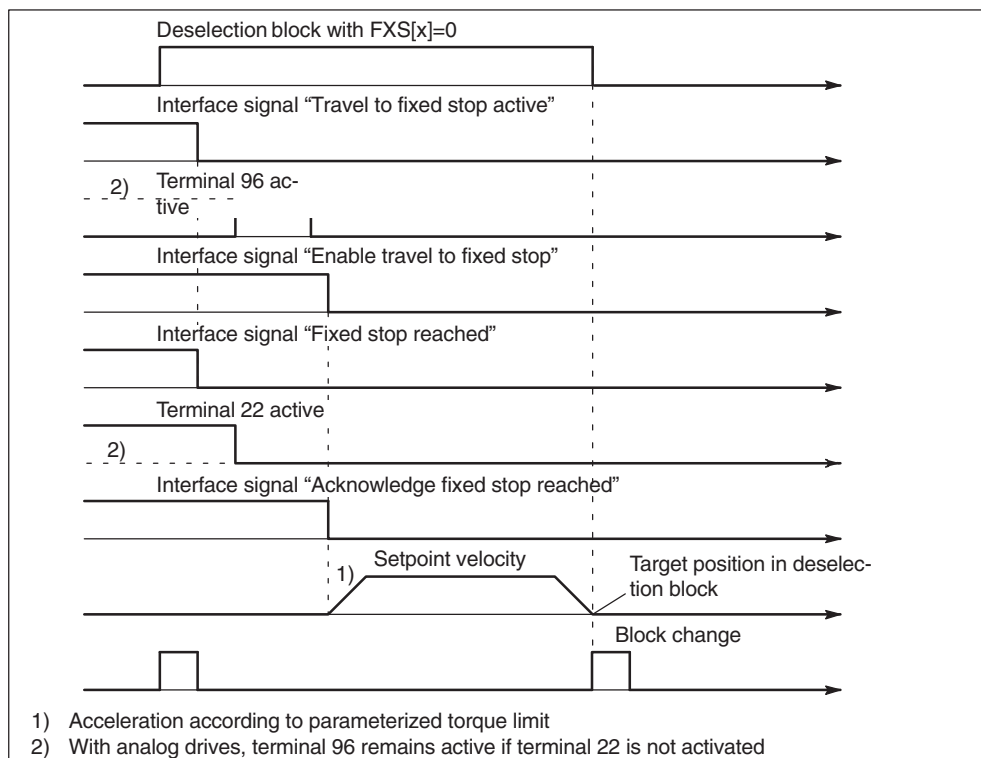


Fig. 9-49 Diagram showing deselection “Fixed stop reached”

9.18.4 Further information

Function abort

The function is deselected in response to a function abort command or if the fixed stop cannot be reached. The following response can be controlled by setting parameter "Error message":

- with error message: Program abort and error message
- without error message: Block change and program continuation (if possible)

The abort operation is executed such that a "nearly reached" fixed stop (setpoint is already the other side of the fixed stop, but still within the following error for fixed stop detection) does not cause any damage (due to momentary follow-up).

As soon as the fixed stop is reached, the function remains active, even after a Reset.

An EMERGENCY STOP command cancels "Travel to fixed stop" in the drive.



Warning

Care must be taken to ensure that no risk situation on the machine can develop after cancellation of "Travel to fixed stop" due to an EMERGENCY STOP command.

Miscellaneous

System variable \$AA_IM[...] can be used to read the actual position of the machine axis, e.g. for measurement purposes, after the fixed stop has been reached.

If a travel command for an axis is set (e.g. from the NC program or user program) after a fixed stop has been reached, then error message "Axis travel to fixed stop still active" is output. The axis remains stationary.

Interface signal "Controller enable" (user DB, "AXy", DBX2.1+m) remains inoperative until the function is deselected.

Error message "Monitoring window travel to fixed stop" is output if an axis is moved out of position by more than the programmed or parameterized value for monitoring window after the fixed stop has been reached. The "Travel to fixed stop" function is deselected for this axis and system variable \$AA_FXS[...]=2 set.

Notice

The monitoring window must be programmed such that the window will respond if the fixed stop is shifted (yields) illegally.

The following error is not monitored while "Travel to fixed stop" is active.

9.19 EMERGENCY STOP

General

If a dangerous situation develops, all axis motions can be braked as quickly as possible by the EMERGENCY STOP sequence. After an EMERGENCY STOP, the module is **not** in the Reset state, it may be possible to continue to the program after any damage has been cleared.

The machine manufacturer is responsible for ensuring that the machine reaches a safe status after shutdown of the axes in cases where personnel will need to enter the motion space of the axes.

The operation must be initiated by a special EMERGENCY STOP signal. According to the applicable safety regulations, it is not permissible to use the EMERGENCY STOP button.

The function can be used only in conjunction with analog drives.

Parameters

The following parameters are relevant for the EMERGENCY STOP function:

Parameters	Value/Meaning	Unit
EMERGENCY STOP braking time (with reference to the maximum velocity)	0.05 (default value) 0.02 to 1 000	[s]
Cutout delay Controller enable EMERGENCY STOP	0.1 (default value) 0.02 to 1 000	[s]

EMERGENCY STOP sequence

The EMERGENCY STOP state must be sent to the CPU as an input signal (from user).

This sends the following signals to the FM 357-2:

Stop = 1 (user DB "FMx", DBX108.3+n)

Controller enable = 0 (user DB "AXy", DBX2.1+m)

Follow-up mode = 1 (user DB "AXy", DBX1.4+m)

The axis signals must be set for all axes which need to be braked.

The NC program is stopped with Stop in the FM 357-2 and the position control loops separated with controller enable = 0.

The axes are then shut down under speed control along the ramp defined in parameter "Braking time EMERGENCY STOP". A preset path motion can be exited during braking.

After the time setting in parameter "Cutout delay controller enable EMERGENCY STOP" has run out, the FM resets the controller enable to the drive.

The setting in parameter "Braking time EMERGENCY STOP" must be adapted to

the mechanical load capability of the installation. Parameter “Cutout delay controller enable EMERGENCY STOP” must be set to a higher value than the braking time. A setpoint of 0 V is output as the controller enable to the drive is cancelled.

When the EMERGENCY STOP state no longer exists, the following signals must be set by the CPU:

- Stop = 0 (user DB “FMx”, DBX108.3+n)
- Controller enable = 1 (user DB “AXy”, DBX2.1+m)
- Follow-up mode = 0 (user DB “AXy”, DBX1.4+m)

The axes are switched back to position control. As they were shut down in follow-up mode, they are still referenced.

The interrupted NC program can be continued with Start in “Automatic” mode. After Start, the axes first approach the interruption point. The NC program then continues from this position.

Figure 9-50 shows a possible sequence of operations.

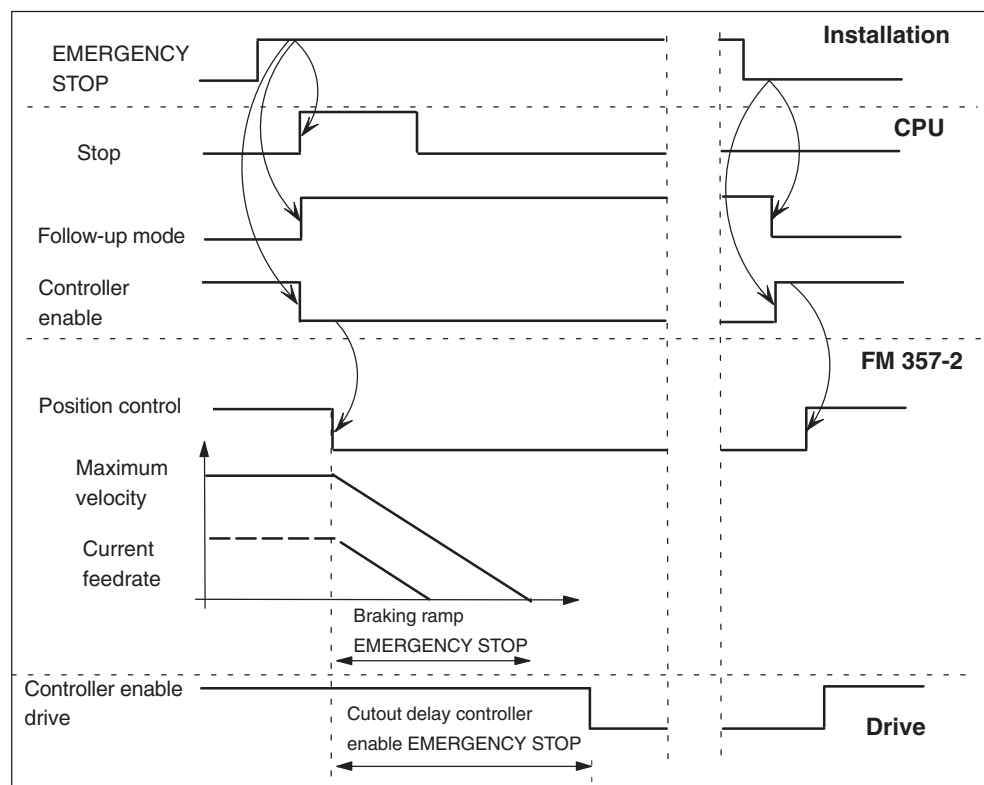


Fig. 9-50 EMERGENCY STOP sequence

Error messages

When the controller enable signal is cancelled while axes are moving, error message “Controller enable reset during motion” is output.

The error should be deleted with CANCEL before the interrupted NC program can be continued.

9.20 Controlling

General

The status “Controlling” designates the traversing of an axis in speed-controlled mode of the drive.

This provides the possibility, e.g. of traversing axes which can hardly or not be controlled in the terms of closed-loop control technology by specifying a velocity value.

The difference of setpoint and actual position is monitored. If the parameter “Setpoint/actual value tolerance” is exceeded, the axis is stopped and an error is output. Any limitations (SW limit switches), as well as contours or limit positions can be monitored or approached only within this tolerance.

This function can only be used for analog or PROFIBUS drives.

Activation

For the appropriate axis, the parameter “Controlling” and the interface signal “Control axis” (user DB AXy, DBX8.1+m) must be set.

At standstill, the axis must be in position control, i.e. servo enable must be set.

Based on this initial status, the FM will open the closed-loop position control loop and will traverse the axis controlled.

With the end of the motion, the servo control is closed again, and the setpoint position set to the actual position really reached in the interpolator.

The function is not permitted under the following conditions:

- Reference point approach of the axis
- Master or slave axis in Gantry compound
- Master or slave axis of an axis coupling (coupled motion, master value coupling, tangential control)
- Handling transformation
- Travel to fixed stop

Parameters	Value/Meaning	Unit
Controlling	No (default value) Axis traverses in position control Yes Axis can traverse controlled (speed-controlled)	–

Motion profile

The interpolator does not differ between speed-controlled and position-controlled mode. The velocity specification is provided independently of the axis parameterization and the programmed motion.

Relevant axis parameters are:

- Acceleration
- Load gearbox ratio
- Distance traversed per spindle revolution
- Velocity assignment: max. motor speed/max. setpoint voltage

The FM will calculate a manipulated variable corresponding to the velocity from the parameterized motor and drive adaptation (with analog drives, e.g. a voltage).

Example

A linear axis is to traverse in a synchronized action in controlled axis mode by means of a POS instruction:

Axis parameterization:

- Controlled axis mode: yes
- Acceleration: $a_{\max} = 1 \text{ m/s}^2$
- Load gearbox: LG = 1:1
- Distance traversed per spindle revolution:
 $S_{\text{sp}} = 10 \text{ mm/rev.}$
- Velocity assignment:
Max. motor speed: $n_{\max} = 1,000 \text{ r.p.m.}$
Max. setpoint voltage: $U_{\max} = 8 \text{ V}$

Synchronized action:

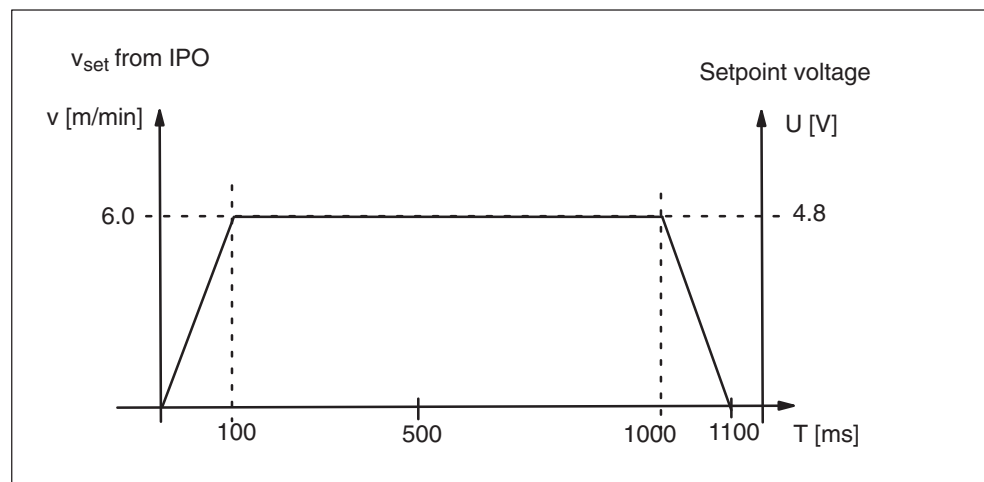
- WHEN TRUE DO POS[X]=100 FA[X]=6000
Total travel: $S_G = 100 \text{ mm}$
Setpoint velocity: $v_{\text{soll}} = 6,000 \text{ mm/min}$

Setpoint motion profile specified from IPO:

- Acceleration and braking ramp
Time: $T_a = v_{\text{setl}} / a_{\max} = 6 \text{ [m/min]} / 1 \text{ [m/s}^2] = 0.1 \text{ s}$
Distance traversed: $S_a = v_{\text{setl}}^2 / 2 * a_{\max} = 6 \text{ [m/min]}^2 / 2 \text{ [m/s}^2] = 5 \text{ mm}$
- Range of constant velocity
Distance traversed: $S_k = S_G - S_a - S_a = 100 \text{ mm} - 5 \text{ mm} - 5 \text{ mm} = 90 \text{ mm}$
Time: $T_k = S_k / v_{\text{set}} = 90 \text{ [mm]} / 6 \text{ [m/min]} = 0.9 \text{ s}$

Velocity-voltage assignment:

- $U \text{ [V]} = v_{\text{set}} * (U_{\max} / n_{\max}) * (LG / S_{\text{sp}}) = v_{\text{set}} * 8/10 \text{ [min/m} * \text{V]} = 4.8 \text{ V}$



Actual value sensing

Generally, an encoder actual value must be provided for the axis since:

- the position-control loop is closed at standstill to prevent drifting of the axis;
- the position setpoint position in the interpolator is set to the really reached position with the end of the motion to prevent an error accumulation.

Sample program 1

Traversing program with 2 axes whereby the X axis is to be traversed controlled and the Y axis position-controlled from synchronized actions.

...

CYCLE:

; Approach initial position

N20 WHEN TRUE DO POS[X]=0 FA[X]=2000

N30 WHENEVER \$AA_IM[X] > 0 DO RDISABLE ; Stop block sequence

N40 G0 Y0

; Start X axis with a Y axis position greater than 150 mm

N50 WHEN \$AA_IM[Y] >150 DO POS[X]=250

N60 G1 Y300 F1000

; Endless motion of the X axis in the range between 300 and 600 mm

N70 ID=1 EVERY \$VA_IM[X] < 300 DO MOV[X]= 1 FA[X]=6000

N80 ID=3 EVERY \$VA_IM[X] > 600 DO MOV[X]= -1 FA[X]=4000

; Traverse Y axis

N110 G1 Y20 F1000

; Stop X axis at 400 mm

N120 WHEN \$VA_IM[X] < 400 DO MOV[X]= 0

N130 WHENEVER \$AA_IM[X]>400 DO RDISABLE ; Stop block sequence

N130 G4 F0.1 ; Dummy block for synchronized action

GOTOB CYCLE

M30

Sample program 2

Both X and Y axes are to be traversed controlled; the program "CYCLE" is started from a synchronized action and is executed

```
...
N10 CLEAR (1) ; Clear marker 1
N10 WHEN TRUE DO ZYKLUS ; Start program "CYCLE"
N20 G4 F0.1

N30 WAITM(1,1) ; Wait for end of program "CYCLE"
N40 M30

; Program "CYCLE" ; Call from synchronized action !

N10 POS[X]=0 POS[Y]=0 FA[X]=1000 FA[Y]=2000
N20 $A_OUT[2]=TRUE
N30 POS[X]=200 FA[X]=3000
N40 POS[Y]=400
N50 POS[Y]=100
N60 POS[X]=0
N70 $A_OUT[2]=0 SETM (1) ; Synchronism marker 1 for the NC program
N80 M17
```

Sample program 3

The axis is to be traversed controlled whereby the signal "Control axis"
(user DB "AXy", DBX8.1+m) is influenced via an M command.

```

N05 M22                ; Activate the signal "Control axis"

N10 G0 G60 X0 Y0 Z0    ; G0: X axis traverses controlled
                        ; Block preparation synchronizes to reached actual position of
                        ; X axis
N20 G1 X100 F100        ;G1: X axis traverses controlled
                        ; Block preparation synchronizes to reached actual position of
                        ; X axis

N30 EVERY TRUE DO POS[X] = 60 FA[X]=500 ; X axis traverses controlled
                                                ; from synchronized action

N40 G4 F0.5
                        ; No synchronization

N50 EVERY TRUE DO POS[X] = 20 FA[X]=500
N60 G4 F0.5
                        ; Block preparation synchronized to reached
                        ; actual position of X axis

N70 POS[X]=100 G1 Y30 Z50 F1000 ;X axis traverses controlled as POS
                                ; axis
                                ;Block preparation synchronized
                                ; to reached actual position of X axis

N80 G64 G1 X50          ; G64 block sequence, X axis traverses controlled
N90 X70                 ; Block change without switching to position control
N100 X80
N110 X90 M23            ; Deactivate "Control axis" during motion
                        ; Switching to position control at standstill of axis
                        ; Block preparation synchronized to reached
                        ; actual position of X axis

N120 G60 G1 X100 F1     ; G1: X axis traverses position-controlled

N130 M2
    
```


9.21 Axis replacement

General

Using the function “Axis replacement”, it is possible to enable an axis in one channel and to assign it to a different channel. The axis must be assigned several channels including a master channel. The master channel defines the default assignment for the axis after switching on the module.

The axis replacement function can be controlled either by the NC or by the user program.

During axis replacement, a synchronization to the current position must be carried out. For the resulting particularities when executing the program, see Section 10.36.

This function is available as per from software version 5.

Axis state

As a result of the axis replacement function, the axis may have the following states:

- **Axis in channel**
The kind of axis is assigned to the current channel, i.e. it can be programmed here and be traversed.
- **CPU axis**
This kind of axis is assigned to the CPU and can only be positioned by the CPU.
- **Neutral axis**
This kind of axis is in a neutral state. By axis replacement, a transition either to a CPU axis or to a channel axis is possible.
- **Axis not in channel**
This kind of axis is not assigned to the current channel. The axis can be assigned to a neutral axis, to a CPU axis or to another channel.

Programming

Axis replacement can be programmed in the NC program using the following NC statements:

RELEASE (axis, axis, ...)	; Bring axis to the neutral condition
GET (axis, axis, ...)	; Accept axis from the neutral condition into the ; current channel
GETD (axis, axis, ...)	; Accept the axis directly from a channel into the ; current channel

Implied axis replacement

An implied axis replacement is initiated if:

- a neutral axis is programmed in a channel
Status: neutral axis → channel axis
- a CPU axis is not positioned and enabled by the CPU
Status: CPU axis → neutral axis
- a channel axis does not traverse and is to be positioned by the CPU
Status: channel axis → CPU axis
- a neutral axis is to be positioned by the CPU
Status: neutral axis → CPU axis
- reset is carried out in the status “neutral axis”
Status: neutral axis → channel axis

Axis replacement carried out by the user program

The following interfaces are provided:

- “Request CPU axis“ (user DB “AXy”, DBX50.7+m)
- “CPU axis“ (user DB “AXy”, DBX66.7+m)
- “Neutral axis” (user DB “AXy”, DBX25.6+m)
- “Axis replacement possible” (user DB “AXy”, DBX25.5+m)
- “Axis is in channels 1...4” (user DB “AXy”, DBX25.3...0+m)

Example: Replacement using NC statements

The axis is assigned to the first and second channels whereby the first channel is the master channel. Axis replacement is possible via RELEASE in channel 1 and GET in channel 2.

	User DB “AXy”	
	DBB25+m	DBB66+m
After Power On	0010 0001	0000 0000
Channel 1: N10 RELEASE(AX)	0110 0001 ↓ 0100 0001 ↓	0000 0000 ↓ 0000 0000 ↓
Channel 2: N10 GET(AX)	0010 0010 ↓ 0000 0010 ↓	0000 0000 ↓ 0000 0000 ↓

Example: Axis replacement carried out by the user program

The axis is assigned to the first and to the second channels whereby the first channel is the master channel. A channel axis is turned to a CPU axis and then to a neutral axis.

	User DB "AXy"		
	DBB50	DBB25	DBB66
After Power On	0000 0000	0010 0001	0000 0000
Request CPU axis (set by the user)	1000 0000	0010 0001	0000 0000
Axis is CPU axis	1000 0000	0010 0001	1000 0000
...			
Request CPU axis (reset by the user)	0000 0000 ↓ 0000 0000 ↓	0010 0001 ↓ 0010 0001 ↓	1000 0000 ↓ 0000 0000 ↓
Axis is neutral axis	0000 0000	0110 0001	0000 0000



NC Programming

10

Chapter overview

Section	Title	Page
10.1	Basic principles of NC programming	10-3
10.2	Coordinate systems and dimensions	10-10
10.3	Zero offsets (frames)	10-22
10.4	Set actual value (PRESETON)	10-30
10.5	Programming axis movements	10-31
10.6	Path response	10-64
10.7	Dwell time (G4)	10-73
10.8	Coupled motion (TRAILON, TRAILOF)	10-74
10.9	Tangential control (TANG, TANGON, TANGOFF)	10-76
10.10	Measurement (MEAS, MEAW)	10-78
10.11	Travel to fixed stop (FXST, FXSW, FXS)	10-84
10.12	Stop preprocessor (STOPRE)	10-86
10.13	Working area limitation (G25, G26, WALIMON, WALIMOF)	10-86
10.14	M functions	10-88
10.15	H functions	10-90
10.16	Tool offset values (T functions)	10-91
10.17	Protection zones (NPROTDEF, EXECUTE, NPROT)	10-93
10.18	Fundamentals of variable NC programming	10-97
10.19	R parameters (arithmetic parameters)	10-103
10.20	System variables: \$P_, \$A_, \$AC_, \$AA_	10-105
10.21	FIFO variable (\$AC_FIFO1[...] to \$AC_FIFO10[...])	10-113
10.22	User variables	10-118
10.23	Program jumps (GOTOF, GOTOB, GOTO, GOTOC LABEL, IF)	10-124
10.24	Control structures	10-126
10.25	Axis variable	10-131
10.26	String operations	10-132
10.27	Reading, writing and deleting a file	10-134
10.28	Program coordination (INIT, START, WAITE, WAITM, WAITMC, SETM, CLEARM)	10-139

Section	Title	Page
10.29	Subroutine technology (L, P, RET)	10-143
10.30	Asynchronous subroutines (ASUB)	10-149
10.31	Activating machine data (NEWCONF)	10-153
10.32	Synchronized actions	10-154
10.33	Oscillation	10-176
10.34	Master value coupling	10-181
10.35	Electronic gear	10-187
10.36	Axis replacement	10-192
10.37	Speed feedforward control (FFWON, FFWOF)	10-195
10.38	Overview of statements	10-196

Overview

You can program the statements needed to move axes and control the machine in an NC program.

You can create an NC program with the editor of the “Parameterize FM 357-2” tool (see Section 5.5.5).

Notice

All units in this document are given in the **metric** basic system.

10.1 Basic principles of NC programming

Guideline

NC programs must be structured according to the guidelines given in DIN 66025.

Program memory

There is a minimum of 128 KB NC program memory available on the FM 357-2.

You can display the percentage of free system resources by selecting menu commands **PLC > FM Properties...** in the “Parameterize FM 357-2” tool.

10.1.1 Program structure and program name

Structure and contents

The program consists of a sequence of blocks in which the necessary statements are written. The last block in the program contains the end of program character.

Block	Statements				Comments
1	N10	G0	X20	–	; with rapid traverse to position X20
2	N20	G1	X100	F100	; with a feed of 100 mm/min to X100
3	N30	G91	Y10	–	; move Y axis 10 mm in plus direction
4	N40	–	–	–	
5	N50	M2			; end of program (last block)

Program name

The name of the program is derived from the file name. You can choose any name that conforms to the following rules:

- The first two characters must be letters
- The name can have a maximum of 25 characters; the first 24 characters are displayed
- You cannot use the space or tab character
- Use only characters that are part of the character set of the control system (see Section 10.1.4).

Example: MPF100

10.1.2 Statements

General

For an overview of all available programming statements, please refer to Section 10.38.

Statements with address and numeric value

There are both permanently and variably assigned address letters. The permanently assigned address letters have a defined meaning and cannot be changed. The variably assigned addresses can be changed by modifying their parameters.

Example:

Permanent addresses: L, P, G, F, T, M, ...

Variable addresses: X, Y, Z, A, ...

The numerical value consists of a sequence of digits. With certain addresses, the digits may include a decimal point and a leading sign. A positive sign (+) can be omitted.

	Address/value		
Example:	G1	X-20.1	F300
Notes:	Linear interpolation with feed	Path or position for X axis: -20.1	Feed: 300 mm/min

Fig. 10-1 Structure of statements with an address and numerical value

Leading zeros in statements can be omitted (e.g. G1 or G01).

Exception: See Section 10.29, Subroutine system

Multiple address characters

A statement can also contain multiple address letters. In this case, however, the numerical value must be assigned by a “=” symbol.

Example:

CR=5.33 ; Circle radius for a circle with radius and end point

G functions

The G functions specify how a position is to be approached. They are used to activate and deactivate functions.

Example:

G0 ... ; Linear interpolation with rapid traverse

G1 ... ; Linear interpolation with feed

The G functions are subdivided into groups according to their meaning. Each G group has an initial setting, i.e. one of the G functions in the group is always active immediately when the program starts.

Only one G function of a G group may be active at a time.

M functions

M functions are used to control machine functions which the user defines. Some of the M functions have fixed functionality (e.g. M2 for end of program)

R parameters

R parameters R0...R99 (REAL type) are available for optional use by the user, e.g. as arithmetic parameters.

System variable

The programmer can use system variables to read current values from the control system and to write some values as well. The system variables begin with the "\$" sign and are written in upper case letters.

Example:

R34=\$AA_IW[X] ; Read actual position of X axis and save in R34

Supplementary statements

There are statements which supplement the programming of functions.

These include statements for:

- Operations and mathematical functions
- Offsets and working area limitations
- Messages, jump statements, ...

10.1.3 Block structure

Block contents

A block should contain all the data required to execute a machining step. A block consists generally of several statements and the character “L_F” for “end of block” (line feed). The “L_F” character is generated automatically on a line break. If a block number is used, this must always appear at the start of the block.

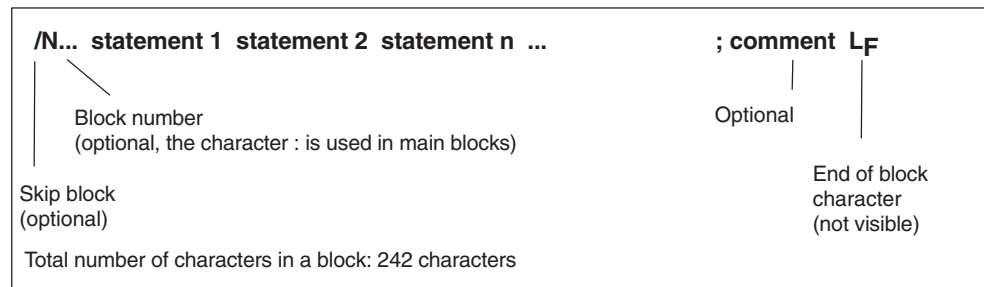


Fig. 10-2 Block structure

Sequence of statements

To obtain a clearly defined block format, the statements in the block should be written in the following sequence with a separator (blank or tab) between them.

Example:

N9235 G... X... Y... Z... F... T... M... L_F

N	–	Address of block number
9235	–	Block number
G...	–	Preparatory function
X... Y... Z...	–	Positional data
F...	–	Feed
T...	–	Tool
M...	–	Miscellaneous function
L _F	–	End of block

Some addresses can be used several times in the block (e.g. G..., M...).

Program section

A program section consists of a main block and several subblocks. In the main block you should specify all the statements which are required to start the machining sequence in the program section that begins there. Subblocks are identified by the character “ N ” and a positive block number (integer) at the start of the block.

Example:

```

...
:10 F200           ; Main block, identified by “:” and block number
N20 G1 X14 Y35    ; 1st subblock, identified by “N” and block number
N30 X20 Y40       ; 2nd subblock, identified by “N” and block number
N40 Y-10          ; 3rd subblock, identified by “N” and block number
...

```

Skip block

Blocks in a program that must not be executed in every program run can be marked by an oblique “/” in front of the block number. The statements in blocks which are identified in this way are not executed when the function “skip block” is activated.

The function is activated and deactivated via the interface signal “Skip block” (user DB “FMx”, DBX105.0+n).

Example:

```

N10 ...           ; is executed
/N20 ...          ; is skipped if “skip block” is active
N30 ...           ; is executed
/:40 ...          ; is skipped if “skip block” is active
N50 ...           ; is executed
...

```

Comments

Comments are added to explain the program and individual blocks. A comment appears at the end of the block and is separated from the words of the block by a semicolon “ ; ”.

Comments are stored and displayed together with the contents of the remaining block in the current block display while the program is running.

Example:

```

N1                ; G&S Co., Order No. 1271
N2                ; Program written by John Smith
N5 G1 F100 X10 Y20 ; Comment

```

Output messages

Messages can be programmed to appear on the screen while the program is running. You can program a message by enclosing the message in parentheses “()” after “MSG”. The message is displayed until it is deactivated, a new message is programmed or the program is terminated.

Example:

```
N1 MSG (“Travel to Position 1”)           ; Activate message
N2 G1 X... Y...
N3 ...
N40 MSG ( )                               ; Deactivate message from N1
```

Note:

Messages are not displayed by the “FM 357-2 Parameterization” tool.

The variable can be read from the user program via FB 2 and be saved in a data block. This saved variable can be displayed in the OP.

10.1.4 Character set of control

General

The following set of characters is provided for the writing of NC programs:

Letters

A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z

No distinction is made between upper and lower case letters; they are interpreted as equal.

Digits

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Printable special characters

%	–	Start of program character
(–	Left parenthesis
)	–	Right parenthesis
[–	Left bracket
]	–	Right bracket
<	–	Less than
>	–	Greater than
:	–	Main block, label terminator
=	–	Assignment, part of equality
/	–	Division, skip block
*	–	Multiplication
+	–	Addition
–	–	Subtraction
“	–	Quotation mark / identifier for character string
'	–	Inverted comma / identifier for special numerical values: hexadecimal, binary, ...
\$	–	Identifier for system variable
_	–	Underscore (belongs to letters)
?	–	Reserved
!	–	Reserved
.	–	Decimal point
,	–	Comma, separator
;	–	Start of comment
&	–	(Formatting character), same effect as space

Non-printable special characters

L _F	–	End of block
Tabulator	–	Separator
Space	–	Separator (space)

10.2 Coordinate systems and dimensions

Overview

In this section, you can find information about:

- Coordinate systems
- Axis types
- Absolute dimensions and incremental dimensions (G90, G91, AC, IC)
- Absolute dimensions for rotary axes (DC, ACP, ACN)
- Dimensioning inches and metric (G70, G71)
- Plane selection (G17, G18, G19)

10.2.1 Coordinate systems

General

Right-handed, rectangular coordinate systems according to DIN 66217 are used.

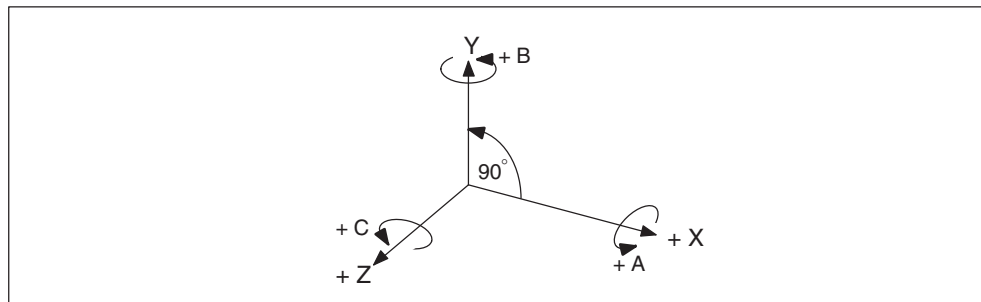


Fig. 10-3 Definition of axis directions

X, Y, Z – Perpendicular axes

A, B, C – Rotary axes rotating around X, Y, Z

Further address letters are available for additional axes.

Machine coordinate system (MCS)

The machine coordinate system comprises all axes that physically exist on the machine. For example, reference points (zero points) are defined in the machine coordinate system.

Workpiece coordinate system (WCS)

The geometry of a workpiece is programmed in the workpiece coordinate system. The workpiece coordinate system is a rectangular Cartesian coordinate system. The reference to the machine coordinate system is established by means of zero offsets.

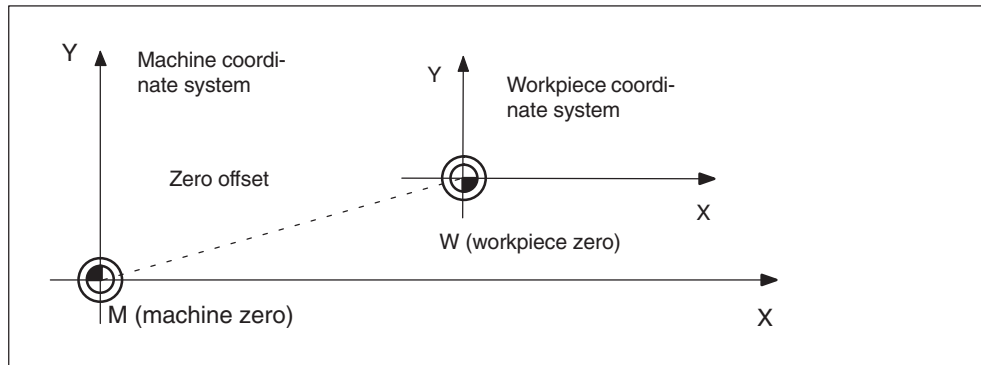


Fig. 10-4 Machine and workpiece coordinate systems

10.2.2 Axis types

General

The FM 357-2 has the following different types of axes:

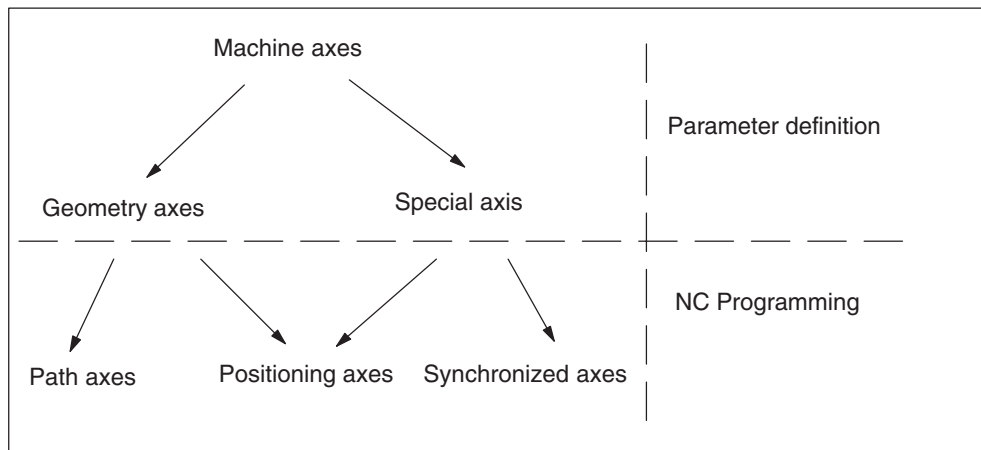


Fig. 10-5 Relationship between axis types

Machine axes

This refers to all axes installed on the machine tool. They are defined either as geometry axes or as special axes. The axis names can be defined in parameters (default: X1, Y1, Z1, A1).

- **Geometry axes**

Geometry axes are used to program the workpiece geometry. The geometry axes describe a rectangular coordinate system.

The tool offsets are only included in calculations involving geometry axes.

You can assign an axis name by means of a parameter (default: X, Y, Z)

- **Path axes**

Path axes describe the contour in space, and are interpolated with a common path feed. Geometry axes are defined as path axes in the standard configuration.

- **Positioning axes**

Positioning axes are traversed, independently of path axes, with their own axis-specific feedrate. All axes can be programmed as positioning axes using the traversing statements POS[...] or POSA[...]. Synchronized axes and geometry axes can be traversed as positioning axes within the scope of a block.

- **Special axes**

Unlike geometry axes, special axes have no geometrical relationship (e.g. rotary axes).

- **Synchronized axes**

Synchronized axes are those which are not included in the path axis grouping. They merely move in synchronism with the path distance from the start position to the programmed end position, i.e. synchronized axes require for their paths the same time as geometry axes for theirs.

The feed programmed with F only applies to the path axes programmed in the block. The feed is calculated internally for synchronized axes.

- **Positioning axes**

as described under geometry axes

10.2.3 Absolute dimensions and incremental dimensions (G90, G91, AC, IC)

General

Statements G90/G91 are set to define whether programmed path data must be interpreted as absolute values (as coordinate point) or as relative values (as distance to be traversed).

This applies to linear and rotary axes.

Value range for path data:

$\pm 0.001...10^8$ mm or $\pm 0.0001...10^8$ inch

Programming

G90 ; Absolute dimension, modal

or

X=AC(...) Y=AC(...) Z=AC(...) ; Absolute dimension, axis-specific,
; non-modal

G91 ; Incremental dimension, modal

or

X=IC(...) Y=IC(...) Z=IC(...) ; Incremental dimension, axis-specific,
; non-modal

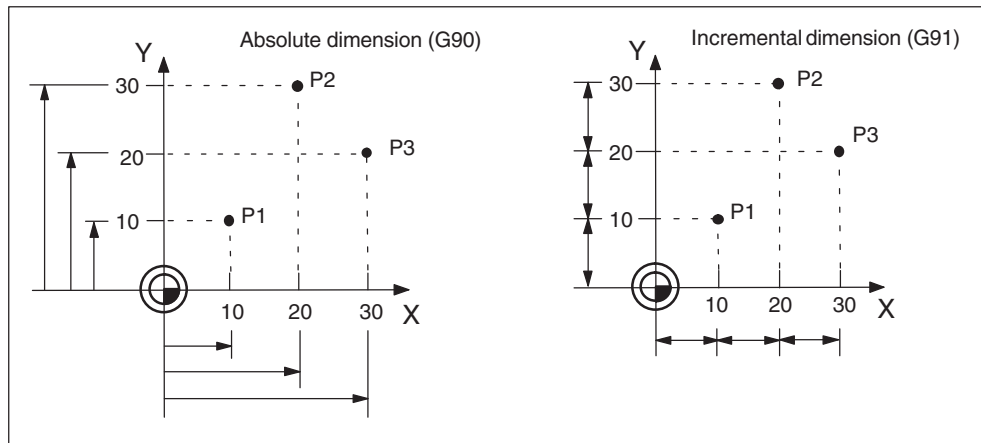


Fig. 10-6 Absolute and incremental dimensioning

Absolute dimensioning G90

The programmed dimension refers to the **zero point of the current workpiece coordinate system**.

G90 is active for all axes in the block, and remains active until canceled by G91.

Example:

```
...
G90          ; Absolute dimension
X10 Y10     ; P1 with reference to zero point
X20 Y30     ; P2 with reference to zero point
X30 Y20     ; P3 with reference to zero point
...
```

Incremental dimensioning G91

Every programmed dimension refers to the last programmed contour point. The sign specifies the travel direction, and the numerical value specifies the **distance to travel**.

G91 is active for all axes in the block, and remains active until canceled by G90.

Example:

```
...
N10 G90     ; Absolute dimension
N20 X10 Y10 ; P1 with reference to zero point
N30 G91     ; Incremental dimension
N40 X10 Y20 ; P2 with reference to P1
N50 X10 Y-10 ; P3 with reference to P2
...
```

G90, G91, AC(...), IC(...)

You can switch between absolute and incremental between blocks. You can also program the dimensions for each axis individually within a block by specifying AC(...) absolute dimensions or IC(...) incremental dimensions.

Example:

```
N1 X=AC(400) ; Axis X travels to position 400 (absolute dimensions)
N2 X=IC(100) ; Axis X travels a distance of 100 in plus direction
              ; (incremental dimensions)
...
N10 G90 X20 Y30 Z=IC(-5) ; X, Y = absolute dimensions,
                        ; Z = incremental dimensions
N11 X70 Y50 Z20         ; X, Y, Z = absolute dimensions
N12 G91 X33 Y22 Z=AC(3.4) ; X, Y = incremental dimensions,
                        ; Z = absolute dimensions
```

10.2.4 Absolute dimensions for rotary axes (DC, ACP, ACN)

General

Special statements for defined approach conditions are provided for rotary axes (traversing range 0...360°).

Programming

Axis=DC(...) ; Approach position directly via shortest possible distance, non-modal

Axis=ACP(...) ; Approach position in positive direction, non-modal

Axis=ACN(...) ; Approach position in negative direction, non-modal

Shortest route DC

The rotary axis approaches the programmed position absolutely via the shortest possible distance. The direction of axis rotation results automatically. The rotary axis travels in a maximum range of 180°.

If the path is the same in both directions, the plus direction takes priority.

Example:

N10 G90 A45 ; Approach position 45°

N20 A=DC(315) ; Axis A approaches position 315° across the shortest path

...

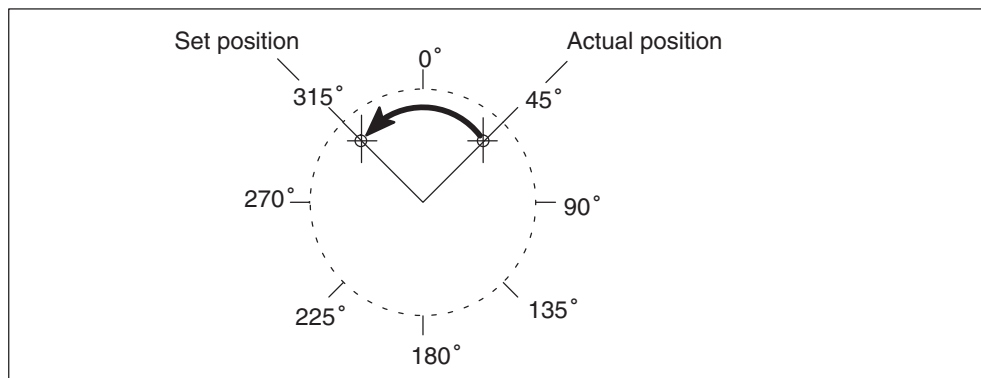


Fig. 10-7 Move rotary axis across shortest path

Positive direction ACP

The rotary axis approaches the programmed position absolutely and in a positive direction of rotation. The function is non-modal and is dependent on G90 or G91.

Example:

N10 G90 A135 ; Approach position 135°

N20 A=ACP(45); Axis A approaches position 45° in a positive direction of rotation

...

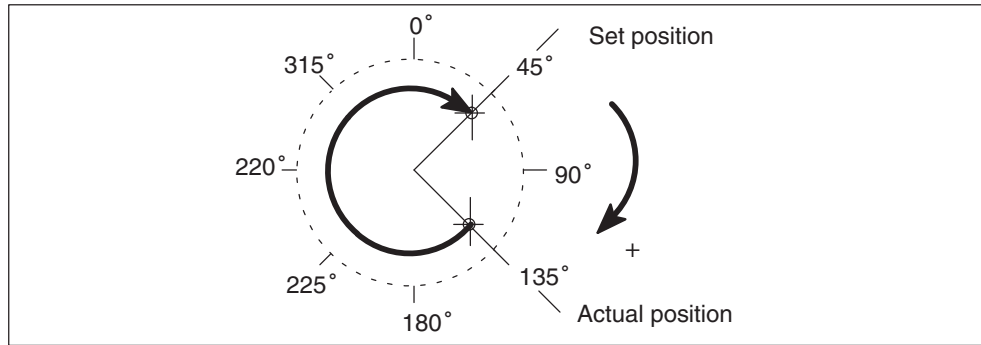


Fig. 10-8 Move rotary axis in positive direction to absolute position

Negative direction ACN

The rotary axis approaches the programmed position absolutely and in a negative direction of rotation. The function is non-modal and is dependent on G90 or G91.

Example:

N10 G90 A315 ; Approach position 315°

N20 A=ACN(45); Axis A approaches position 45° in a negative direction of rotation

...

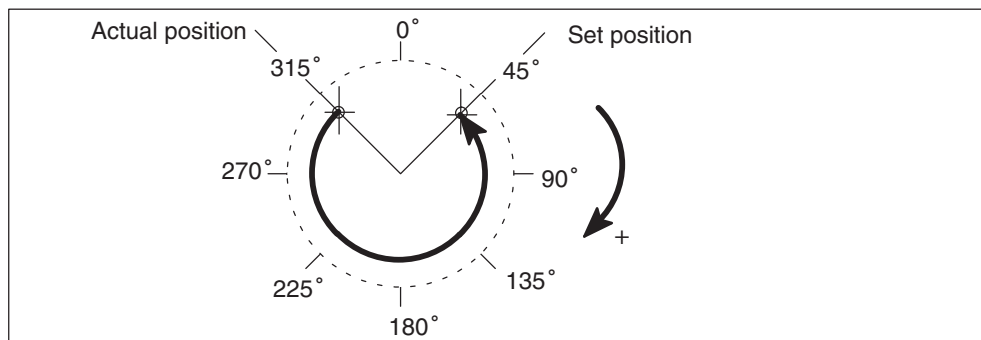


Fig. 10-9 Move rotary axis in negative direction to absolute position

Traversing range greater than 360°

When absolute positioning and a specified direction (ACP, ACN) are used, a rotary axis can be moved in a traversing range between 0° and 360°.

To move a rotary axis in a block by more than 360°, program G91 or IC(...).

10.2.5 Polar coordinates (G110, G111, G112, RP, AP)

General

If dimensioning is based on a central point (pole) with radius and angle data, then it is useful to program dimensions directly as polar coordinates.

Interpolation types G0, G1, G2 and G3 are permitted here.

The polar coordinates refer to the abscissa of the plane selected with G17, G18 or G19. The 3rd geometry axis, which is perpendicular to this plane, can also be specified as a Cartesian coordinate. This allows you to program spatial dimensions as cylinder coordinates.

Cartesian address dimensions cannot be programmed in the current plane in blocks with polar coordinates.

Programming

G110	; Polar dimension with reference to the last programmed position
G111	; Polar dimension with reference to workpiece zero
G112	; Polar dimension with reference to the last valid pole
X... Y... Z...	; Define pole with Cartesian coordinates
RP=	; Polar radius
AP=	; Polar angle

Polar dimension G110, G111, G112

G commands G110 to G112 uniquely define the pole for polar coordinates. They each require a separate block. The pole can be specified in rectangular coordinates or in polar coordinates.

Dimensions with IC(...) or AC(...) e.g. G110 X=AC(50) for rectangular coordinates have no effect, since the G commands G110...G112 already define the unique reference.

If no pole is defined and polar coordinates (polar angle, polar radius) are programmed, the zero point of the current workpiece coordinate system is the valid pole. The same applies if the plane was changed with G17, G18 or G19.

Example 1: G110 X... Y...

N10 G0 X10 Y30 ; Last position
 N11 G110 X20 Y-18 ; Pole
 N12 G1 AP=45 RP=50 F300

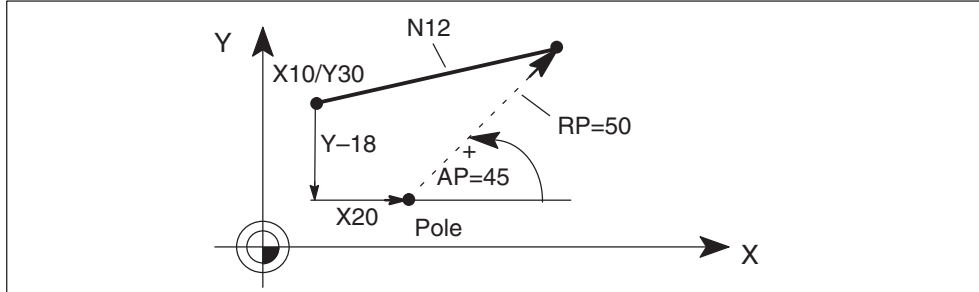


Fig. 10-10 Programming G110

Example 2: G110 AP=... RP=... (in polar coordinates)

N10 G0 X10 Y30 ; Last position
 N11 G110 RP=37 AP=315
 N12 G1 AP=45 RP=50 F300

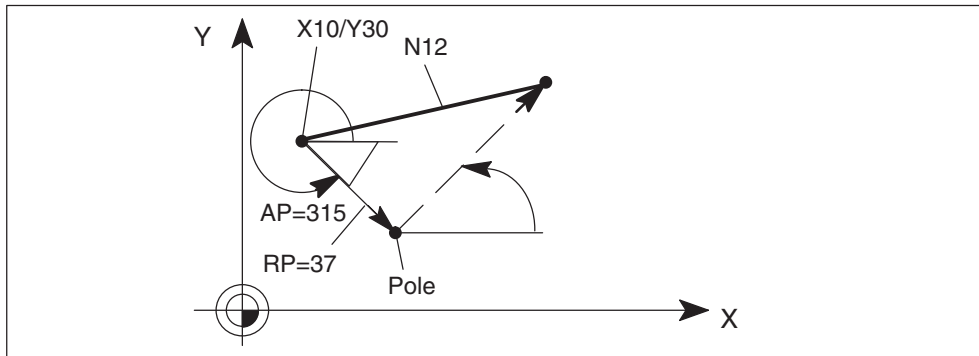


Fig. 10-11 Programming G110 (in polar coordinates)

Example 3: G111 X... Y...

N10 G111 X20 Y18 ; Pole

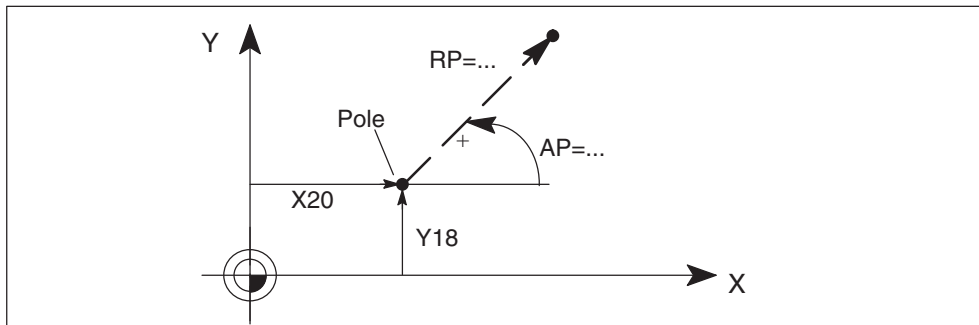


Fig. 10-12 Programming G111

Example 4: **G112 X... Y...**
 N1 G111 X10 Y50 ; Old pole

 N10 G112 X20 Y-18 ; New pole

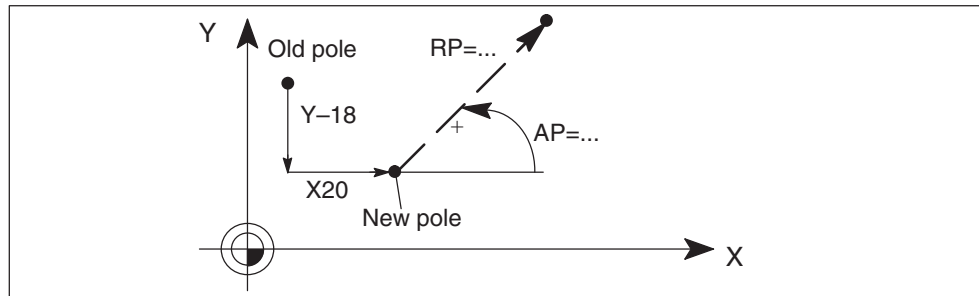


Fig. 10-13 Programming G112

Polar radius RP

The polar radius is written in address **RP=...** in accordance with the valid unit of length (mm or inch), but only positive absolute values are allowed.

The polar radius is modal and must only be rewritten in blocks in which it changes.

Polar angle AP

The polar angle is programmed in address **AP=...** in degrees.

The angle always refers to the horizontal axis (abscissa) of the plane (e.g. for G17: X axis). You can specify both positive and negative angles and incremental dimensions with **AP=IC(...)**. The reference for incremental angle dimensions is the last programmed polar angle. If none exists, the value refers to 0 degrees.

The polar angle is modal and must only be rewritten:

- if a pole change is programmed.
- if a new plane is selected.

Example:

G17: X/Y plane

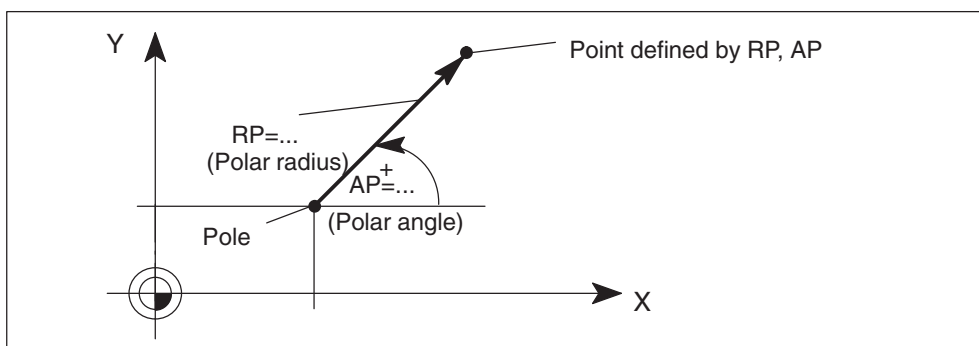


Fig. 10-14 Polar radius and polar angle

10.2.6 Meters and inches (G70, G71)

General

The control is configured for an internal system of measurement in either inches or mm. If units in the program are specified in the non-standard system of measurement, you must first switch over the system of measurement with G71/G70. The controller then converts the units to the new system of measurement.

Programming

G70 ; Dimension unit inches
G71 ; Dimension units metric

The following geometrical dimensions are converted:

- Positional data X, Y, Z, ... (linear axes, positioning axes)
- Interpolation parameters I, J, K
- Programmable zero offsets (TRANS)
- Circle radius CR, polar radius RP

All other parameters such as feedrates, tool offsets or settable zero offsets are not converted and refer to the system of measurement configured on the controller.

Rotary axes are always programmed in degrees.

Example:

```
N10 G70 X10 Y30 ; Dimension unit inches, modal
N11 X40 Y50 ; G70 active until canceled by G71
...
N80 G71 X19 Y17.3 ; Dimension unit metric, modal
... ; G71 active until canceled by G70
```


10.2.7 Plane selection (G17, G18, G19)

General

The geometry axes describe a rectangular, cartesian coordinate system. With G17, G18 and G19 you can select the individual planes.

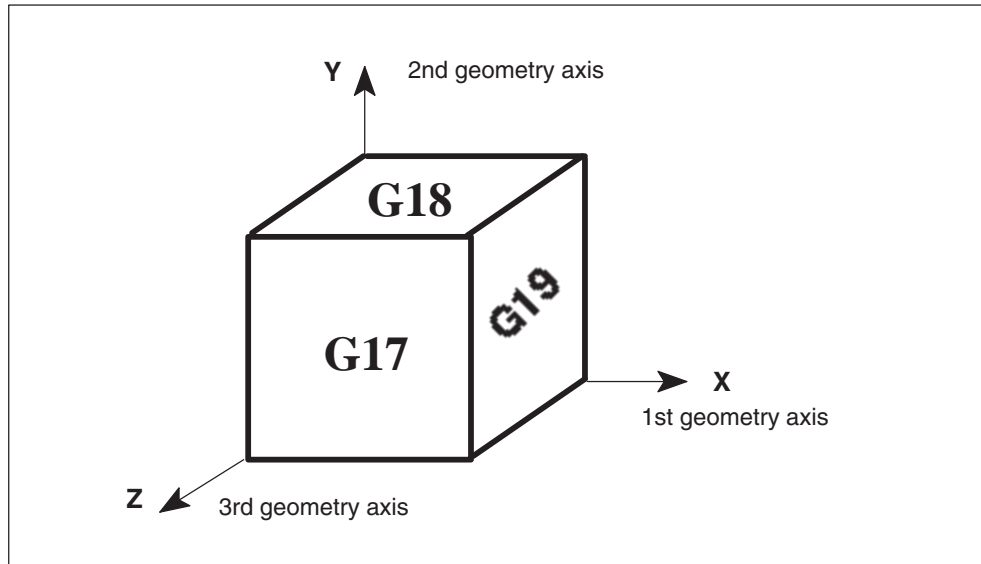


Fig. 10-15 Plane and axis assignment

Programming

Statement	Plane (abscissa/ordinate)	Perpendicular axis to plane (applicate)
G17	X/Y	Z (default)
G18	Z/X	Y
G19	Y/Z	X

The specification of the plane defines the effect of the tool length compensation (see Section 10.16).

10.3 Zero offsets (frames)

General

The zero offset defines the position of the workpiece zero in relation to the machine zero.

There are three components

- Offset
- Rotation of the workpiece coordinate system (WCS)
- Mirroring of the WCS

The components for rotation and mirroring are only possible if three geometry axes (complete Cartesian coordinate system) are available.

10.3.1 Settable zero offsets (G54, G55, G56, G57, G500, G53)

General

The values for the settable zero offset are entered in the data field provided (using the parameterization tool, see Section 5.5.3, and/or the OP).

There are four possible groups of settable zero offset. They are activated or deactivated by programming.

The parameters can be used to define a default setting that is active on power-up. This is active in all operating modes. The setting activated in the program remains active after program interruptions and mode changes.

Programming

G54	; 1st settable zero offset
G55	; 2nd settable zero offset
G56	; 3rd settable zero offset
G57	; 4th settable zero offset
G500	; Settable zero offset off – modal
G53	; All zero offsets off – non-modal

G54, G55, G56, G57

These statements belong to a G group and are alternately operative.

The stored values are activated when you program G54 to G57.

When you change or deactivate the zero offset, an override compensating movement occurs in the next traversing block. This always produces a **resulting movement** (as opposed to a separate compensating movement) and is possible with all types of interpolation.

Deactivation of G53, G500

Statement G53 deactivates set zero offsets non-modally.

The G500 statement deactivates zero offsets until canceled by G54, G55, G56, G57.

Examples Representation:

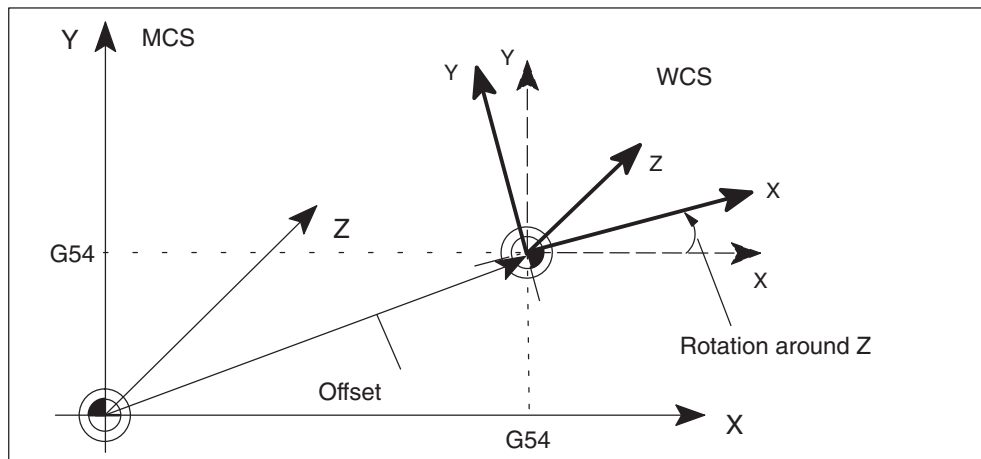


Fig. 10-16 Settable zero offset G54 (shift and rotation)

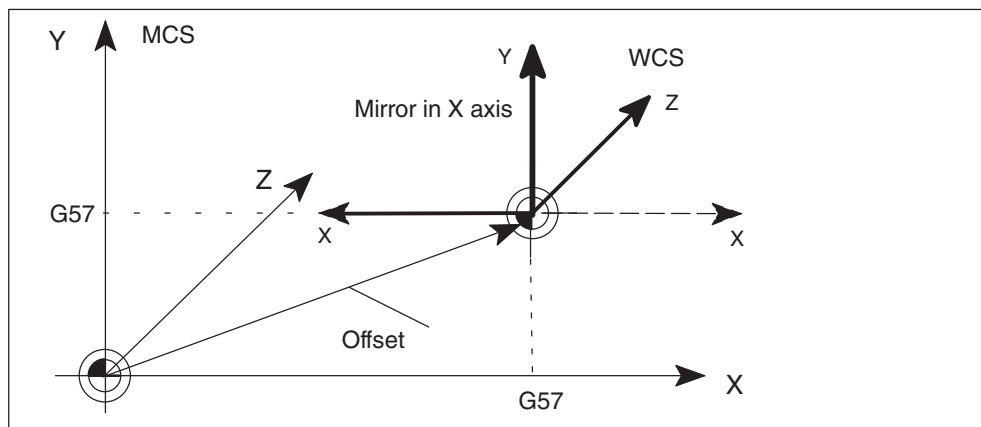


Fig. 10-17 Settable zero offset G57 (shift and mirror)

Programming:

```

N10 G54 ...           ; Call first settable zero offset
N20 X10 Y30          ; Approach position X/Y in WCS
...
N90 G500 G0 X100     ; Deactivate settable zero offset,
                    ; Approach position X in MCS
    
```

10.3.2 Programmable zero offsets (TRANS, ATRANS, ROT, AROT, RPL, MIRROR, AMIRROR)

General

Programmable zero offsets are active in addition to settable zero offsets.

They are only effective in the active NC program (program running, program interrupted – irrespective of the mode).

You must specify the value for the shift/rotation in the NC program.

Programming

TRANS	; Programmed zero offset absolute
ATRANS	; Programmed zero offset additive
ROT	; Programmed rotation absolute
AROT	; Programmed rotation additive
RPL	; Angle of rotation in the active plane
MIRROR	; Programmable mirror absolute
AMIRROR	; Programmable mirror additive
G53	; all zero offsets off – non-modal
	; (settable and programmable)

Notice

Programmable zero offsets which are absolute deselect each other.

Additive programmable zero offsets apply in the order in which they were programmed, and are added to all active offsets.

TRANS, ATRANS

Statements TRANS and ATRANS are operative for path and positioning axes.

TRANS and ATRANS must be programmed in separate blocks.

The zero offset is deselected by setting the offset values for each individual axis to zero or by writing TRANS in the abbreviated form without specifying an axis.

Example:

```
N5 ...  
...  
N10 TRANS X2.5 Y8.43 ; Offset, absolute  
...  
N100 TRANS X60 Y40 ; New offset, absolute  
...  
N150 ATRANS X2 Y4 ; Additive offset, total offset  
; (with N100) X = 62, Y = 44  
...  
N170 TRANS ; Deselection of all programmable  
; zero offsets
```

ROT, AROT

The WCS can be rotated about each of the three geometry axes with statement ROT or AROT. This rotation can be programmed **only** for geometry axes.

The sign of the programmed angle of rotation defines the direction of rotation.

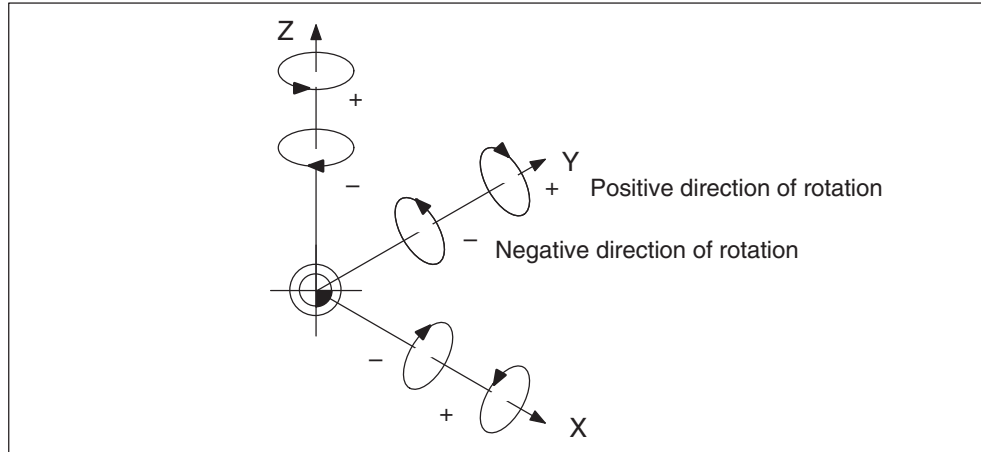


Fig. 10-18 Directions of the angle of rotation

The following order is defined for rotation around multiple axes within a ROT statement:

1. About 3rd geometry axis (Z)
2. About 2nd geometry axis (Y)
3. About 1st geometry axis (X)

The rotation is deselected by setting the offset values for each individual axis to zero or simultaneously for all axes valid in the abbreviated form ROT without an axis name.

Example:

```
N9 TRANS Z...
N10 AROT X30 Y45 Z90
```

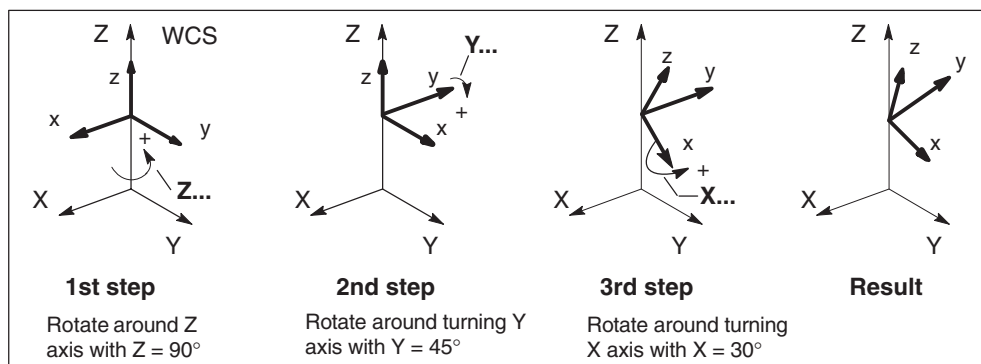


Fig. 10-19 Order of rotation for three angle dimensions in **one** block

RPL

Statement ROT or AROT can be programmed in conjunction with address RPL (instead of axis addresses) to rotate the WCS in the plane activated with G17 to G19.

This form of programming enables a rotation of the plane with **only** two geometry axes.

The rotation is deselected by setting the rotation values for each individual axis or RPL to zero or by writing ROT in the abbreviated form without specifying an axis.

Notice

If a plane change (G17 to G19) is programmed when a rotation is active, the programmed angles of rotation around the axis are retained. If necessary, you should deactivate the rotation first.

Examples:

1. Shift, then rotate:

```
N10 TRANS X... Y...  
N11 AROT RPL=...
```

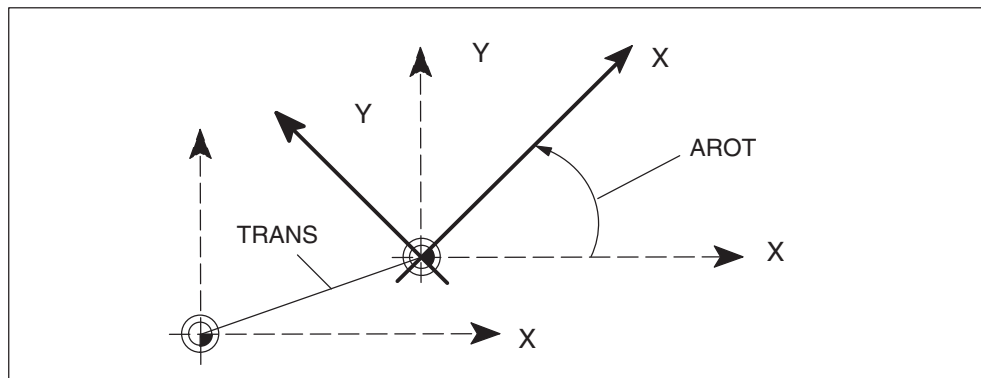


Fig. 10-20 RPL – Shift, then rotate

2. Rotate, then shift:

```
N10 ROT RPL=...
N11 ATRANS X... Y...
```

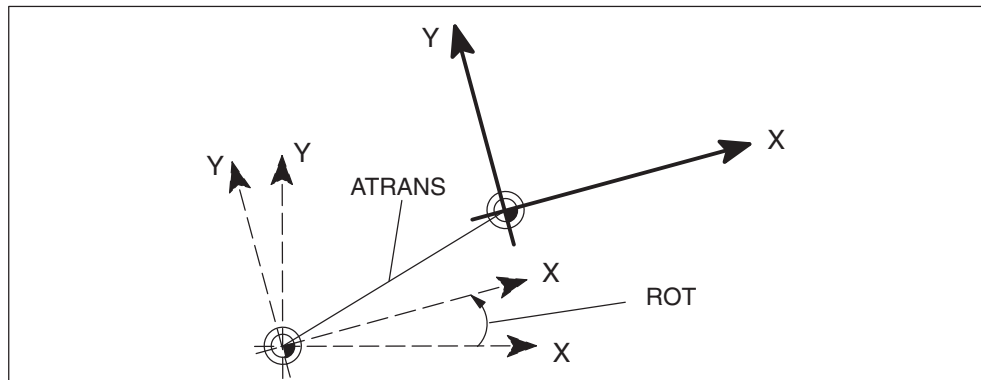


Fig. 10-21 RPL – Rotate, then shift

MIRROR, AMIRROR

Statement MIRROR, AMIRROR can be programmed to mirror the WCS in the specified geometry axis. You can **only** mirror geometry axes.

The axis which is mirrored is specified by the axis name and a value of zero.

The mirror is deselected by specifying MIRROR without an axis.

Example:

```
N10 MIRROR X0
...
N50 MIRROR
```

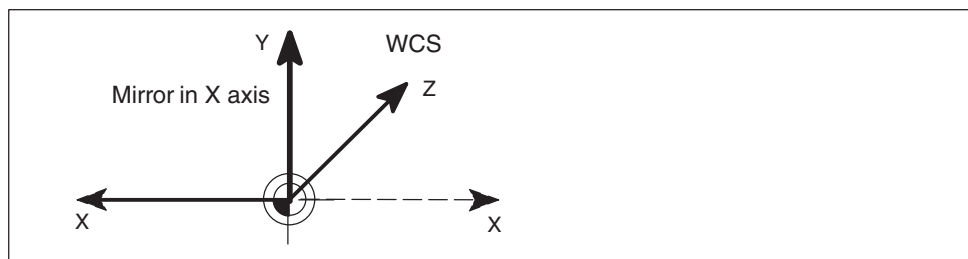


Fig. 10-22 Mirror in X axis

10.4 Set actual value (PRESETON)

General

For special statements, it may be necessary to assign a new, programmed actual value to one or several axes stationary at the current position.

Programming

```
PRESETON(MA,IW) ; Set actual value  
                ; MA – machine axis  
                ; IW – actual value
```

PRESETON

Actual values are assigned in the machine coordinate system. The values refer to the machine axis.

Example:

```
N10 G0 X=200 ; Axis X travels to position 200 in WCS  
N20 PRESETON(X1, 0) ; X1 receives the new position 0 in MCS  
                ; From here, positioning takes place in the new  
                ; actual value system.
```

Notice

The reference point value is invalidated with the function PRESETON. If the original system is to be restored, you must perform a reference point approach or set the old actual value with PRESETON.

10.5 Programming axis movements

Overview

In this section, you can find information about:

- Programming feeds (F, FA, FL), page 10-32
- Feed interpolation (FNORM, FLIN, FCUB), page 10-33
- Path group (FGROUP), page 10-36
- Linear interpolation with rapid traverse (G0), page 10-37
- Positioning movements at rapid traverse (G0, RTLION, RTLIOF), page 10-38
- Linear interpolation with feed (G1), page 10-39
- Positioning movements (POS, POSA, WAITP), page 10-40
- Programmable block change for positioning axes (FINEA, COARSEA, IPOENDA, IPOBRAKE), page 10-41
- Circular interpolation (G2, G3, I, J, K, CR), page 10-44
- Spline (ASPLINE, CSPLINE, BSPLINE), page 10-47
- Polynomial interpolation (POLY), page 10-54
- Involute interpolation (INVCW, INVCCW), page 10-58
- Chamfer and rounding (CHF, CHR, RND, RNDM, FRC, FRCM), page 10-62

10.5.1 Programming feeds (F, FA, FL)

Programming

F... ; Path feed, only **one** F value can exist in the same block
 FA[axis]=... ; Feed for positioning axes
 FL[axis]=... ; Limit feed for synchronized axes

Feed value for linear axes: mm/min or inch/min
Feed value for rotary axes: degrees/min

Value range $0.001 \leq F \leq 999\,999.999$ [mm/min]
 $399\,999.999$ [inch/min]

Feedrate for path axes F

The path feed is programmed in address **F** and is operative only for path axes.

Feedrate for positioning axes FA

FA[axis]=... ; Feed for the specified positioning axis,
 FA is modal

Feed for synchronized axes

Two methods can be used to program the feed for synchronized axes.

1. Only one synchronized axis is programmed in a block.

Example:

```
N5 G0 G90 A0
N10 G1 G91 A3600 F10000 ; The axis travels with F10000
```

2. Both path and synchronized axes are programmed in a block. In this case, the synchronized axes are moved such that they require the same time as the path axes to cover their distance. All axes arrive at the end point at the same time.

Example:

```
N5 G0 G90 X0 Y0 A0
N10 G1 G91 X100 Y100 A720 ; The A axis traverses in synchronism
                          ; with the path motion of axes X and Y.
                          ; All axes reach their end point at the same
                          ; time.
```

Limit feed FL

Statement **FL[axis]=...** can be programmed to define a limit feedrate for the synchronized axis. The function is modal.

10.5.2 Feed interpolation (FNORM, FLIN, FCUB)

General

In addition to constant feedrate F , it is possible to program a distance-dependent feed characteristic for path axes. The feed is always an absolute value irrespective of the G90/G91 setting.

The function is available for the FM 357-2LX.

Programming

FNORM ; Constant feed characteristic
FLIN ; Linear feed characteristic
FCUB ; Cubic characteristic (spline)

FNORM

This statement switches to constant feed characteristic (see Section 10.5.1). Step changes in feedrate are approached at maximum acceleration rate.

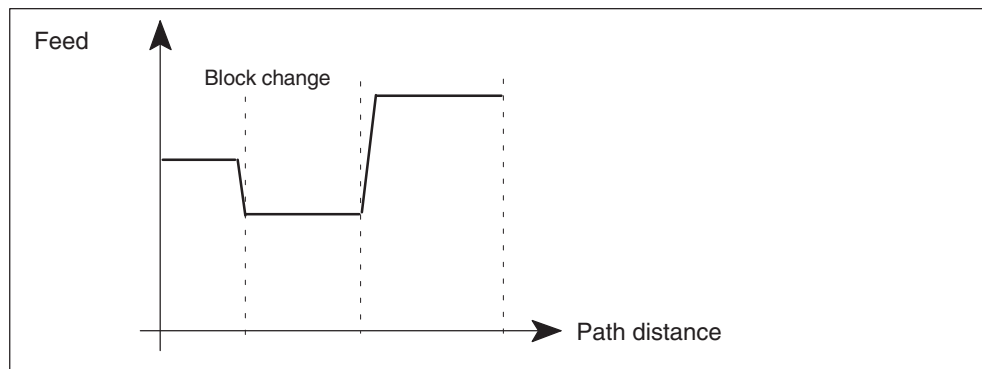


Fig. 10-23 Example of constant feed characteristic

FLIN

The feedrate has a linear characteristic from the current feed to the programmed value at the block end.

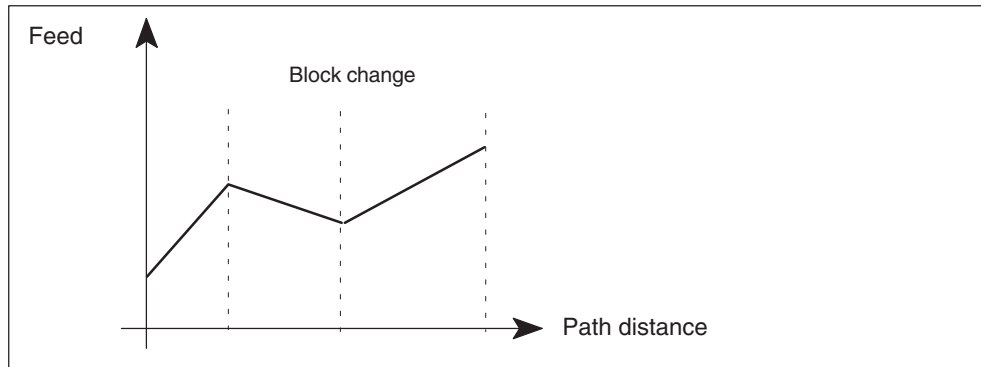


Fig. 10-24 Example of linear feed characteristic

FCUB

The feedrate has a cubic characteristic from the current feed to the programmed value at the block end. When FCUB is active, the FM links the programmed feed values by cubic splines (see Section 10.5.10).

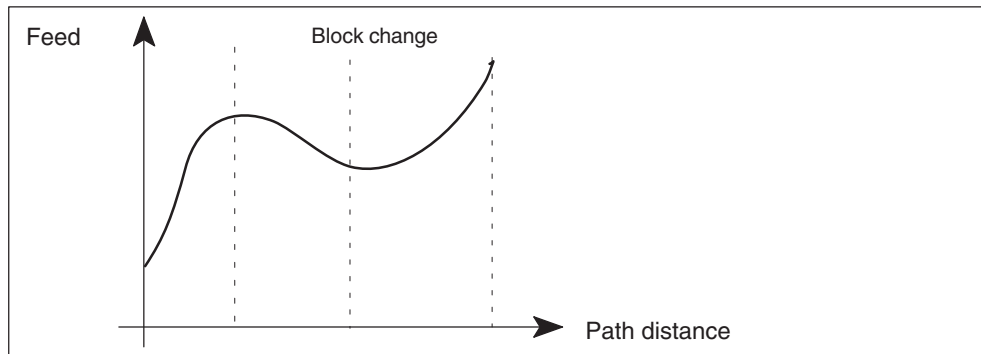


Fig. 10-25 Example of cubic feed characteristic

Programming example

```

N10 G1 G64 G91 X0 FNORM F100 ; Constant feed
N20 X10 F200
N30 X20 FLIN F300 ; Linear feed from 200 to 300 mm/min
N40 X30 F200 ; Linear feed from 300 to 200 mm/min
N50 X40 FCUB F210 ; Cubic feed, all others
N60 X50 F430 ; Feed points are linked as splines
;
N70 X60 F500
N80 X70 FNORM F400 ; Constant feed 400 mm/min
N90 M02

```

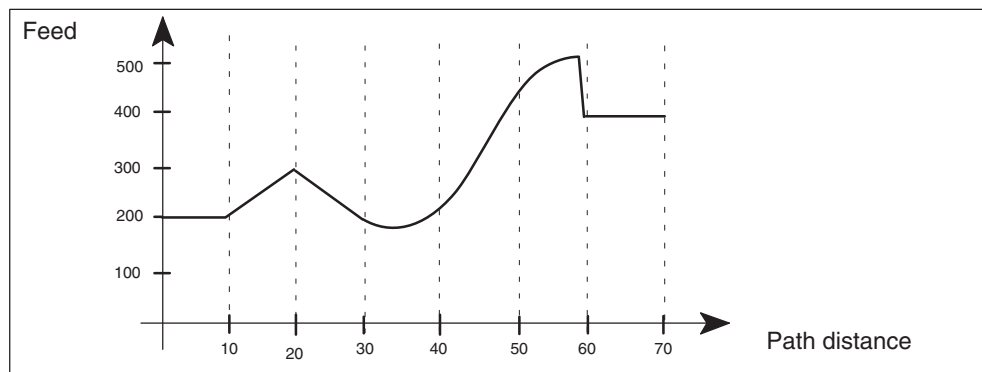


Fig. 10-26 Example of feed interpolation

10.5.3 Path group (FGROUP)

General

The feed programmed with F only applies to the path axes (geometry axes) programmed in the block. FGROUP can be used to include a synchronized axis in the calculation of the path feed or to exclude a path axis from the calculation.

Programming

FGROUP(axis,axis,...) ; Axes to be included in the feed group calculation
 FGROUP() ; No axis specified, the default configuration is restored
 FGREF[axis]= ; Reference radius for rotary axes

Programming example 1

... ; X, Y, Z are path axes
 N10 FGOUP(X,Y) ; The path feed is to be traversed with X and Y
 N20 X100 Y100 Z1 F100 ; X and Y traverse at a path velocity of
 ; 100 mm/min. The Z axis is synchronized with
 ; the path group.

Programming example 2

N10 FGROUP(X,Y,Z,A) ; X, Y, Z are path axes, A is a synchronized axis
 N20 X100 Y100 A360° F100 ; The path velocity resulting from the X, Y and A axis
 ; amounts to
 ; 100 mm/min

Rotary axes

If linear and rotary axes are associated via FGROUP, the feedrate is interpreted in the unit of measurement of the linear axis.

The tangential velocity of the rotary axis is interpreted in mm/min or inch/min.

$$F_T \text{ [mm/min]} = \frac{U \text{ [deg/min]} * p * 2R}{360}$$

R is the reference radius of the rotary axis, and can be defined with FGREF[axis]. If no FGREF[axis] is programmed, the following reference radius applies:

$$R = 360 / \text{mm} / (2p) = 57.296 \text{ mm}$$

This is equivalent to 1 degree = 1 mm.

10.5.4 Linear interpolation with rapid traverse (G0)

General

The path programmed with G0 is traversed at the highest possible velocity, i.e. rapid traverse, along a straight line (linear interpolation). If more than one axis is programmed in the block, the path velocity is determined by the axis which requires the longest time for its part of the path movement. The path velocity is a function of all velocity components and can be greater than the rapid traverse of the fastest axis.

The controller monitors the maximum permissible axis velocity. When you program G0, the feed programmed under F is saved and is reactivated later, e.g. with G1. G0 can be executed with all path axes in the block.

Programming

```
G0 X.. Y.. Z.. ; Travel with rapid traverse to end point of straight
; line
```

Example:

```
...
N5 G0 G90 X10 Y10 ; Linear interpolation with rapid traverse from
; P1 to P2
...
```

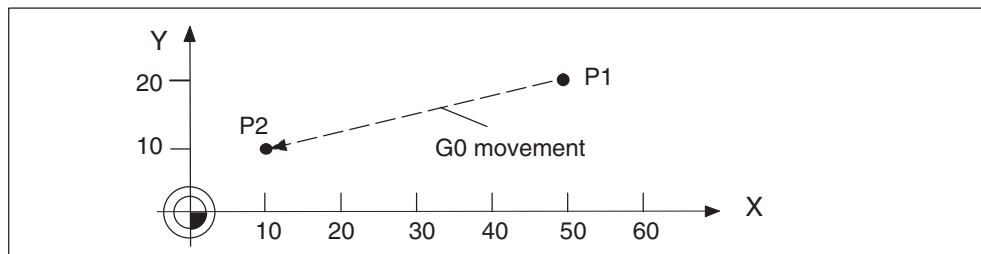


Fig. 10-27 Linear interpolation with rapid traverse

10.5.5 Positioning motion at rapid traverse (G0, RTLION, RTLIOF)

General

With software version 5 and higher, it is possible to traverse a positioning motion at rapid traverse.

The path programmed with G0 is traversed at the maximum possible velocity. The axes will traverse independently of each other to the programmed end position at their appropriate rapid traverse rates.

If G0 and G64 are programmed, linear interpolation is always used for traversing.

Programming

G0 X.. Y.. Z.. RTLIOF ; Rapid traverse with positioning motion

G0 X.. Y.. Z.. RTLION ; Rapid traverse with linear interpolation

Example:

```
...
N5 G0 G90 X10 Y10 RTLION ; Rapid traverse with positioning motion
; from P1 to P2
...
```

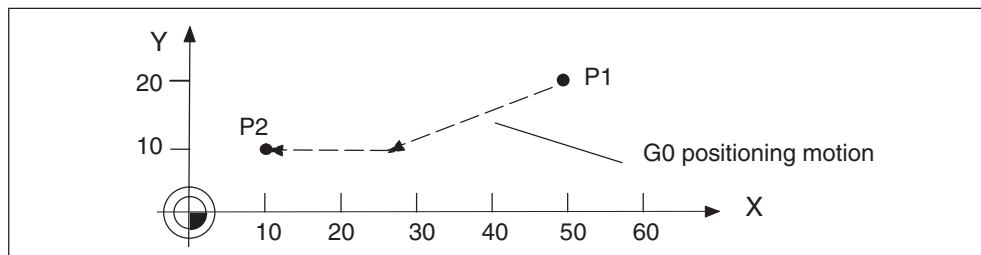


Fig. 10-28 Positioning motion at rapid traverse

10.5.6 Linear interpolation with feed (G1)

General

The axis traverses a straight path from the start to the end point.

The programmed F word determines the path velocity.

Programming

G1 X... Y... Z.. F... ; Travel with feed F to end point of straight line

Example:

```
N5 G0 X50 Y20 ; Linear interpolation with rapid traverse to P1
N10 G1 X10 Y10 F500 ; Linear interpolation with feed 500 mm/min from P1
; to P2
```

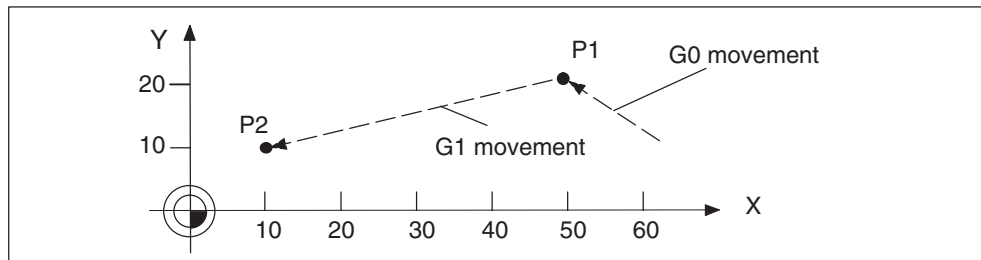


Fig. 10-29 Linear interpolation with feed

The feedrate F in mm/min is only valid for the path axes. If additional axes are programmed, these travel in the same time, as synchronized axes.

10.5.7 Positioning movements (POS, POSA, WAITP)

General

Positioning axes are traversed at their own, axis specified feedrate irrespective of the path axes or G commands (G0, G1, G2, G3, ...). Any axis can be traversed as a positioning axis within a block.

Path axes programmed with POS or POSA are removed from the path axis group for this block.

Programming

POS[axis]=... ; Positioning movement with impact on block changeover
POSA[axis]=... ; Positioning movement with no effect on switch to next block
WAITP(axis) ; Wait until position reached

POS

The block change is delayed until the axis has reached its position.

POSA

The positioning axis can traverse beyond the block limit, i.e. the block change is not affected by this positioning axis.

WAITP

This statement must be programmed in a separate block. It can be used to pause the program until a positioning axis programmed with POSA has reached its end position.

It is possible to wait for several positioning axes with this statement.

Example:

WAITP(X, Y) ; Wait for X and Y

Positioning statement

Example:

```

...
N9 POS[V]=500 FA[V]=2180 ; Position and feed for axis V
N10 POSA[U]=900 FA[U]=180 ; Position and feed for axis U
N11 X10 Y20
N12 X13 Y22
N13 WAITP(U) ; Wait until axis U has reached its position,
; then switch to next block
N14 X... Y...
...

```

10.5.8 Programmable block change for positioning axes (FINEA, COARSEA, IPOENDA, IPOBRAKE)

General

With software version 5 and higher, you can define the block change time for positioning axis motions. The behavior is equivalent to the target range G601 and G602 for path axes. The block change is carried out if the relevant block change conditions are fulfilled for all path and positioning axes in the block.

The set block change time is modally active and is maintained even beyond Reset.

Programming

FINE[axis]	; Block change at “Exact stop fine“
COARSEA[axis]	; Block change at “Exact stop coarse“
IPOENDA[axis]	; Block change at “IPO end“
IPOBRKA(axis, % value)	; Block change at % value of the braking ramp ; 0% according to IPO end ; 100% according to the beginning of the braking ramp

The statement IPOBRKA() must be programmed in a separate block.

System variable \$AA_MOTEND[axis]

The programmed end of motion can be read from this system variable:

\$AA_MOTENDA[axis]:	
1	Block change “Exact stop fine“
2	Block change “Exact stop coarse“
3	Block change “IPO end“
4	Block change “Braking ramp“

Example

The positions for the individual block change times are to be recorded for two axes traversed at different feedrates.

```

N10 DEF INT IND                                ; Auxiliary counter

N20 FOR IND=0 TO 50
N30  R[IND] = 0                                ; Delete R0 to R50
N30 ENDFOR

N40 R1=10 R2=20 R3=0 R4= 0                    ; Setpoint positions for X and Y
                                           ; Rectangle 20 x 20

N50 G0 X0 Y0
N60 FA[X]=1000 FA[Y]=1500

N70 $AC_MARKER[0]=5                           ; Identifier and index

CYCLE:

N100 FINEA[X] FINEA[Y]                        ; Set exact stop fine
N110 IF $AC_MARKER[0] == 5 GOTOF POS_XY

N120 COARSEA[X] COARSEA[Y]                   ; Set exact stop coarse
N130 IF $AC_MARKER[0] == 10 GOTOF POS_XY

N140 IPOENDA[X] IPOENDA[Y]                   ; Set IPO end
N150 IF $AC_MARKER[0] == 15 GOTOF POS_XY
N160 IPOBRKA(X, 50)                           ; Set braking ramp of 50 %
N170 IPOBRKA(Y, 50)
N180 IF $AC_MARKER[0] == 20 GOTOF POS_XY

N200 M30

POS_XY:
                                           ; Position and save block change position using a synchronized action

N300 POS[Y] = R1

N310 EVERY TRUE DO $R[$AC_MARKER[0]] = $R1 - $VA_IM[Y]
N320 POS[X] = R2

N330 EVERY TRUE DO $R[$AC_MARKER[0]] = $R2 - $VA_IM[Y]
N340 POS[Y] = R3

N350 EVERY TRUE DO $R[$AC_MARKER[0]] = $VA_IM[Y]
N360 POS[X] = R4

```

```
N370 EVERY TRUE DO $R[$AC_MARKER[0]] = $VA_IM[Y]
N380 G4 F1
```

```
N390 STOPRE
N400 $AC_MARKER[0] = $AC_MARKER[0] + 5
```

```
N410 GOTOB CYCLE
```

```
N420 M30
```

The following differences between setpoint position and block change position have been determined using the example. The axis parameterization corresponds to the standard values.

FINEA	COARSEA	IPOENDA	IPOBRKA 50%
R5 = 0.008	R10 = 0.030	R15 = 1.13	R20 = 1.769
R6 = 0.008	R11 = 0.032	R16 = 0.721	R21 = 1.144
R7 = 0.008	R12 = 0.030	R17 = 1.13	R22 = 1.769
R8 = 0.008	R13 = 0.032	R18 = 0.721	R23 = 1.144

10.5.9 Circular interpolation (G2, G3, I, J, K, CR)

General

The axis traverses a circular path from the start to the end point. Arcs or complete circles can be traversed clockwise or counterclockwise. The tool path velocity is determined by the programmed F value.

Programming

G2 X... Y... I... J... ; Circular interpolation clockwise
 G3 X... Z... I... K... ; Circular interpolation counterclockwise
 X... Y... Z... ; Circle end points
 I... J... K... ; Interpolation parameter for determining the
 ; arc center
 CR= ; Circle radius

Direction of rotation around circle G2, G3

To determine the direction of circular rotation for G2 and G3, the control needs a plane specification (G17, G18 or G19). The direction of rotation is defined according the selected plane.

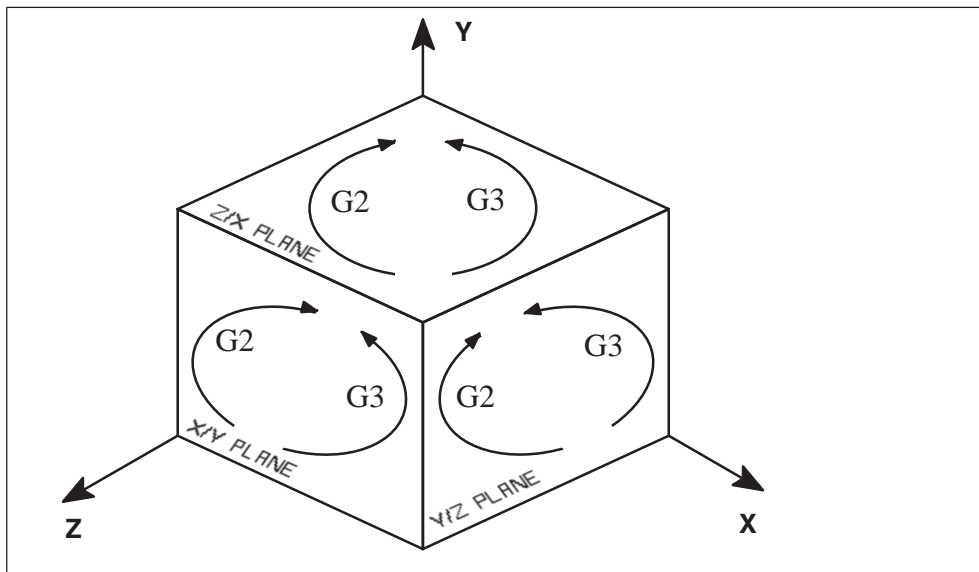


Fig. 10-30 Direction of circle rotation in the planes

Circle end points X, Y, Z

The circle end point can be specified as an absolute or incremental dimension with G90 or G91.

Interpolation parameters I, J, K

The arc center is described with I, J and K.

- I – Coordinate of the circle center in X direction
- J – Coordinate of the circle center in Y direction
- K – Coordinate of the circle center in Z direction

The standard interpretation of the center point coordinates I, J, K, is as incremental dimensions with reference to the starting point of the circle.

The absolute center point is specified non-modally with I=AC(...) J=AC(...) K=AC(...). The center point refers to the workpiece zero.

Example of incremental dimensions:

N5 G90 X30 Y40 ; Circle starting point for N10
 N10 G2 X50 Y40 I10 J-7 ; End point and center point

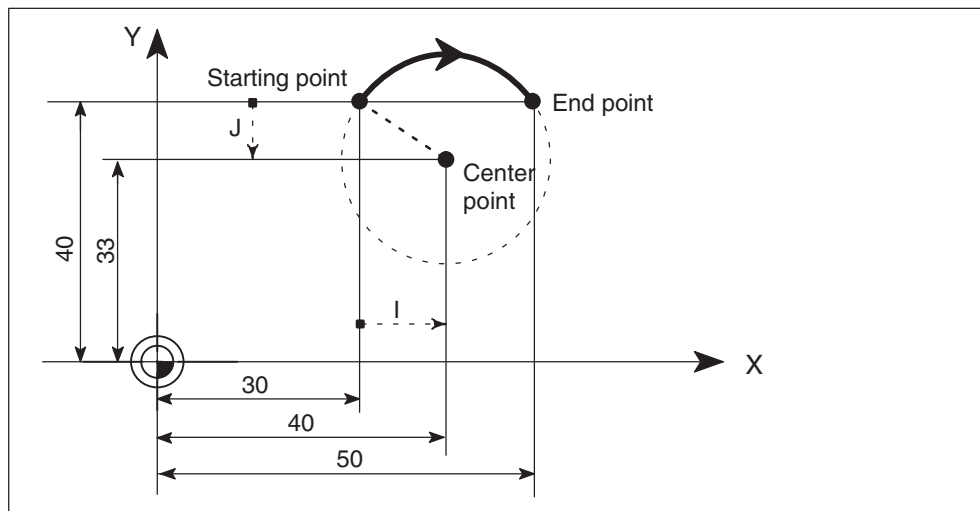


Fig. 10-31 Example of center point and end point dimensions

Circle radius CR

The circle radius is described with CR.

CR=+... ; Angle less than or equal to 180 degrees (+ can be omitted)

CR=-... ; Angle greater than 180 degrees

A full circle is not possible with this form of programming.

Example:

N5 G90 X30 Y40 ; Circle starting point for N10

N10 G2 X50 Y40 CR=12.207 ; End point and radius

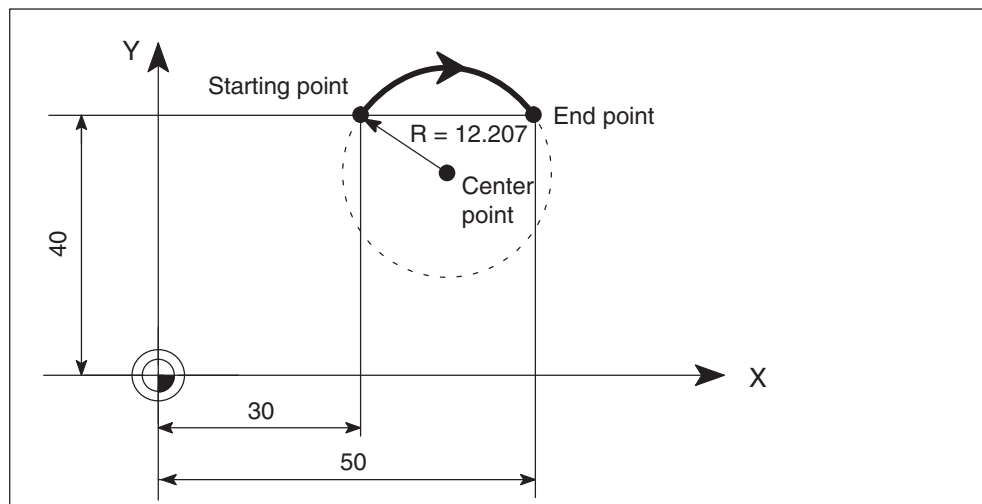


Fig. 10-32 Example of end point and radius dimensions

Input tolerances for circle

With circular interpolation, the control uses the programmed data to calculate the circle and, where there are deviations between the programmed data and calculation, sets the exact arc center internally.

This is only performed within a tolerance which you can set in the parameters. Deviations outside the tolerance cause an error.

10.5.10 Spline (ASPLINE, CSPLINE, BSPLINE)

General

The function is available for the FM 357-2LX.

Spline interpolation allows you to connect programmed sequences of points through continuous curve transitions.

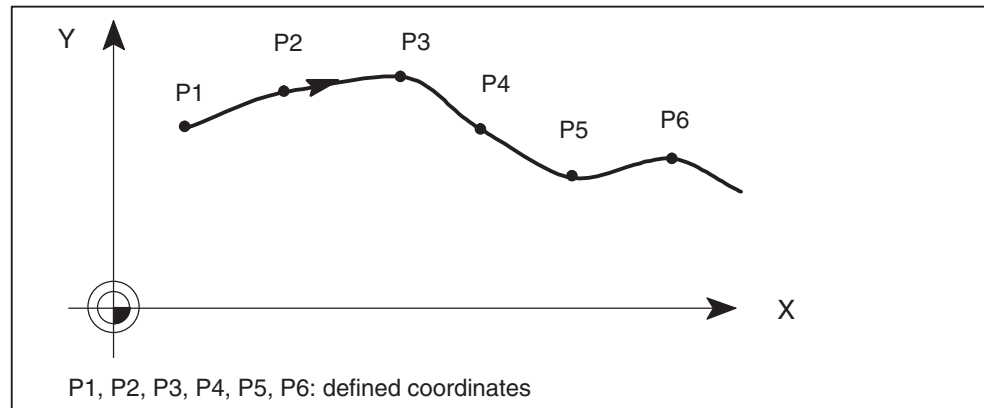


Fig. 10-33 Spline interpolation

3 types of spline are supported:

- ASPLINE
- CSPLINE
- BSPLINE

These statements belong to the first G group (G0, G1, G2, G3, ...).

Path axes which are to be included in a spline grouping (i.e. to form a spline curve) can be selected using the language command **SPLINEPATH**.

Programming

```
ASPLINE      ; Akima spline
CSPLINE      ; Cubic spline
BSPLINE      ; B spline
```

ASPLINE

The akima spline runs as a continuous tangent exactly through the programmed positions (interpolation points), but is not continuously curved at the nodes. The advantage of the akima spline is its proximity to the intermediate points, which avoids unwanted oscillations such as occur with the CSPLINE. The akima spline is local, i.e. a change in an intermediate point only has an impact on six neighboring blocks.

This spline should be used when measured points are to be interpolated smoothly.

A third-degree polynomial is used.

Example: ASPLINE, tangential transitions at the start and end

```

N10 G1 F200 G64 X0 Y0
N20 X10
N30 ASPLINE X20 Y10      ; Spline interpolation, P1
N40 X30                  ; P2
N50 X40 Y5              ; P3
N60 X50 Y15             ; P4
N70 X55 Y7              ; P5
N80 X60 Y20             ; P6
N90 X65 Y20             ; P7
N100 X70 Y0             ; P8
N110 X80 Y10            ; P9
N120 X90 Y0             ; P10
N130 M2
...

```

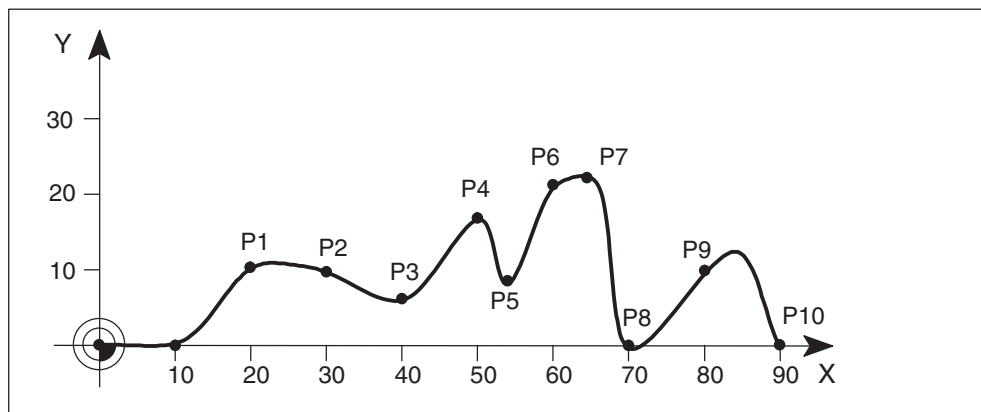


Fig. 10-34 ASPLINE

CSPLINE

The cubic spline differs from the akima spline in that it has continuously curved transitions at the nodes. This is the most widely known and used type of spline. The advantage of continuous curvature is offset by the disadvantage of unexpected oscillations. It is suitable when the points on an analytically known curve are to be calculated, and where oscillations can be eliminated by the insertion of additional intermediate points. It is also appropriate when continuous curvature is a requirement.

The spline is not local, i.e. a change in an intermediate point can have an impact on a large number of blocks (with decreasing intensity).

A third-degree polynomial is used.

The parameter interval is calculated internally. The distance between two consecutive nodes is equal to the distance between the two intermediate points.

Example: CSPLINE, curvature 0 at the start and end

```

N10 G1 X0 Y0 F300
N15 X10
N20 BNAT ENAT           ; Curvature 0 at the start and end
N30 CSPLINE X20 Y10     ; CSPLINE, P1
N40 X30                 ; P2
N50 X40 Y5              ; P3
N60 X50 Y15             ; P4
N70 X55 Y7              ; P5
N80 X60 Y20             ; P6
N90 X65 Y20             ; P7
N100 X70 Y0             ; P8
N110 X80 Y10            ; P9
N120 X90 Y0             ; P10
N130 M2

```

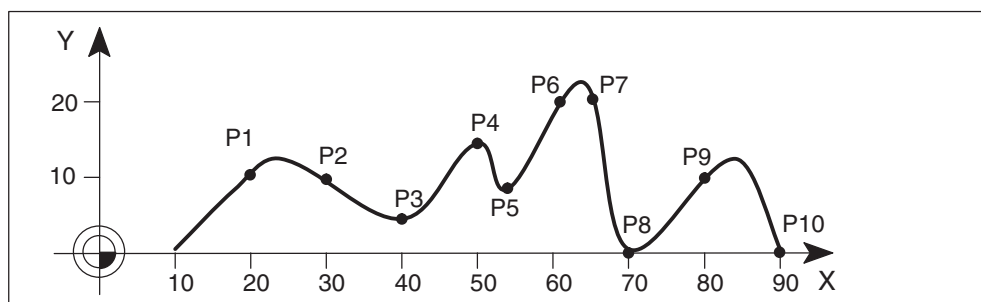


Fig. 10-35 CSPLINE

Notice

If a tangential transition is not possible (e.g. because no connecting path exists), BAUTO or EAUTO is executed (see conditions).

Supplementary conditions for ASPLINE and CSPLINE

The transition characteristics (start or end) of these spline curves can be set by means of two groups of statements with three commands each (can be treated like G group).

Start of spline curve

- BAUTO** – No command, start results from the position of the first points
- BNAT** – Zero curvature
- BTAN** – Tangential transition to previous block (initial setting)

End of the spline curve

- EAUTO** – No command, end results from the position of the last points
- ENAT** – Zero curvature
- ETAN** – Tangential transition to next block (initial setting)

The above statements must be programmed, at the latest, in the block with ASP-LINE or CSPLINE. Once the spline has been started, you cannot make changes.

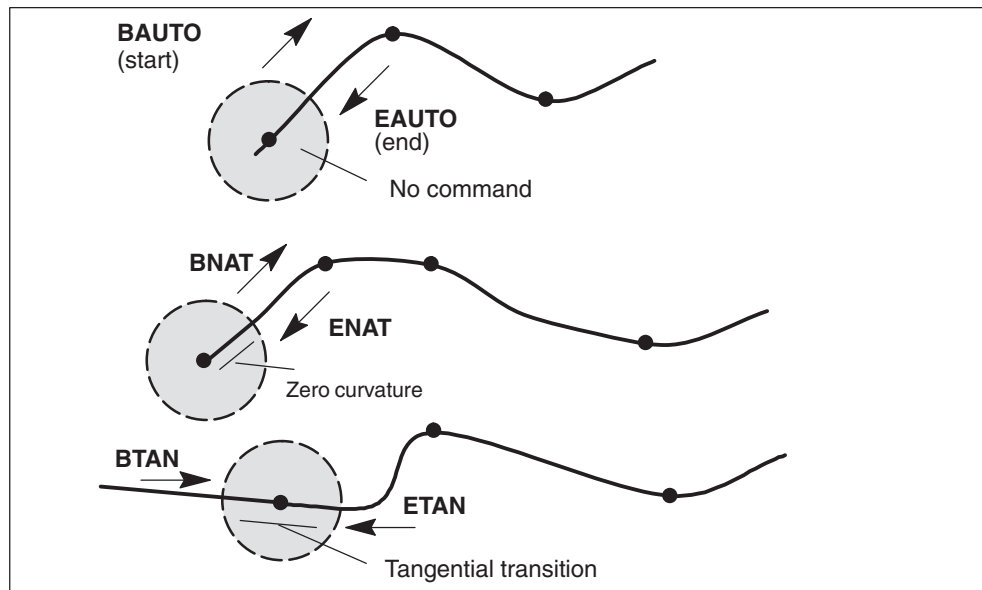


Fig. 10-36 Conditions for ASPLINE and CSPLINE

BSPLINE

On a B spline, the desired degree can be programmed (2 or 3) with **SD=**. If no degree is programmed at the start of the spline, 3 is the default value.

The programmed positions are not intermediate points, but are simply “control points” of the spline. The curve runs past, but not necessarily through the control points, which determine the shape of the curve. The control polygon of the spline connects the control points by straight lines, thus providing the initial approximation for the curve. You obtain the control polygon by programming G1 instead of BSPLINE.

A quadratic B spline (SD=2) touches the control polygon tangentially between two control points and is closer to the control polygon than a cubic B spline (SD=3).

Supplementary conditions for BSPLINE

The curve is always tangential in relation to the check polygon at the start and end points. No start and end conditions can be programmed.

You can program an additional weight for each control point with PW (point weight). This has the effect of drawing the curve towards the control point (PW > 1). All sections of a cone (parabola, hyperbola, ellipse, circle) can be obtained exactly through the use of suitable weights.

This spline is an optimum tool for the creation of sculptured surfaces, and is the preferred form on CAD systems.

A 3rd degree B spline combines the advantages of akima and conventional cubic splines. There are no undesired oscillations in spite of continuously curved transitions.

Point weight PW:

A weight parameter can be specified for each control point with address **PW=...**

The curve is drawn towards the control point if PW>1 and is pushed further away if PW<1.

Value range of PW: Positive, 0 to 3 in increments of 0.0001

Spline degree SD:

The desired spline degree for BSPLINE is written at address **SD=...**

Value range: 2 or 3

If no address SD= is programmed, SD=3 is the default.

Distance between nodes PL:

The distance between two nodes is programmed with **PL=...**

Value range: same as path dimension

If no node distances are programmed, suitable spacing is calculated internally.

Example: BSPLINE, all weights 1

```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30
N60 X40 Y45
N70 X50 Y0
...
```

Example: BSPLINE, various weights

```
N10 G1 X0 Y0
N20 BSPLINE PW=0.3
N30 X10 Y20 PW=2
N40 X20 Y30
N50 X30 Y35 PW=0.5
N60 X40 Y45
N70 X50 Y0
...
```

Example: associated control polygon

```
N10 G1 X0 Y0
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30 PW=0.5
N60 X40 Y45
N70 X50 Y0
...
```

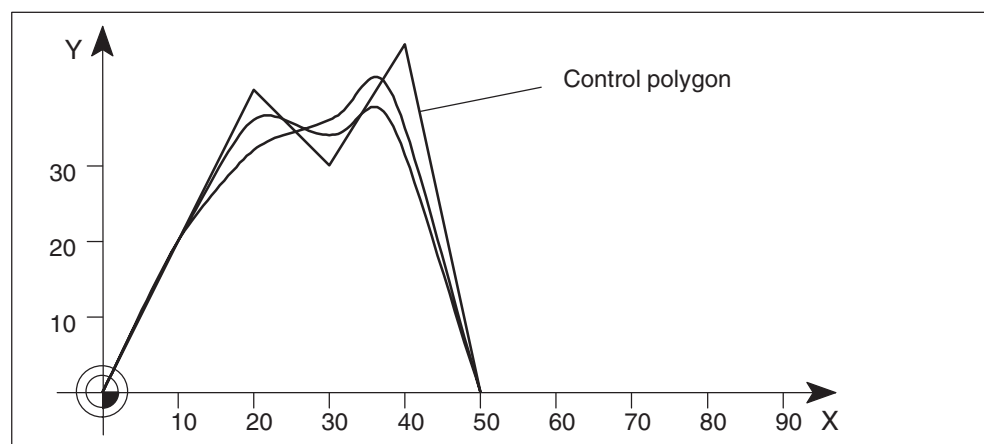


Fig. 10-37 BSPLINE, associated control polygon

Spline group SPLINEPATH

This statement is used to define the axes involved in the spline. A maximum of 5 axes is possible.

If SPLINEPATH is not programmed, the first 3 axes of the channel are traversed as a spline group.

A special block is used for the definition. The block contains the following statement:

```
SPLINEPATH(n,X,Y,Z,... ) ; n = 1, fixed value X,Y,Z,... path axis names
```

Example:

```
N10 G1 X10 Y20 Z30 F350
N11 SPLINEPATH(1,X,Y,Z)
N13 CSPLINE BAUTO EAUTO X20 Y30 Z40
N14 X30 Y40 Z50
...
N100 G1 X... Y... ; Deselect spline interpolation
...
```

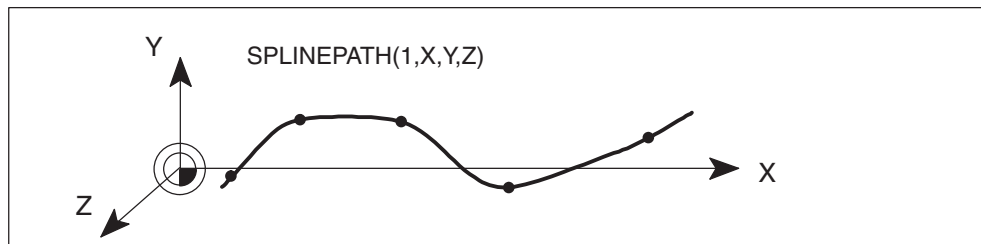


Fig. 10-38 Spline grouping, e.g. with three path axes

10.5.11 Polynomial interpolation (POLY)

General

The FM is able to traverse curves (paths) in which each selected path axis follows a function (polynomial max. 3rd degree).

The function is available for the FM 357-2LX.

The general form of the polynomial function is:

$$f(p) = a_0 + a_1p + a_2p^2 + a_3p^3$$

That means:

a_n : Constant coefficients

p : Parameters

By assigning specific values to the coefficients, a wide variety of curves, e.g. linear, parabolic, exponential, etc., can be generated.

For example, setting coefficients $a_2 = a_3 = 0$ produces a straight line with:

$$f(p) = a_0 + a_1p$$

The following applies:

a_0 = Axis position at the end of the previous block

a_1 = Axis position at the end of the definition area (PL)

Programming

POLY PO[X]=(x_{e1}, a_2, a_3) PO[Y]=(y_{e1}, b_2, b_3) PO[Z]=(z_{e1}, c_2, c_3) PL= n

POLY	; Activate polynomial interpolation
PO[]=(\dots, \dots, \dots)	; End points and polynomial coefficients
X_e, Y_e, Z_e	; Specifies the end position for the respective axis; Value range as for path dimension
$a_2, a_3,$; Coefficients a_2 and a_3 are written with their value; Value range as for path dimension. The last coefficient can be omitted if it has a value of zero.
PL	; Length of the parameter interval at which the polynomials are defined (definition area of function $f(p)$). The interval always starts with 0. p can have values from 0 to PL. The theoretical value range for PL: 0.0001...99 999.9999. The PL value is valid for the block in which it is programmed. If no PL is programmed, PL = 1.

POLY

Polynomial interpolation is incorporated together with G0, G1, G2, G3, A-Spline, B-Spline and C-Spline in the first G group. When it is active, it is not necessary to program the polynomial syntax. Axes which are only programmed with their name and end point are traversed across a linear path to their end point. If all axes are programmed in this way, the control system response is the same as for G1.

Polynomial interpolation is deactivated by any other command of the G group (e.g. G0, G1).

Polynomial coefficient PO[]=(...)

The PO value (PO[]=) specifies all polynomial coefficients for an axis. Several values are separated by commas, according to the degree of the polynomial. Different polynomial degrees are possible for different axes within a block.

Programming example

```

N10 G1 X0 Y0 Z0 F600
N11 POLY PO[X]=(1,2.5,0.7) PO[Y]=(0.3,1,3.2) PL=1.5
                                ; Polynomial interpolation on
N12 PO[X]=(0, 2.5,1.7) PO[Y]=(2.3,1.7) PL=3
...
N20 M8 H126 ...
N25 X70 PO[Y]=(9.3,1,7.67) PL=5          ; Different parameters for the axes
N27 PO[X]=(10,2.5) PO[Y]=(2.3)         ; PL not programmed; PL=1
N30 G1 X... Y... Z...                  ; Polynomial interpolation off
...

```

Example of a curve in the X/Y plane

```
N9 X0 Y0 G90 F100
N10 POLY PO[Y]=(2) PO[X]=(4,0.25) PL=4
```

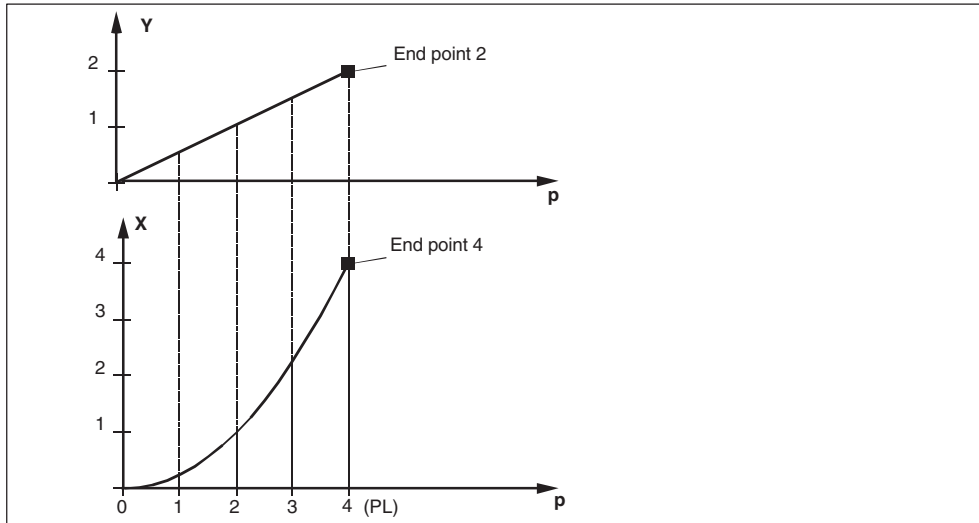
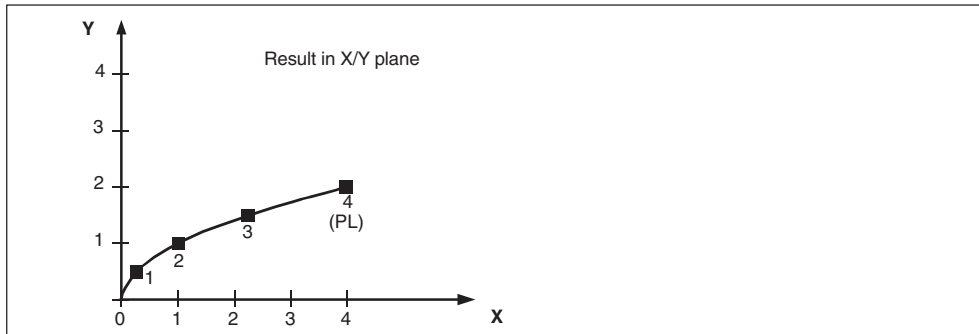


Fig. 10-39 Example of a curve in the X/Y plane (polynomial interpolation)

Result in X/Y plane



Special feature: denominator polynomial

A common denominator polynomial can be programmed for the geometry axes with PO[]=(...) without specifying an axis name, i.e. the motion of the geometry axes is interpolated as the quotient of two polynomials.

This allows, for example, cone sections (circle, ellipsis, parabola, hyperbola) to be represented accurately.

Example

```
POLY G90 X10 Y0 F100 ; Linear movement of geometry axes
; to position X10, Y0
PO[X]=(0,-10) PO[Y]=(10) PO[ ]=(2,1) ; Geometry axes traverse in
; quadrant to X0, Y10
```

The constant coefficient (a_0) of the denominator polynomial is always assumed as 1, the specified end point is independent of G90/G91.

The result is as follows:

$$X(p)=10(1-p^2)/(1+p^2) \text{ and } Y(p)=20p/(1+p^2) \text{ where } 0 \leq p \leq 1$$

The following intermediate values are produced as a result of the programmed start points, end points, coefficient a_2 and $PL=1$:

$$\text{Numerator (X)}=10+0*p-10p^2$$

$$\text{Numerator (Y)}=0+20*p+0*p^2$$

$$\text{Numerator} = 1+2*p+1*p^2$$

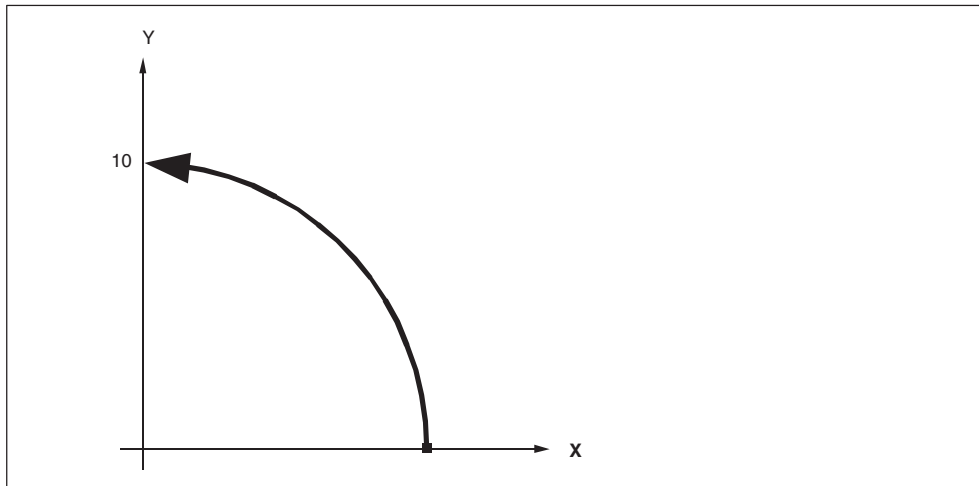


Fig. 10-40 Example denominator polynomial

When polynomial interpolation is activated and a denominator polynomial is programmed with zero digits within the interval $[0, PL]$, the operation is denied with an error. The denominator polynomial has no effect on the movement of special axes.

10.5.12 Involute interpolation (INVCW, INVCCW)

General

Using the interpolation types linear interpolation, circular interpolation and polynomials or splines, a helical contour can be represented only approximately. Using the involute interpolation, it is possible to program any circle involute in a block and to interpolate with accuracy. The path velocity is determined by the programmed F value.

The involute of the circle is a curve which is described by a thread tensioned firmly from the end point and wound off a circle (basic circle).

The function “Involute interpolation” is available as from software version 5.

Programming

INVCW	; Involute interpolation CW
INVCCW	; Involute interpolation CCW
X... Y... Z...	; Involute end points
I... J... K...	; Coordinates of the center point of the base circle relative to ; to starting point
CR=	; Radius of the base circle of the involute
AR=	; Angle of rotation between starting and end points, ; alternatively to the involute end points

The interpolation of the involute is carried out in the plane defined by G17, G18 or G19. If – when programming the end point – in addition, a coordinate outside the plane is specified, an involute results in the space.

Starting and end points must therefore be located outside the base circle.

The specification of the end point on the involute can be carried out in two different ways:

- Programming of the end point coordinates X, Y, Z

If the end point is not exactly on the involute defined by the starting point, the interpolation is carried out between both involutes defined by starting and end points. The maximum deviation of the end point, however, may not be greater than 0.01 mm.

- by specifying the angle of rotation between starting and end points

AR > 0: Involute will move away from the base circle

AR < 0: Involute will move towards the base circle

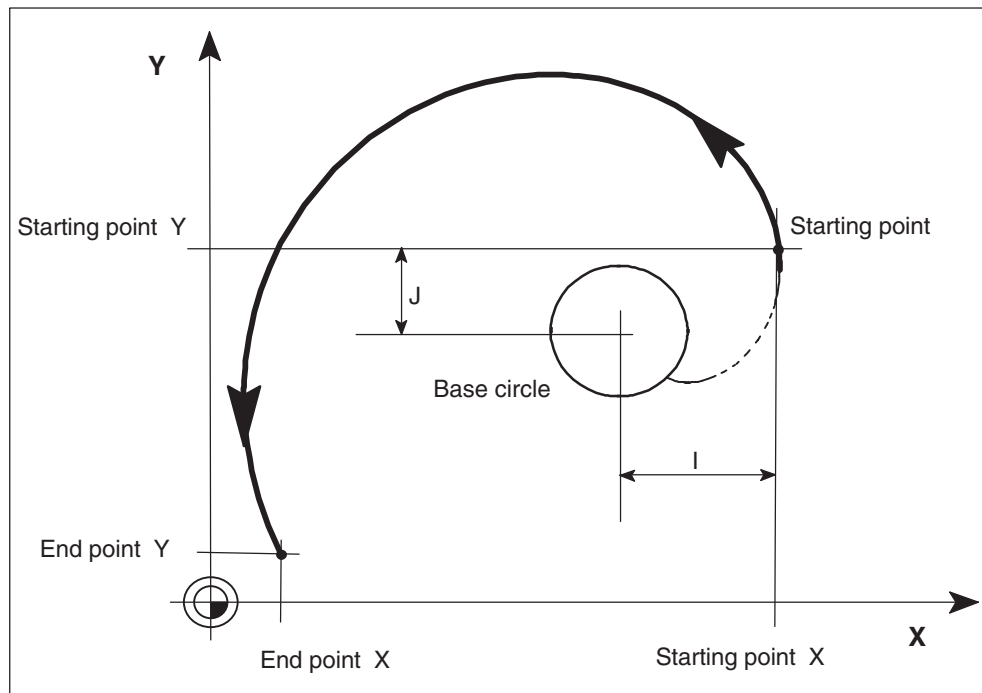


Fig. 10-41 Circle involute CW, programming of the end point coordinates

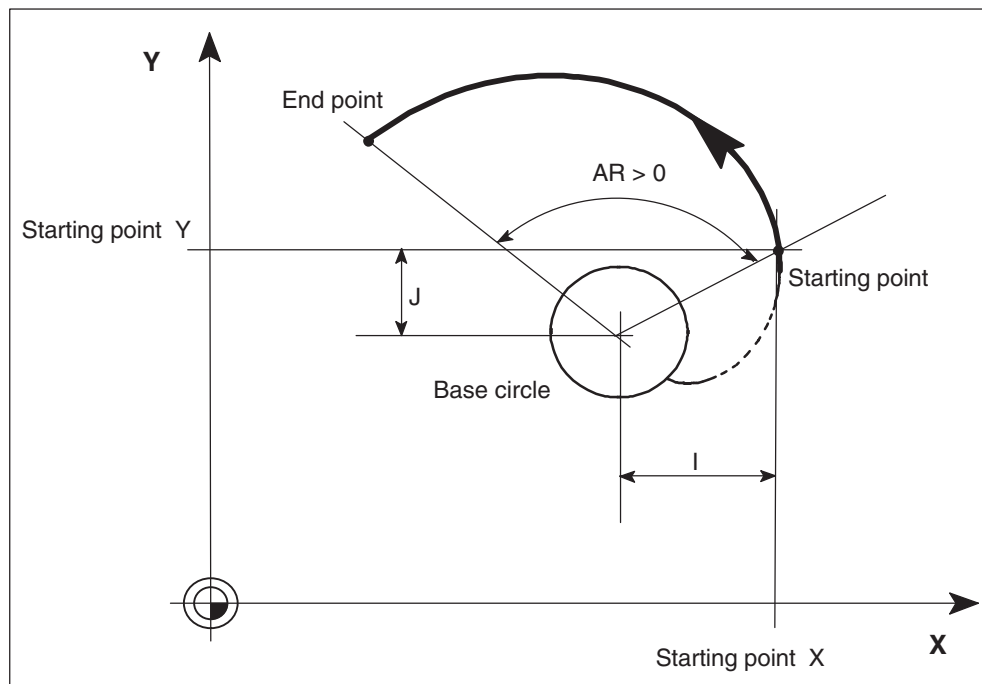


Fig. 10-42 Circle involute CCW, programming of the angle of rotation AR

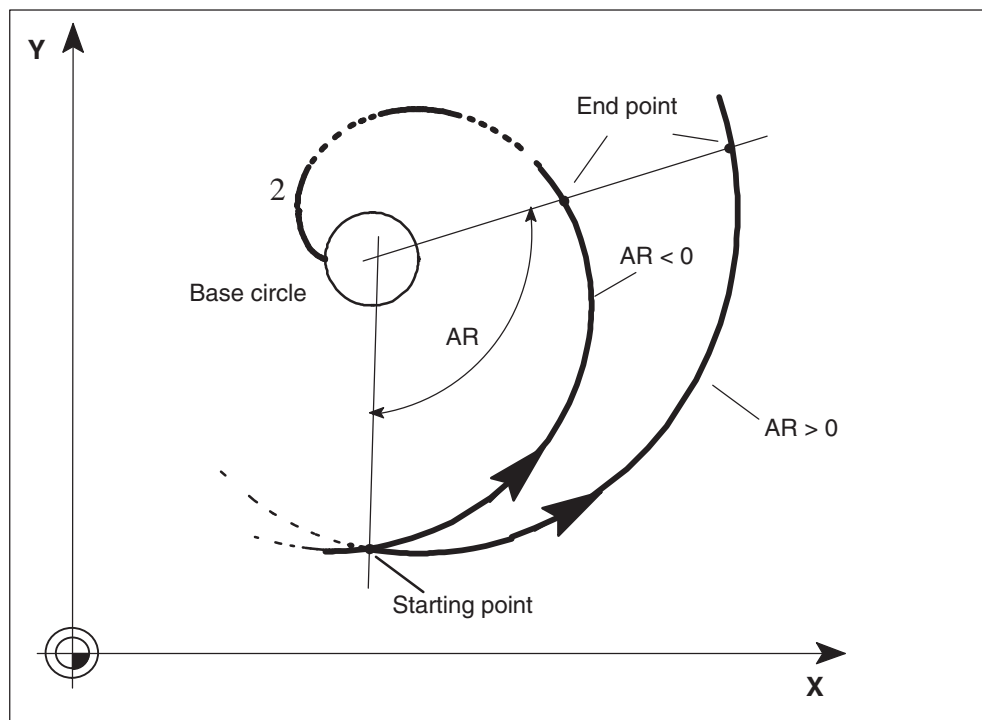


Fig. 10-43 Influence of the sign of the angle of rotation AR

Involute geometry

The coordinates of the plane are interpolated using the following interrelation:

$$X(\phi) = X_0 + R[\cos \phi + (\phi - \phi_0) \sin \phi]$$

$$Y(\phi) = Y_0 + R[\sin \phi - (\phi - \phi_0) \cos \phi]$$

R is the radius and (X_0, Y_0) is the center point of the base circle. The angle ϕ is interpolated linearly between the initial value ϕ_s and the end value ϕ_e . The starting and the end angles are defined by the starting point and/or the programmed end point (or the programmed angle of rotation).

The offset angle ϕ_0 is defined by the starting point and the programmed direction of rotation (INVCW or INVCCW).

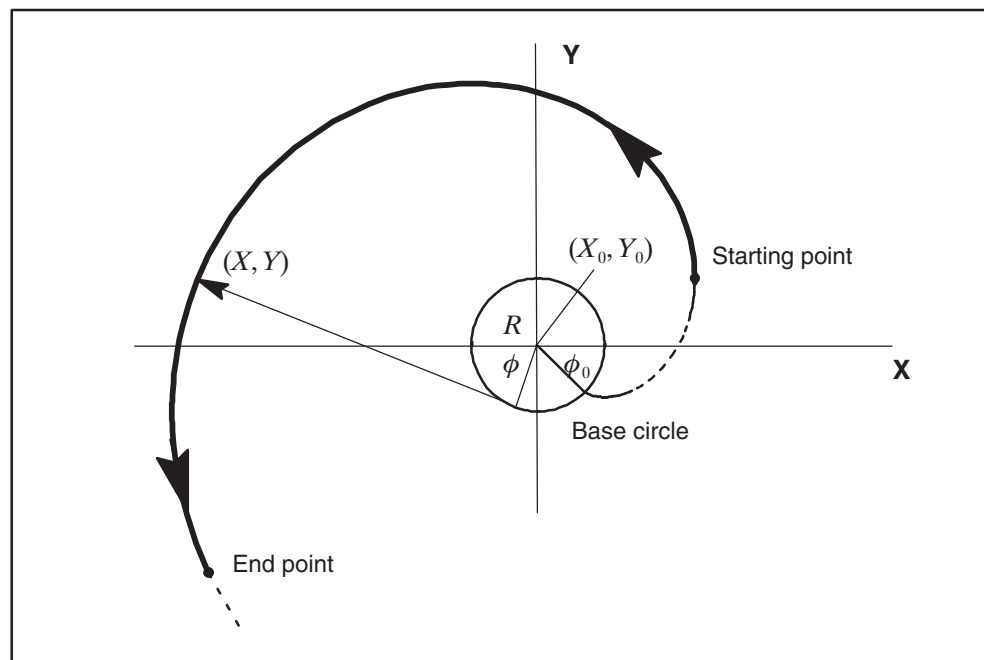


Fig. 10-44 Involute geometry

Example 1

An involute is interpolated in the CCW direction from the starting point X_{10}, Y_0 to the end point $X_{32.777}, Y_{32.777}$. The base circle with the center point X_0, Y_0 has a radius of 5 mm.

```

N10 G1 X10 Y0 F5000           ; Starting point
N15 G17                       ; in the X/Y plane
N20 INVCCW X32.777 Y32.777 CR=5 I-10 J0 ; Involute CCW
N30 INVCW X10 Y0 CR=5 I-32.777 J-32.777 ; The same involute reverse in the
                                           ; CW direction

```

Example 2

The end point is specified via the angle of rotation. An involute is interpolated with 3.5 revolutions, starting from the starting point X_{10}, Y_0 . Both the end point and the starting point are located on the X axis.

```

N10 G1 X10 Y0 F5000           ; Starting point
N15 G17                       ; in the X/Y plane
N20 INVCCW CR=5 I-10 J0 AR=3.5* 360 ; Involute CCW

```

10.5.13 Chamfer and rounding (CHF, CHR, RND, RNDM, FRC, FRCM)

General

A chamfer or a rounding can be inserted between linear and/or circle blocks. The insertion is carried out by cutting the adjacent blocks. A chamfer or rounding can only be created in the active plane (G17 ... G19) between geometry axes.

This function is provided as of software release 5.2.

Programming

CHF = ; Chamfer; value corresponds to the chamfer length
 CHR = ; Chamfer; value corresponds to the distance from chamfer
 ; start to the programmed end-of-block position
 RND = ; Rounding; value corresponds to the radius
 RNDM = ; Rounding modal; value corresponds to the radius
 RNDM = 0 ; Deselect rounding, modal
 FRC = ; Feedrate chamfer/rounding, non-modal
 FRCM = ; Feedrate chamfer/rounding, modal
 FRCM = 0 ; Deselect feedrate chamfer/rounding, modal

Chamfer CHF, CHR

A straight line is inserted between two blocks (G0, G1, G2, G3) in the active plane using CHF or CHR.

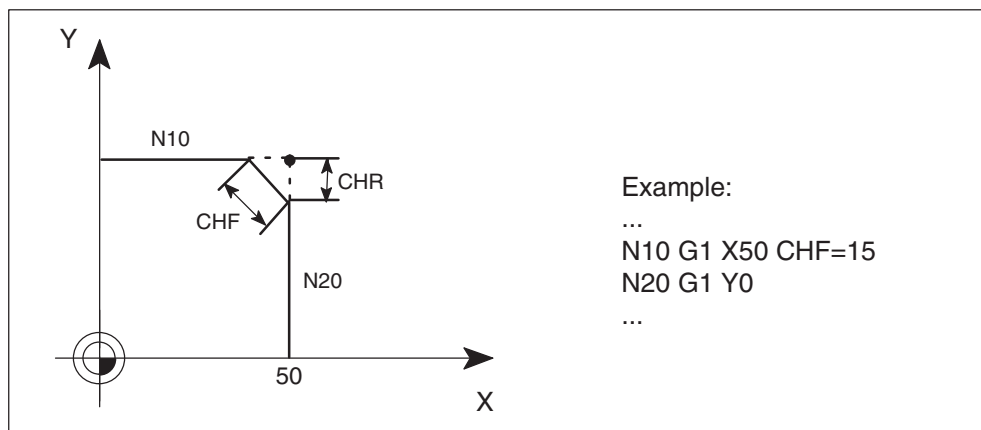


Fig. 10-45 Example of a chamfer inserted between two G1 blocks, with G17 active

The connection angle between chamfer and previous block (N10), as well as between chamfer and subsequent block (N20) is always the same, even with any combination of linear and circle blocks.

Rounding RND, RNDM

An arc is inserted between two blocks (G0, G1, G2, G3) in the active plane using RND or RNDM.

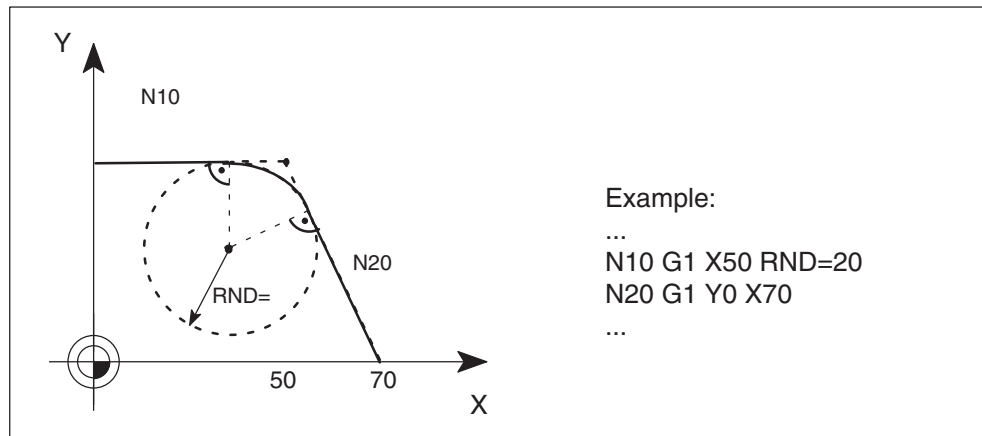


Fig. 10-46 Example of a rounding inserted between two G1 blocks, with G17 active

With a straight line/straight line transition, the connection angle between rounding and previous block (N10), as well as between rounding and subsequent block is always tangential.

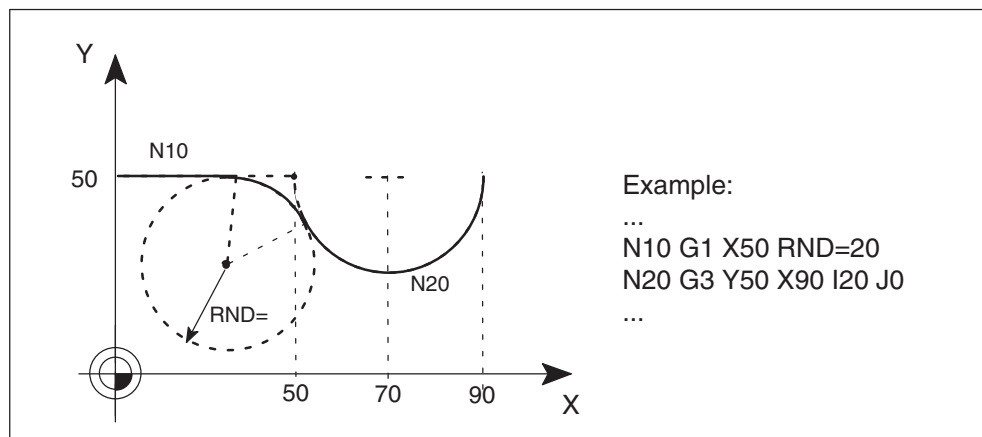


Fig. 10-47 Example of a rounding inserted between the G1 and the G3 blocks, with G17 active

With a straight line/ circle transition or a circle/circle transition, the connection angle between rounding and previous block (N10), as well as between rounding and subsequent block is always the same. A tangential transition is a special case.

Feedrate for chamfer and rounding FRC, FRCM

If the chamfer or rounding is to be traversed with its own feedrate, FRC or FRCM must be programmed. If FRC or FRCM is not programmed, the standard feedrate F will be used for traversing. If G0 is active, FRC or FRCM will not act.

Notes with regard to chamfer or rounding

No chamfer or rounding is inserted if:

- no straight line or circle exists in the plane;
- a motion is performed outside the plane;
- with tangential transition between previous and subsequent blocks
- the plane is changed;
- the maximum number of 3 blocks without movement in the plane is exceeded (error no. 10881, overflow of the local block buffer with chamfers or radii)

If the chamfer or rounding is too large for the adjacent blocks, the chamfer or rounding is automatically reduced to the largest possible value whereby the shorter of the two adjacent blocks will then contain no more movement in the plane.

The chamfer or rounding is derived from the subsequent block. In other words, a path feed programmed there or an M or H command with output prior to the movement will come into effect already in the block for the chamfer or rounding.

Programming example

```
N10 G0 X0 Y0 Z0 G17 F2000
N20 X100 CHF=5           ; Chamfer N20...N30 with rapid traverse
N30 Y100
N40 X0 RND=10           ; Rounding N40...N50 F=2,000 mm/min
N50 Y0 G1 CHF=5         ; Chamfer N50...N70
N60 Z5                  ; The Z motion is carried out before the chamfer
N70 X50 RNDM=2          ; Rounding modal N70...N80 FRC=100
N80 Y50 FRC=100         ; Rounding modal N80...N90 F=2,000 mm/min
N90 X0                  ; Rounding modal N90...N100 F=2,000 mm/min
N100 Y40                 ; No rounding N100 ... N110, since
                        ; tangential transition
N110 Y10 RND=30         ; Rounding 10 mm (reduced), no G1 in Y
N120 X50
N130 M30
```

10.6 Path response

Overview

In this section, you can find information about:

- Exact stop (G9, G60), target range (G601, G602, G03)
- Continuous-path mode (G64, G641, ADIS, ADISPOS)
- Acceleration response (BRISK, SOFT, DRIVE)
- Programmable dynamic limitation

10.6.1 Exact stop (G60, G9), target range (G601, G602, G603)

General

The exact stop functions G60 and G9 can be programmed to move an axis to a target position within specified exact stop limits. When the target range is reached (G601, G602), the axis decelerates and the block change is initiated.

Target range fine and coarse can be defined in parameters.

The G command G601, G602 or G603 determines when the block is ended. If you want to execute a rapid traverse with target range coarse, G602 must be programmed in the block.

Programming

G60	; Exact stop modal
G9	; Exact stop non-modal
G601	; Block change when target range fine has been reached, modal
G602	; Block change when target range coarse has been reached, ; modal
G603	; Block change when IPO end is reached; modal

The statements G601, G602 and G603 are modal and are only active with G9 or G60.

Exact stop G60, G9

If the exact stop function (G60 or G9) is active, the velocity for reaching the exact start position is reduced to zero at the end of the block.

You can use a further modal G group to define when the traversing movement of the block ends and the program changes to the next block.

The choice of target range has a large impact on the overall time where many positioning operations are performed. Fine adjustments take longer.

Fine target range G601

The block change is enabled when all axes have reached the “target range fine” (value in the machine data).

Coarse target range G602

The block change is enabled when all axes have reached the “target range coarse” (value in the machine data).

Quit IPO G603

Block advancing is carried out with "Quit IPO", i.e. if zero is calculated for the desired speed of the axes involved in the interpolation. At this time, the actual value is back by a follow-up component, depending on the dynamic properties of the axes and the path velocity.

Programming example

```
N10 G1 G60 G601 X100 Y100 F200 ; Block change on fine target range
N15 G0 G53 Z0
N20 G0 X300 Y200 G602 ; Block change on coarse target range
N25 G0 Z-200
N30 G1 X400 F500
```

Behaviour at corners

Depending on target range functions G601 and G602, the block transitions (corners) are either sharp or rounded.

With target range fine and coarse, the rounding depends on the target range and following error (see Section 9.6.1).

Example:

```
N1 X... Y... G60 G601 ; or G602
N2 Y...
```

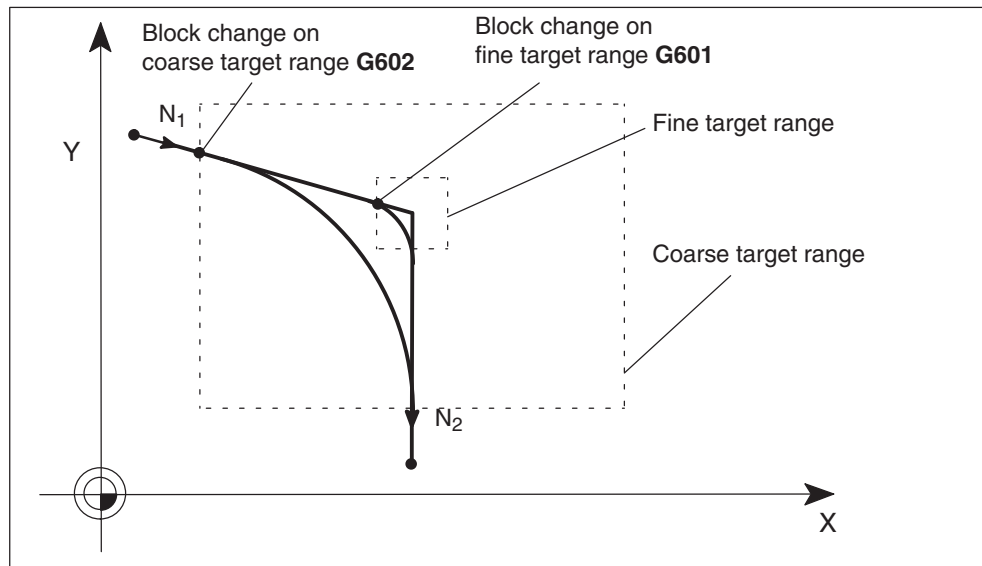


Fig. 10-48 Block change depending on the size of the exact stop limit

10.6.2 Continuous-path mode (G64, G641, ADIS, ADISPOS)

General

The purpose of continuous path mode is to prevent braking at block limits so that the next block can be reached at the most constant path velocity possible (at tangential transitions). G64 and G641 use a predictive velocity control algorithm. At non-tangential path transitions (corners), the velocity is reduced such that a velocity jump greater than the maximum acceleration rate does not occur on any of the axes. This results in velocity-dependent machining of contour corners.

Programming

G64 ; Continuous-path mode
 G641 ; Continuous-path mode with programmed clearance
 ADIS= ; Rounding clearance for path feed G1, G2, G3, ...
 ADISPOS= ; Rounding clearance for rapid traverse G0

All functions are modal.

Continuous-path mode G64

The rounding clearance cannot be programmed. It depends on the following error.

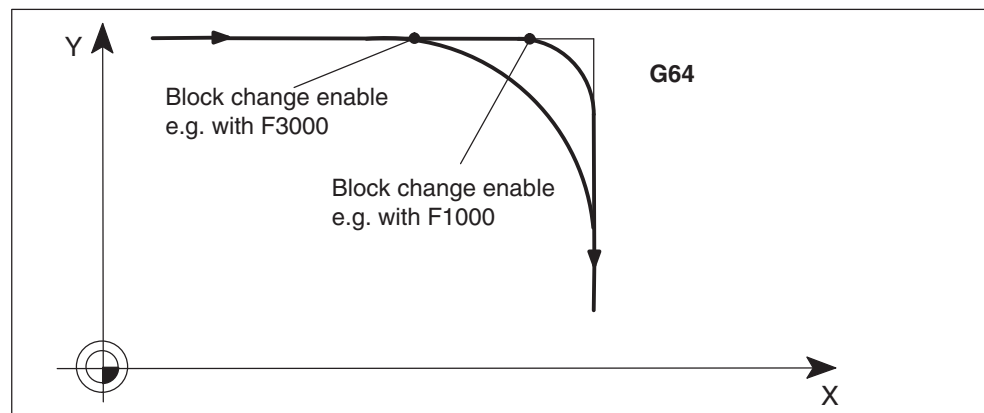


Fig. 10-49 Velocity-dependent machining of contour corners with G64

Continuous-path mode G641, ADIS, ADISPOS

When G641 is active, the control inserts programmed transition elements at contour transitions. The rounding clearance is programmed with ADIS or ADISPOS:

ADIS=... ; for blocks with feed (G1, G2, G3, ...)

ADISPOS=... ; for blocks with rapid traverse G0

Example:

```
N10 G1 G94 X10 Y100 F1000 ; P1
N15 G641 ADIS=0.1 X110 Y80 ; P2
N20 Y8
```

The rounding clearance is basically a circle around the end of block point. The circle radius is specified by ADIS/ADISPOS. The intersection points determine the start and end of the inserted transition element.

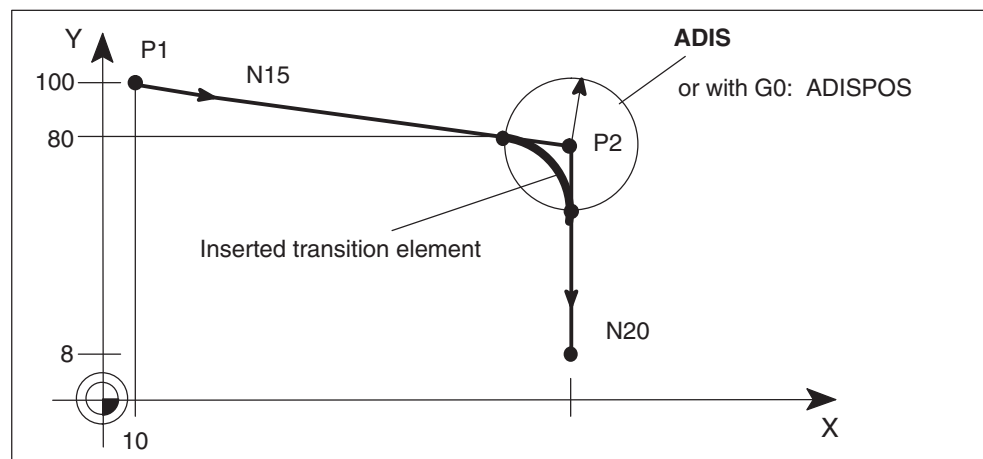


Fig. 10-50 Continuous-path mode with rounding clearance: G641 with ADIS /ADISPOS

If G641 is programmed without ADIS/ADISPOS, a value of 0 applies, and the response is thus the same as G64.

With short traversing paths, the rounding clearance is reduced automatically. At least 75 % of the programmed contour remains.

Example:

```
N10 G0 G90 G60 G602 X0 Y0 Z0 ; Rapid traverse with exact stop coarse
N20 G1 G641 ADIS=0.1 X10 Y10 F500 ; Continuous-path mode with rounding
N30 X20
N40 G9 G601 X30 Y20 ; Modal exact stop fine
N50 X10 ; Switch back to G641, ADIS=0.1
N60 Y10
N70 G0 G60 G602 X... Y... ; Rapid traverse with exact stop coarse
; G60/G9 required
N80...
```


Continuous-path mode over several blocks

This can be achieved by programming path axes in all blocks with traversing motions other than 0. Otherwise the last block in which path axes travel with exact stop is terminated and continuous-path mode is interrupted. Intermediate blocks with only comments, computing blocks or subroutine calls are permitted.

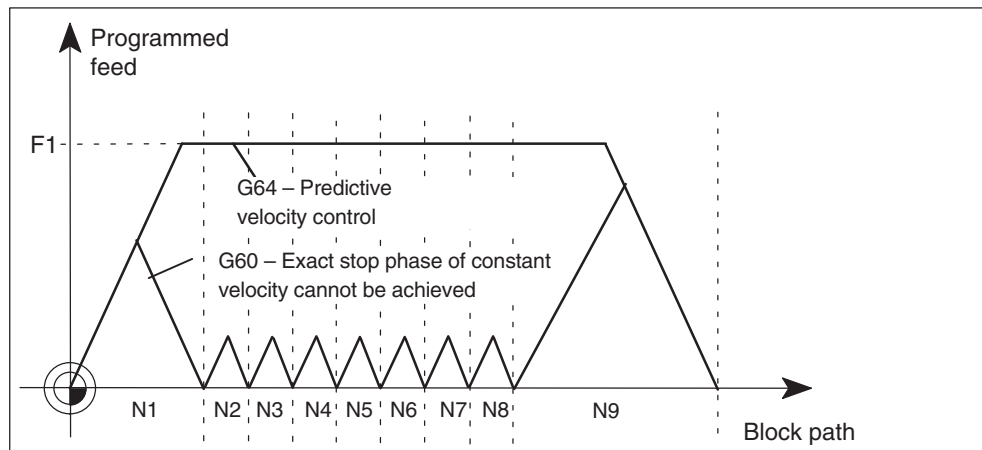


Fig. 10-51 Comparison of velocity response in G60 and G64 with short paths

Positioning axes

G60/G64/G641 are not applicable to positioning axes. They always move after exact stop "fine". If a block has to wait for positioning axes, continuous-path mode is interrupted.

Output of statements

Auxiliary functions that are output at the block end or prior to a motion in the next block interrupt continuous-path mode and generate an internal exact stop.

Rapid traverse

One of the functions named above, i.e. G60/G9 or G64/G641, must also be programmed for rapid traverse mode. Otherwise the default setting in the parameters is used.

10.6.3 Acceleration response (BRISK, SOFT, DRIVE)

General

Statements BRISK, SOFT and DRIVE are programmed to define the active acceleration pattern.

Programming

BRISK ; Brisk acceleration for path axes
BRISKA(...) ; Brisk acceleration for positioning axes
SOFT ; Soft acceleration for path axes
SOFTA(...) ; Soft acceleration for positioning axes
DRIVE ; Reduction in acceleration rate for path axes above a parameterized
; velocity limit
DRIVEA(...) ; Reduction in acceleration rate for positioning axes above a
; parameterized velocity limit

BRISK, BRISKA

When BRISK is selected, the axes accelerate at the maximum rate until the feed velocity setpoint is reached. BRISK enables time-optimized traversing, but is associated with jumps in the acceleration characteristic.

SOFT, SOFTA

When SOFT is selected, the axes accelerate at a constant rate until the feed velocity setpoint is reached. The smooth acceleration characteristic with SOFT enables higher path accuracy and lower machine stress.

DRIVE, DRIVEA

On DRIVE the axes accelerate at the maximum rate up to a creep velocity set in a parameter. The acceleration is then reduced to a level set in a machine data. This “knee-bend” acceleration characteristic allows the acceleration curve to be optimally adapted to a given motor characteristic, e.g. for stepper motors.

Example for DRIVE:

N50 DRIVE
N60 G1 X10 Y100 ; Path axes accelerate
; in DRIVE acceleration pattern
N70 DRIVEA(A)
N80 POS[A]=100 FA[A]=1000 ; Positioning axis A accelerates
; in DRIVE acceleration pattern

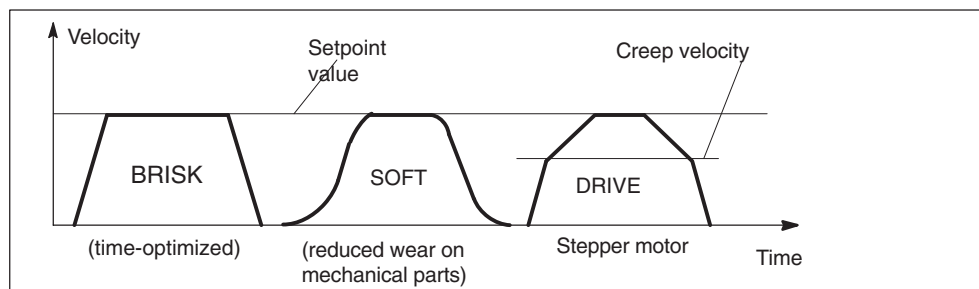


Fig. 10-52 Acceleration characteristic with BRISK / SOFT / DRIVE

Example for BRISK, SOFT and DRIVE:

N10 G1 X100 Y100 G90 G60 G601 F2000 SOFT ; Path axes accelerate in SOFT
; pattern

N20 X30 Y10

N30 BRISKA(A, B) POS[A]=200 POS[B]=300 ; Positioning axes A and B
; accelerate in BRISK pattern

N40 X100 Y-10 ; Path axes continue to
; accelerate in SOFT

A change between BRISK and SOFT causes an internal exact stop at the block transition. The acceleration profile is set in parameters and can only be selected with BRISK or SOFT.

10.6.4 Programmable dynamic limitation

General

The programmable dynamic limitation can be used to change the values for axis acceleration, axis jerk and maximum velocity set via the parameterization.

Programming

```
ACC[machine axis name]=...      ; Programmable acceleration
JERKLIM[Maschinenachsname]=... ; Programmable jerk
VELOLIM[Maschinenachsname]=... ; Programmable maximum velocity
$AC_PATHACC=...                  ; Programmable acceleration for
                                  ; external events
```

ACC, JERKLIM, VELOLIM

The parameterized values for acceleration, jerk and maximum velocity can be changed in a percentage range $> 0\%$ and $\leq 200\%$.

The programmable dynamic limitation acts in all types of interpolation of the modes **AUTOMATIC** and **MDI**.

The statements come into effect immediately and are modally active.

Contrary to JERKLIM and VELOLIM, ACC can also be used in synchronized actions.

With Reset or end of program, the default setting of 100 % automatically comes into effect.

Example:

```
N10 ACC[X]=50                    ; X axis traverses with 50 % of the parameterized
                                  ; axis acceleration
N20 VELOLIM[X]=80                 ; X axis traverses with 80 % of the parameterized
                                  ; maximum velocity
```

\$AC_PATHACC

If the path acceleration has been reduced using ACC, very long braking paths can result, e.g. in the case of external events, such as "Stop" or "Override = 0".

The system variable \$AC_PATHACC can be used to increase the path acceleration in braking or acceleration processes triggered by external events. Values less than the effective path acceleration are ignored. The maximum value is limited to the double of the axis acceleration.

Programming

```
$AC_PATHACC =...                 ; Acceleration m/s2
$AC_PATHACC = 0                   ; Deactivation of the function
```

Supplementary conditions

The function is deactivated with RESET. \$AC_PATHACC in the NC program will create an implied STOPRE (preprocessing stop).

\$AC_PATHACC can also be used in synchronized actions.

10.7 Dwell (G4)

General

The dwell time function allows the program to be stopped for a defined period. The dwell must be programmed in a separate block.

Programming

G4 F... ; Dwell time in seconds

G4

G4 is active non-modally.

The setting of the previously programmed F value is retained.

Example:

```
N10 G1 F2000 X200 Y200 ; Traverse with feed F2000
N20 G4 F2.5 ; Dwell time 2.5 s
N30 X300 Z100 ; Feed F2000 active again
...
```


Programming example

X is the master axis and Y and Z must be assigned to X as slaves. The Y axis is to travel 2.5 times further than X. The Z axis is to travel the same distance as X.

```
...  
TRAILON(Y,X,2.5)      ; Define coupled-axis grouping  
TRAILON(Z,X)          ; Define coupled-axis grouping  
...  
TRAILOF(Y,X)          ; Deactivate coupled-axis grouping  
TRAILOF(Z,X)          ; Deactivate coupled-axis grouping  
...
```

Notice

For a further description of coupled motion, please see Section 9.16.1.

10.9 Tangential control (TANG, TANGON, TANGOF)

General

This function is used to couple a rotary axis to the tangent of a contour created from two geometry axes.

Programming

TANG(slave axis, master axis1, master axis2, coupling factor) ; Definition of the tangential control
TANGON(slave axis, angle) ; Enable tangential control
TANGOF(slave axis) ; Disable tangential control
TLIFT(slave axis) ; Permit intermediate block at contour corners

Define TANG

Specify the slave axis, the two master axes and optionally a coupling factor. The slave axis must be a rotary axis, and the two master axes must be geometry axes.

The coupling factor specifies the ratio of change of the angle of the contour tangent to the change of the contour of the slave axis.

$$\text{Coupling factor} = \frac{\text{Angle of slave axis}}{\text{Angle of contour tangent}}$$

If the coupling factor is not specified, coupling factor 1 will apply by default.

Enable TANGON

Specify the slave axis and optionally an offset angle between contour tangent and slave axis.

A slave axis can be assigned only one tangential control at a time.

Disable TANGOF

Specify the slave axis.

This statement will initiate exact stop at the end of the block and preprocessing stop.

Permit intermediate block at contour corners TLIFT

Specify the slave axis. At contour corners whose angle is greater than the parameter “Intermediate block limit angle”, an intermediate block is inserted for a slave axis motion. The slave axis will traverse to the position corresponding to the contour tangent after the corner as fast as possible.

Reprogramming TANG without TLIFT following will disable this behavior. In this case, the path velocity of the master axes is reduced such that the slave axis will reach its target position synchronously with the two master axes.

Programming example

X and Y are the master axes, and the A axis is to be followed as the rotary axis tangentially at an angle of 30 degrees. Intermediate blocks for the slave axis must be possible at contour corners.

...

```
G0 X0 Y0
POS[A]=0
```

```
N30 TANG(A,X,Y)           ; Define tangential control
N40 TANGON(A,30)          ; Tangential control ON, offset angle 30 degrees
N50 TLIFT(A)              ; Permit intermediate blocks
```

```
; Master axis contour
N70 G1 Y100 F1000
N80 X100
N90 Y0
N100 X0
```

```
TANGOF(A)                 ; Disable tangential control
```

...

Notice

For further notes with regard to the tangential control function, please refer to Section 9.16.5.

10.10 Measurement

Overview

In this section, you can find information about:

- Block-specific measurement (MEAS, MEAW)
- Axial measurement (MEASA, MEAWA, MEAC)

10.10.1 Block-specific measurement (MEAS, MEAW)

General

In block-related measurement mode, the positions of all axes programmed in the block as the probe switches are acquired and stored in system variables. Only one measurement job can be programmed in each block.

Programming

MEAS= ± 1 (± 2)	; Measure and delete distance to go +, –: Probe with positive, negative edge 1, 2: Probe on measurement input 1, 2, non-modal
MEAW= ± 1 (± 2)	; Measure and do not delete distance to go +, –: Probe with positive, negative edge 1, 2: Probe on measurement input 1, 2, non-modal
\$AA_MM[axis]	; Measured value in machine coordinate system
\$AA_MW[axis]	; Measured value in workpiece coordinate system
\$AC_MEA[n]	; Status measurement job, n = number of probe 0: Measurement job conditions not fulfilled (automatically after start of measurement block) 1: Measurement job conditions fulfilled

Measurement is possible in interpolation modes G0, G1, G2 and G3.

After the measurement has been taken, the results are stored in system variables \$AA_MM[axis] and \$AA_MW[axis].

Reading these variables does not initiate an internal preprocessing stop.

In order to evaluate the measurement results immediately after the measurement block, you should program STOPRE first (see Section 10.12).

The accuracy of measurement depends on the approach velocity to the probe.

Measurement with deletion of distance to go MEAS

When this statement is programmed, the axis is braked after a measurement and the distance to go deleted.

Measurement without deletion of distance to go MEAW

With this statement, the axis always traverses up to the programmed end position.

Programming examples

Example 1:

```
...  
N10 MEAS=1 G1 F100 X100 Y730 ; Measurement block, with a positive probe  
; signal edge at measurement input 1, the  
; measurement is taken with deletion of  
; distance to go  
N20 R10=$AA_MM[X] ; Save measured position in R10
```

Example 2:

```
N10 MEAW=2 G1 Y200 F 1000 ; Measure without deleting distance to go  
N20 Y100  
N30 STOPRE  
N40 R10=$AA_MM[X] ; Save measured position  
; after STOPRE
```

10.10.2 Axial measurement (MEASA, MEAWA, MEAC)

General

The function is available for the FM 357-2LX.

Several measurement jobs for different axes can be programmed simultaneously in the same block.

Programming

MEASA[axis]=(mode,TE_1,...,TE_4) ; Axial measurement with deletion of distance to go

MEAWA[axis]=(mode,TE_1,...,TE_4) ; Axial measurement without deletion of distance to go

MEAC[axis]=(mode,memory,TE_1,...,TE_4) ; Axial measuring continuously (SW 5.2 and higher)

MEAC[axis]=(0) ; Cancel axial measuring continuously

Mode: 0 Abort measurement task (use in synchronized actions)
 1 Reserved
 2 Activate measuring job, activate trigger events **one after the other**; the trigger events are evaluated in the given sequence order
 3 Activate measurement task, equivalent to mode 2 with no error message with trigger event 1 active

Memory: Number of the FIFO memory for storing the measured values

TE_1...4: (trigger event 1...4)
 1 rising edge probe 1
 -1 falling edge probe 1
 2 rising edge probe 2
 -2 falling edge probe 2

\$AA_MM1...4[axis] ; Measured value of trigger event 1...4 in machine coordinate system

\$AA_MW1...4[axis] ; Measured value of trigger event 1...4 in workpiece coordinate system

\$A_PROBE[n] ; Probe status, n = number of probe
 0: Probe not deflected
 1: Probe deflected

\$AA_MEAACT[axis] ; Status axial measurement
 0: Measurement task for axis terminated/not active
 1: Measurement task for axis active

\$AC_MEA[n] ; Status measurement job, n = number of probe
 0: All axial meas. job conditions not yet fulfilled
 1: All axial meas. job conditions fulfilled

Axial measurement can be programmed for positioning or geometry axes. A separate measurement job must be programmed for each axis.

Only one trigger event can be acquired in the servo cycle. The interval between two trigger events must therefore be greater than $2 * \text{servo cycle}$.

It is not permissible to program block-specific and axial measurement in the same block.

On completion of a measurement, the measurement results and the associated trigger events are stored in system variables $\$AA_MM1\dots4[\text{axis}]$ and $\$AA_MW1\dots4[\text{axis}]$.

Reading these variables does not initiate an internal preprocessing stop.

In order to evaluate the measurement results immediately after the measurement block, you should program STOPRE first (see Section 10.12).

The accuracy of measurement depends on the approach velocity to the probe.

The status for all the measurement jobs of one probe can be read from $\$AC_MEA[n]$ or, for specific axes, from $\$AA_MEAACT[\text{axis}]$.

Axial measurement with deletion of distance to go MEASA

If all trigger events have occurred, the axis is stopped and the distance to go deleted.

Axial measurement without deletion of distance to go MEAWA

With this statement, the axis always traverses up to the programmed end position.

MEAWA can be started from synchronized actions (only with FM 357-2LX, see Section 10.32).

In this case, variables $\$AA_MW1\dots4[\text{axis}]$ and $\$AC_MEA[n]$ are not available.

Axial measuring continuously MEAC

When measuring continuously (MEAC), the measured values exist only in the machine coordinate system. The measured values are stored in the specified FIFO[n] memory.

If two sensing probes are specified, the measured values of the second probe are stored separately in the FIFO[n+1] memory.

The FIFO memory is a circulating memory. For further information on $\$AC_FIFO$ variables, please refer to Section 10.21.

If the number of measured values exceeds the parameterized FIFO range, the measurement is aborted.

If you wish to measure infinitely, the FIFO memory must be read cyclically, at least with the same frequency as it is measured.

MEAC is modally effective and can be terminated with $MEAC[\text{axis}]=0$.

Note

If axial measuring (MEASA, MEAWA) is to be started for a geometry axis, the same measurement job must be programmed explicitly for all of the remaining geometry axes. This is the only way for the FM to calculate the measured value in the workpiece coordinate system. This also applies to axes involved in a transformation.

If a geometry axis is traversed using the POS[...] statement, No interpolation group exists. The remaining geometry axes need not be programmed in the measuring job.

With MEAC, a measuring job is only possible for one geometry axis, since the measured value is determined only in the machine coordinate system.

Example:

```
...
N10 MEASA[Z]=(2,1) MEASA[Y]=(2,1) MEASA[X]=(2,1) G0 Z100 F 1000 ;
...
```

```
...
```

or

```
...
N10 MEASA[Z]=(2,1) POS[Z]=100 FA[Z]=1000 ;
...
```

Hot-spot measurement

A “hot-spot” can be defined for the detection of the probe signal edge (see Section 9.11.5).

Programming example 1

```
...
N10 MEASA[X]=(2,1,-1) POS[X]=100 FA[X]=900 ; Measurement job for X axis:
; trigger events sequentially task
; TE1: Positive edge
; TE2: Negative edge
; Probe 1
; Abort motion and delete
; distance to go
N20 STOPRE ; Preprocessing stop for synchronization
N30 IF $AC_MEA[1]==0 gotof ERROR ; Check measurement
N40 R10=$AA_MM1[X] ; Store measuring position 1
N50 R11=$AA_MM2[X] ; Store measuring position 2
...
ERROR: -
```

Programming example 2

The mean spacing between several measuring points is to be determined when traversing the X axis. A FIFO memory range starting from R10 is parameterized. The number of the FIFO elements will certainly be enough to accommodate the measured values determined.

```

...
N05 $AC_FIFO1[4]=0                ; Clear FIFO

N10 MEAC[X]=(2,1,1) G1 G64 X100 F100 ; Axial measuring continuously:
                                     ; FIFO memory 1
                                     ; TE1: positive edge probe 1
N15 X200                             ; MEAC is still active
N20 MEAC[X]=0                         ; Complete measuring job
N25 STOPRE                             ; Stop block preprocessing until
                                     ; measuring is completed

; Calculate total of measuring point spacings:
N30 IF $AC_FIFO1[4] < 2 GOTOF MEA_ERROR
; At least 2 measuring points were acquired.
N35 R2=1                               ; Run variable
N40 R3=0                               ; Total of the measuring point spacings
N45 WHILE R2 < $AC_FIFO1[4]           ; Number of measuring points:
                                     ; $AC_FIFO1[4]
N50 R3 = R3 + $AC_FIFO1[6+R2] - $AC_FIFO1[6+R2-1]
N55 R2=R2+1
N60 ENDWHILE

; Calculate mean value of measuring point spacings
N65 R0 = R3 / $AC_FIFO1[4]
N70 M30

MEA_ERROR:
N75 M30 ; Error in test series

```

10.11 Travel to fixed stop (FXST, FXSW, FXS)

General

The “Travel to fixed stop” function is used to produce defined forces for the purpose of clamping parts.

When the fixed stop has been reached, the system switches from position control to current control or torque control mode. The operation sequence and signal interplay with the CPU are described in Section 9.18.

The function is available for the FM 357-2LX.

Programming

FXS[axis]=... ; Select/deselect travel to fixed stop
FXST[axis]=... ; Clamping torque
FXSW[axis]=... ; Monitoring window

The fixed stop statements are modal. If no specific statement is programmed, the last programmed value or the parameterized setting is applied.

The function is programmed with machine axes (X1, Y1, Z1 etc.)

FXS

Activate travel to fixed stop FXS=1

The motion to the target point can be described as a path or positioning axis motion. For positioning axes of type POSA, the function remains active beyond block limits.

Travel to fixed stop can be programmed simultaneously for several axes and in parallel to the motion of other axes. The fixed stop must lie between the start and target positions.

Notice

As soon as the “Travel to fixed stop” function has been activated, no new position for the axis in question may be programmed in subsequent NC blocks.

“Measurement with deletion of distance to go” (“MEAS” statement) and “Travel to fixed stop” must not be programmed simultaneously for the same axis in one block.

Deactivate travel to fixed stop FXS=0

A motion that leads away from the fixed stop must be programmed in the deselection block. An internal preprocessing stop is initiated for the purpose of position synchronization.

FXST, FXSW

The clamping torque (FXST) is specified as a percentage of the maximum drive torque. FXST is effective from the block beginning, i.e. even the fixed stop is approached at reduced torque.

The monitoring window (FXSW) is specified in mm or degrees. The window must be programmed such that any impermissible yield in the fixed stop will trigger the monitoring window alarm.

FXST and FXSW can be altered in the NC program at any time. Changes will take effect prior to any traversing motions that are programmed in the same block.

Programming example

```
N10 G0 X0 Y0
N1 X250 Y100 F100 FXS[X1]=1 FXST[X1]=12.3 FXSW[X1]=2
    ; Axis X1 is moved at feed F100 to target position X = 250 mm.
    ; Travel to fixed stop is activated. The clamping torque corresponds
    ; to 12.3 % of the maximum drive torque, the monitoring window is
    ; 2 mm wide
...
N20 X200 Y400 G01 F2000 FXS[X1]=0 ; Axis X1 is retracted from the fixed
    ; stop to position X = 200 mm.
```

Status interrogation in NC program

System variable \$AA_FXS[...] indicates the status of the "Travel to fixed stop" function. Its coding is as follows:

\$AA_FXS[...]= 0	Axis is not at fixed stop
1	Fixed stop has been approached successfully (axis is inside monitoring window)
2	Approach to fixed stop has failed (axis is not positioned at stop)

Interrogation of the system variable in the NC program initiates a preprocessing stop.

Through a status interrogation in the NC program, for example, it is possible to react to an error in the "Travel to fixed stop" sequence.

Example:

The following applies in this example:

Parameter "Error message" = no → an error is not generated, so a block change takes place and the status can be evaluated via the relevant system variable.

```
N1 X300 Y500 F200 FXS[X1]=1 FXST[X1]=25 FXSW[X1]=5
N2 IF $AA_FXS[X1]==2 GOTOF FXS_ERROR
N3 G1 X400 Y200
```

10.12 Stop preprocessor (STOPRE)

General

The control prepares the blocks of an NC program via a preprocessing buffer. The block preparation thus anticipates block execution. Block preparation and block execution are synchronized.

When you program STOPRE, preparation of the NC blocks in the preprocessing buffer is suspended, while block execution continues.

A block is not prepared until the previous block has been executed completely.

Programming

STOPRE ; Stop preprocessor

STOPRE

STOPRE must be programmed in a separate block.

An exact stop is forced in the block before STOPRE.

Internally, STOPRE is generated by accessing system variables (\$A...).

Exception: System variable for measurement results.

10.13 Working area limitations (G25, G26, WALIMON, WALIMOF)

General

Statement G25/G26 can be programmed to limit the area in which the axes operate. You can thus set up no-go areas for the axes. The working area limitation is only active when the reference point has been approached and the function has been activated in the parameters.

You program a lower limit (with G25) and an upper limit (with G26) in the machine coordinate system. These values are valid immediately and are retained even after a RESET or power-up.

Programming

G25 X... Y... Z...	; Minimum working area limitation, MCS
G26 X... Y... Z...	; Maximum working area limitation, MCS
WALIMON	; Switch on working area limitation
WALIMOF	; Switch off working area limitation

Minimum working area limitation G25

The position assigned here to an axis represents its minimum working area limitation (n).

G25 must be programmed in a separate block.

Maximum working area limitation G26

The position assigned here to an axis represents its maximum working area limitation (n).

G26 must be programmed in a separate block.

WALIMON

Statement WALIMON activates the working area limitation for all axes programmed with G25/G26.

WALIMON is the initial setting.

WALIMOF

Statement WALIMOF deactivates the working area limitation for all axes programmed with G25/G26.

Programming example

```
G25 X45 Y40
G26 X220 Y100
```

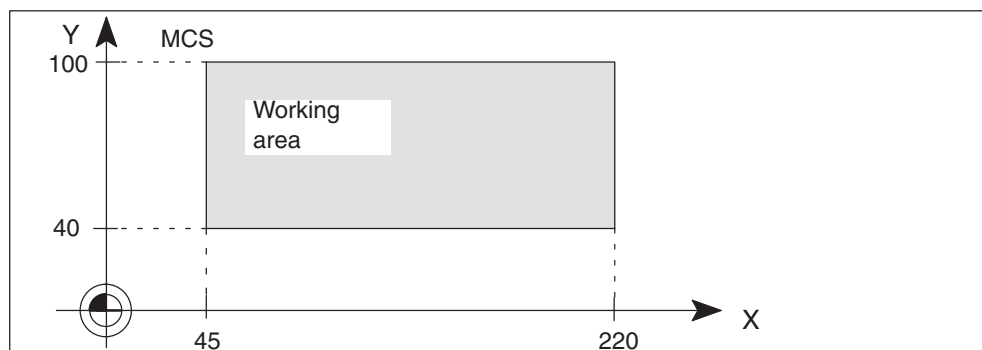


Fig. 10-53 Working area limitations G25 and G26

10.14 M functions

General

M functions can be used, for example, to initiate switching operations from the NC program for a variety of functions in the CPU. Some of the M functions have fixed functionality hard-coded by the controller manufacturer. The remaining functionality is available for programming by the user.

A maximum of five M functions can be programmed in a block.

Value range of the M functions: 0...99

Programming

M... ; M function

Output option for T functions

M functions can be output to the CPU at the following times:

- before the movement
- during the movement
- after the movement

During parameterization you can assign an output option to the available M functions.

You will find further information about the output options for M functions in Section 9.9.

Effect

The effect of M functions in blocks with traversing motions depends on the selected output option for the M function:

Functions output before the traversing movements interrupt continuous-path mode (G64, G641) and generate an exact stop for the previous block. Functions output after the traversing movements interrupt continuous-path mode (G64, G641) and generate an exact stop for this block.

Predefined M functions:

M No.	M Function	Output option
0	Stop at end of block	After the traversing movement
1	Conditional stop	
2, 30	End of program	
17	Disabled	–
3, 4, 5, 70	Disabled	–
6, 40...45	Disabled	–

Programming example

Assumption:

Output of free M functions after the movement.

N10 ...

N20 G0 X1000 M80 ; M80 is output when X1000 is reached

...

10.15 H functions

General

H functions can be output to initiate switching functions on the machine or transfer data from the NC program to the UP.

A maximum of three H functions can be programmed in a block.

Value range of the H functions: 0...99

Programming

H... ; H function

Output option for T functions

H functions can be output to the CPU at the following times:

- before the movement
- during the movement
- after the movement

During parameterization an output option is assigned to the H functions.

You will find further information about the output options for H functions in Section 9.9.

Effect

The effect of H functions in blocks with traversing motions depends on the selected output option for the H function:

- Functions output before the traversing movements interrupt continuous-path mode (G64, G641) and generate an exact stop for the previous block.
- Functions output after the traversing movements interrupt continuous-path mode (G64, G641) and generate an exact stop for this block.

Value transfer

It is possible to transfer a value to the user program (UP) in addition to the number of the H function.

Value range $\pm 99\,999.9999$ precision 0.0001

Example:

N10 H10=123.4567 ; H function 10 transfers value 123.4567 to the UP

N.B. for abbreviated notation: H5 is equivalent to H0=5

10.16 Tool offset values (T functions)

General

T functions can be used to initiate switching operations in the CPU for the purpose of supplying the tool specified in the T number. The associated tool offsets stored on the FM are also activated. A prerequisite is that a corresponding tool has been created using the parameterization tool.

One T function can be programmed in a block.

Value range of the T functions: 0...29

Programming

T1...T29 ; Select tool T1...T29 and tool offset
T0 ; Deselect tool and tool offset

Tool length compensation

Three length compensation values are assigned to each tool. These act as additional offsets in the WCS.

The tool offsets are included in the next traversing movement of the axis. The traversing movement must be a linear interpolation (G0, G1).

The axes on which the tool length compensation is calculated depend on the plane and the assignment of the machine axis to the geometry axes.

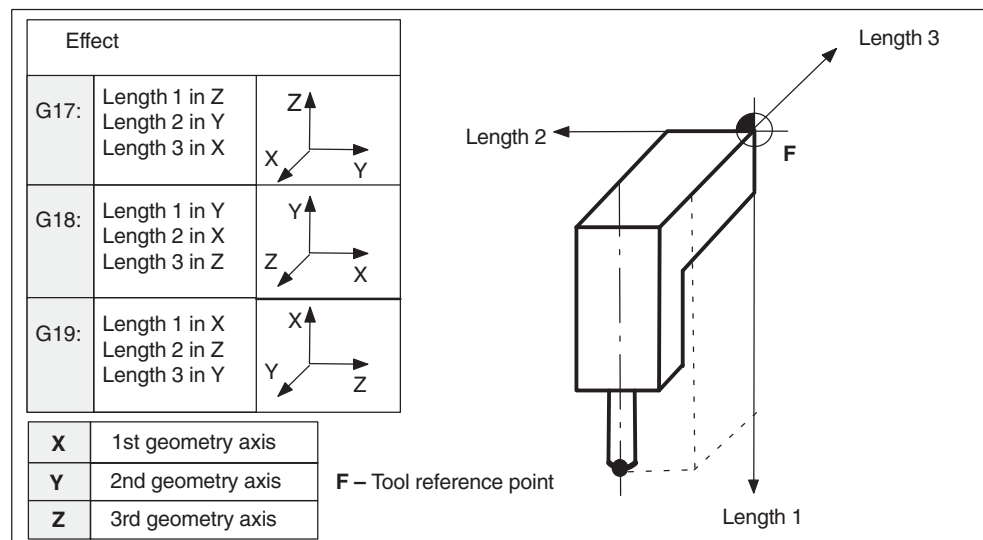


Fig. 10-54 Three-dimensional effect of tool length compensation

Output option for T functions

T functions are output to the CPU prior to any motion.

You will find further information about the output options for T functions in Section 9.9.

Example: Effect of the tool offsets in the G17 plane

X axis = 1st geometry axis

Y axis = 2nd geometry axis

Length 2 = 10 Zero offset G54 X=20

Length 3 = 10 Zero offset G54 Y=15

N05 G53 G0 X0 Y0 G17

N10 G54 G0 X0 Y0 ; Traverse through G54 shift

N15 T1 ; T1 is selected

N20 G0 X15 Y10 ; Traversal of axis with offset applied

; MCS: X20 to X45 WCS: X0 to X15

; Y15 to Y35 Y0 to Y10

; Traversing path:

; X 25 mm

; Y 20 mm

N25 T0 ; Deselect T1

N30 G0 X50 ; Traversal of X axis without application of offset

...

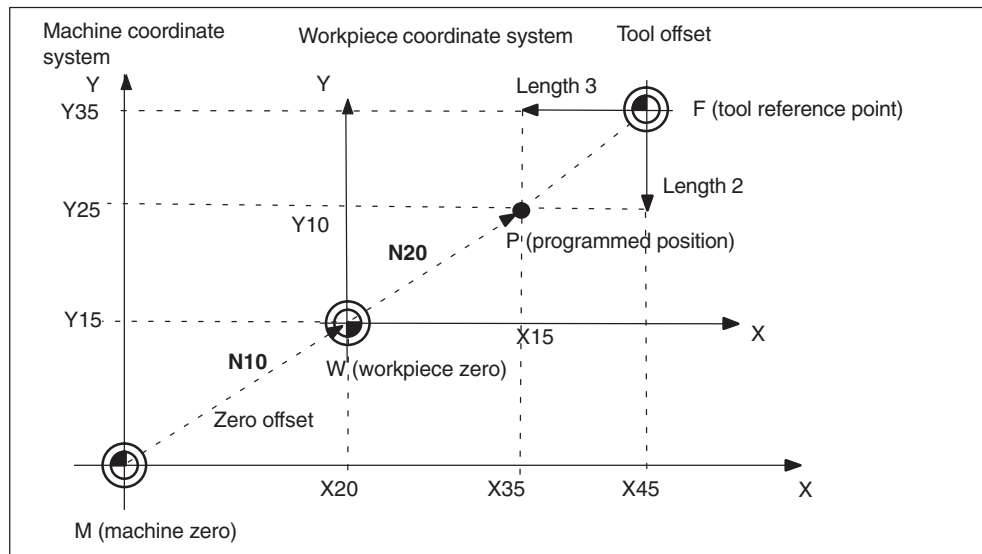


Fig. 10-55 Effect of tool offset and zero offset in the G17 plane

10.17 Protection zones (NPROTDEF, EXECUTE, NPROT)

General

Protection zones can protect the fixed and moving parts of a plant. Up to four protection zones can be defined as a two or three-dimensional contour. The contour is defined in the NC program. Protection zones are not monitored until the axes have been referenced.

Programming

NPROTDEF(n,t,lim,plus,minus) ; Start of contour definition
 EXECUTE(TEMP) ; End of contour definition

Parameters:

n	Number of the protection zone (1, 2, 3 or 4)
t	TRUE = tool-related protection zone FALSE = workpiece-related protection zone
lim	Type of limitation in the 3rd dimension 0: No limitation 1: Limitation in plus direction 2: Limitation in minus direction 3: Limitation in plus and minus direction
plus	Value of limitation in plus direction of 3rd dimension
minus	Value of limitation in minus direction of 3rd dimension

NPROT(n,state, XV,YV,ZV) ; Activate, deactivate
 ; Protection zone

Parameters:

n	Number of the protection zone (1, 2, 3, or 4)
state	State parameter 0: Deactivate 1: Preactivate 2: Activate

XV,YV,ZV offset of protection zone in axes X, Y, Z

NPROTDEF

The contour definition for a protection zone begins with this statement. The contour is specified as a traversing movement.

Permissible contour elements:

- G0, G1 for straight contour elements
- G2 for clockwise arc sections (only for workpiece-related protection zones)
- G3 for counterclockwise arc sections

Contour definition:

Up to 4 contour elements and thus 5 contour points are possible. The contour must be closed, i.e. the last point must coincide with the first point.

The area to the **left** of the contour is valid as the protection zone. The contour for an internal protection zone must therefore be programmed counterclockwise. The contour definition is **not** executed as a traversing movement.

If the protection zone is to be a complete circle, it must be subdivided into two circle segments. A sequence of G2 after G3 and vice-versa is illegal; a short G1 block must be inserted in this case.

You can extend the level contour into the 3rd dimension using the parameters lim, plus and minus.

The following statements are illegal during the definition of protection zones:

Zero offset, STOPRE, M0, M1, M2, G4 .

Reference point of contour definition:

The workpiece-related protection zones are defined in the WCS and are fixed. The tool-related protection zones must refer to the tool reference point F (see Section 10.16). These protection zones move with the reference point F.

Plane:

The desired plane is selected before NPROTDEF with G17, G18, G19 and may not be changed before EXECUTE. It is not permitted to program the axis perpendicular to the plane (applicate) in the contour definition. Workpiece and tool-related protection zones must have the same plane.

EXECUTE(TEMP)

This statement terminates the contour definition.

The TEMP variable is used for temporary storage of values. You must define this variable at the start of the program with the statement:

```
DEF INT TEMP
```

NPROT

You use this statement to activate, preactivate or deactivate a previously defined protection zone.

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

If no workpiece-related protection zone is active, no protection zone monitoring takes place.

Specifying the state:

- 2: Activate

This generally activates a protection zone in the NC program.

- 1: Preactivate

If you want to activate a protection zone from the user program, the required preactivation must be entered with Status = 1 in the NC program.

- 0: Deactivate

You can only deactivate a protection zone from the NC program.

Protection zones can also be activated automatically after power-up followed by reference point approach.

You must enable the system variable `$SN_PA_ACTIV_IMMED[n] = TRUE` for this purpose. The n parameter is the number of the protection zone. In this case, it is not possible to specify the offset.

The contour definition for the protection zones remains stored modally on the FM 357-2.

Shifting protection zones:

When you activate or preactivate protection zones, you can also specify an offset in the three coordinates.

The offset refers to:

- the machine zero for workpiece-related protection zones
- the tool reference point for tool-related protection zones

Programming example

The collision of axes with the pallets stored outside the defined area (external protection zone) is to be programmed on a machine.

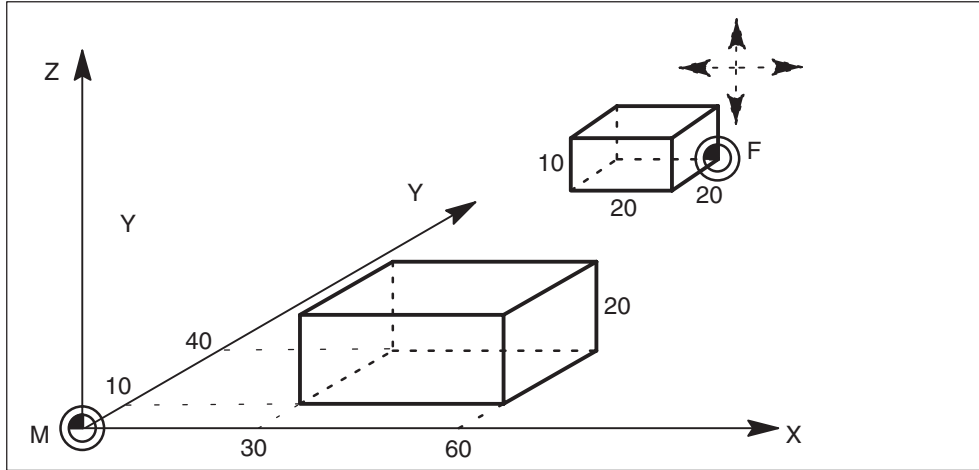


Fig. 10-56 Example of workpiece and tool-related internal protection zones

```

N05 DEF INT TEMP           ; Define temporary variable
N10 G17                   ; Define plane
N15 NPROTDEF(1,FALSE,3,20,0) ; Define workpiece-related
                             ; Protection zone 1
N20 G01 X30 Y10          ; Define contour
N25 X60                   ; The area to the left of the contour is valid
                             ; as the protection zone
N30 Y40
N35 X30
N40 Y10
N45 EXECUTE(TEMP)        ; End of definition
N15 NPROTDEF(2,TRUE,3,10,0) ; Define tool-related
                             ; protection zone 2
N20 G01 X0 Y0           ; Define contour
N25 X20                  ; The area to the left of the contour is valid
                             ; as the protection zone
N30 Y20
N35 X0
N40 Y0
N45 EXECUTE(TEMP)        ; End of definition
...
N100 NPROT(1,2,0,0,0)    ; Activate protection zone 1
N100 NPROT(2,2,0,0,0)    ; Activate protection zone 2
...

```

10.18 Fundamentals of variable NC programming

General

Using the variable NC programming, you can create your own NC programs with little programming expenditure and in a more flexible fashion.

The following scope of functions is available:

- Different variable types; variable definition
- Arithmetic and logic operators, relation operators, arithmetic functions
- Type conversions
- Indirect programming
- Program branches, control structures
- String operations
- Reading, writing and deleting a file

Variable types

Generally, a distinction is made between the following three variable types.

Table 10-1 Variable types

Variable	Meaning
Arithmetic parameters	Predefined arithmetic variable of the data type REAL
System variable	Variable provided by the control system Provides read and partially write access to internal data, e.g. zero offsets, actual values, measured values of the axes etc.
User variable	Variable that can be defined by name and type by the user and which can be used freely by the him

Data types

The following data types are supported. Arithmetic parameters and system variables is assigned a fixed data type.

Table 10-2 Data types for variables

Data Type	Meaning	Range of Values
INT	Integer signed values	$\pm (2^{31}-1)$
REAL	Real numbers (fractional numbers with decimal point, LONG REAL acc. to IEEE)	$\pm(10^{-300} \dots 10^{+300})$
BOOL	Truth values: TRUE (1) and FALSE (0)	1, 0
CHAR	1 ASCII character as per code	0...255
STRING	String, number of characters in [...], max. 200 characters	Sequence of values 0...255
AXIS	Only axis names (axis addresses) for indirect programming in square brackets	All existing axes in the channel

Operators/arithmetic functions

Depending on the data type, the following operators and arithmetic functions can be used.

Table 10-3 Operators and arithmetic functions

	Meaning
Arithmetic operators	
=	Assignment
+	Addition
-	Subtraction
*	Multiplication
/	Division (INT type)/(INT type) = (REAL type) Example: 3/4 = 0.75
DIV	Division (INT or REAL) (INT type)/(INT type) = (INT type) Example: 3 DIV 4 = 0
MOD	Modulo division (INT or REAL) provides the rest of an INT division Example: 3 MOD 4 = 3
Logical operators	
NOT	NOT, negation
AND	AND
OR	OR

Table 10-3 Operators and arithmetic functions, continued

	Meaning
XOR	Exclusive OR
Bit-serial logical operators	
B_NOT	Negated by bits
B_AND	Bit-serial AND
B_OR	Bit-serial OR
B_XOR	Exclusive bit-serial OR
Relation operators	
==	Equal to
<>	Unequal to
>	Greater than ¹⁾
<	Less than ¹⁾
>=	Greater than or equal to ¹⁾
<=	Less than or equal to ¹⁾
Arithmetic functions	
SIN()	Sine
COS()	Cosine
TAN()	Tangens
ASIN()	Arc sine
ACOS()	Arc cosine
ATAN2()	Arc tangens2
SQRT()	Square root
POT()	Square
ABS()	Absolute value
TRUNC()	Integer part
ROUND()	Rounding to an integer value
LN()	Natural logarithm
EXP()	Exponential function

1) These relation operators are not possible for the data types STRING and AXIS.

Value assignment

Values, i.e. constants, variables or expressions of an appropriate type (see type conversion), addresses or variables can be assigned in the NC program. The assignment requires a separate block. Several assignments per block are possible. Assignment to axis addresses require a separate block, unlike variable assignments.

Example:

```
N05 R10 =100 R11=200      ; Value assignment for constants
N10 G1 X=R10 F=R11       ; Value assignment for variable
N20 R10=100+R11+SIN(20)  ; Value assignment for expressions
...
N30 G1 X=R10 R10=50      ; Wrong!
                          ; Variable assignment requires a separate block
```

Type conversion

The value assigned to a variable as a constant, variable or expression must be of the same data type as the variable or an automatically (implicit) type conversion must be possible.

Table 10-4 Possible type conversion

from to	REAL	INT	BOOL	CHAR	STRING	AXIS
REAL	yes	yes ¹⁾	yes ²⁾	yes ¹⁾	–	–
INT	yes	yes	yes ²⁾	yes ³⁾	–	–
BOOL	yes	yes	yes	yes	yes	–
CHAR	yes	yes	yes ²⁾	yes	yes	–
STRING	–	–	yes ⁵⁾	yes ⁴⁾	yes	–
AXIS	–	–	–	–	–	yes

- 1) With type conversion from REAL to INT, the result is rounded up ≥ 0.5 in the case of fractional numbers; otherwise, the result is rounded off
- 2) Value $\neq 0$ corresponds to TRUE, value $= 0$ corresponds to FALSE
- 3) If the value is in the permissible range of numbers
- 4) If only one character
- 5) String length 0 = FALSE, otherwise TRUE

Notice

If a value is greater than the range of numbers when converting, an error message is provided.

If mixed types occur in an expression, an automatic (implicit) type adaptation is carried out if possible.

In addition to the implicit type conversion, there still are various type conversion statements; see e.g. string operations.

Priorities of the operators and execution

Each operator is assigned a priority. When evaluating an expression, the operators of the higher priority are always used first. Operators of the same priority are evaluated from the left to the right.

The order of execution of all operators can be defined in arithmetic expressions by using round brackets, bypassing the priority rules.

Table 10-5 Priorities of the operators

Priority	Operator	Meaning
1.	NOT, B_NOT	Negation, bit-serial negation
2.	*, /, DIV, MOD	Multiplication, division
3.	+, -	Addition, subtraction
4.	B_AND	Bit-serial AND
5.	B_XOR	Bit-serial exclusive OR
6.	B_OR	Bit-serial OR
7.	AND	AND
8.	XOR	Exclusive OR
9.	OR	OR
10.	<<	Linking of strings, result type STRING
11.	==, <>, >, <, >=, <=	Relation operators

Indirect programming

Access to a variable can be carried out via an index. The index itself, in turn, can be a variable or an expression.

Indirect programming is not possible for the following addresses:

- N block number
- G G command
- L subroutine

Example:

```
N10 R10=7 ; Direct programming
; indirect programming
R[R10]=9 ; Parameter R7 is assigned the value 9
R[R10+2]=R[R10] ; Parameter R9 is assigned the value from R7
```

Multiple-nested indexing is also possible.

Example:

```
N10 R1=10 R10=20 R20=30 R30=40 R40=12345 R41=0
N20 R[R[R[R[R1]]]+1] = R[R[R[R[R1]]]] ; Parameter R41 is assigned the
; value from R40
```

Program jumps, control structures

The following statements can be used to create a variable program execution.

Table 10-6 Program jumps and control structures

Statement	Meaning
Program jumps	
GOTOF label	Unconditional jump forward (label corresponding to jump destination)
GOTOB label	Unconditional jump reverse
IF condition GOTOF label	Condition jump forward
IF condition GOTOB label	Conditional jump reverse
Control structures	
IF ELSE ENDIF / IF ENDIF	Selection between 2 / 1 alternatives
LOOP ENDLOOP	Endless loop
FOR ENDFOR	Meter loop
WHILE ENDWHILE	Loop with condition in the beginning of the loop
REPEAT UNTIL	Loop with condition at the end of the loop
CASE	Case differentiation

10.19 R parameters (arithmetic parameters)

General

Arithmetic variables of the REAL type are available in address R. These parameters can be used in the NC program to calculate values and assign other addresses, etc. No further statements, e.g. traversing statements, may be programmed in an arithmetic block.

Programming

R0=...
to
R99=... ; 100 arithmetic parameters (default value) are available

Value assignment

Values in the following range can be assigned to arithmetic parameters:

0 ... $\pm 9999\ 9999$ (8 decimal places and sign and decimal point).
Precision: 0.000 0001

Example:

R0=3.5678 R1=-23.6 R2=-6.77 R4=-43210.1234

Exponential notation: $\pm 10^{-300} \dots 10^{+300}$ (extended numeric range)

Example:

R0=-0.1EX-7 ; means: R0 = -0.000 00001

R1=1.874EX8 ; means: R1 = 187 400 000

Multiple assignments can be programmed in the same block, including the assignment of arithmetic expressions.

Address assignment

You can assign addresses to arithmetic parameters or to arithmetic expressions with arithmetic parameters. This applies to all addresses except for **N**, **G** and **L**.

In the assignment, you write the address character after the "=" symbol. You can program an assignment with a negative sign.

Address assignments can be programmed with other statements in the block, but not in arithmetic blocks.

Example:

N5 R2=100 ; R2 is initialized with a value of 100

N10 G0 X=R2 ; Assign to X axis, X axis travels to 100.

N15 G0 Y=R7+R8 ; Calculate and assign

N20 R8=10+R7 ; An address assignment cannot be programmed
; here

Arithmetic operations and functions

The usual mathematical notation must be applied when operators/functions are used. Priorities for execution are indicated by round brackets. The **multiplication and division before addition and subtraction** calculation rule otherwise applies.

Example:

N10 R1= R1+1 ; The new R1 is calculated from the old R1 plus 1
 N15 R1=R2+R3 R4=R5-R6 R7=R8·R9 R10=R11/R12

N20 R14=R1·R2+R3 ; Multiplication and division have priority
 ; over addition and subtraction
 ; R14 = (R1 · R2)+R3

N14 R14=R3+R2·R1 ; R14 = R3+(R2 · R1)

Operators/arithmetic functions

Example:

N10 R13=SIN(25.3) ; sin 25.3°

N15 R15=SQRT(POT(R1)+POT(R2)) ; internal brackets are eliminated first

Meaning: $R15 = \sqrt{R1^2 + R2^2}$ L_F

Comparison operations

The result of comparison operations can either be assigned as a value or serve to formulate a jump condition. Complex expressions are also comparable.

The result of comparison operations is always of the BOOL type.

Example:

R2=R1>1 ; R2=TRUE if R1 > 1

R1<R2+R3

R6==SIN(POT(R7)) ; Is R6 = SIN (R7)² ?

10.20 System variables (\$P_, \$A_, \$AC_, \$AA_)

General

The control system makes system variables available in all running programs and program levels, e.g. in comparison or arithmetic operations.

System variables are identified specifically by a \$ sign as the first character in the name.

Programming

1st letter

\$P_ ; **Programmed** data
\$A_, \$AC_ ; **Current** general data
\$AA_ ; **Current axis-specific** data

2nd letter

C_ ; **Channel-specific** data
A_ ; **Axis-specific** data
no letter ; **General** data

Example:

N10 R10 = \$AA_IW[X] ; Save actual position of X axis in R10
N15 \$A_OUT[3]=R10 > 100 ; Enable digital output 3 if
; R10 is greater than 100

Preprocessing stop

Since the control prepares the NC blocks in advance in a preprocessing buffer, block preparation is ahead of block execution. When reading and writing **current** system variables, internal synchronization of block preparation and block execution must take place. A preprocessor stop is generated for this purpose (see Section 10.12). An exact stop is forced, and the following block is not prepared until the previous blocks have been executed.

System variable

The following table provides an overview of all available system variables.

Table 10-7 System variables

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
User variables				
\$Rn	Arithmetic parameter in static memory	r / w	r / w	REAL
\$AC_MARKER[n] n = 0...7	Marker variable, counter	r / w	r / w	INT
\$AC_PARAM[n] n = 0...49	Arithmetic parameter in dynamic memory	r / w	r / w	REAL
Digital inputs/outputs				
\$A_IN[n]	Digital input	r	r	BOOL
\$A_OUT[n]	Digital output	r / w	r / w	BOOL
\$A_INA[n]	Analog input	r	r	REAL
\$A_OUTA[n]	Analog output	r / w	r / w	REAL
Timers				
\$A_YEAR	Current system time year	r	r	INT
\$A_MONTH	Current system time month	r	r	INT
\$A_DAY	Current system time day	r	r	INT
\$A_HOUR	Current system time hour	r	r	INT
\$A_MINUTE	Current system time minute	r	r	INT
\$A_SECOND	Current system time second	r	r	INT
\$A_MSECOND	Current system time millisecond	r	r	INT
\$AC_TIME	Time from block start in seconds	r	r	REAL
\$AC_TIMEC	Time from block start in IPO cycles	r	r	REAL
\$SAC_TIMER[n]	Timer variable; n number of the timer variables (1...32)	r/w	r/w	REAL
Measurement				
\$AA_MEAAct[axis]	Status of axial measurement 0: Measurement task for axis terminated/not active 1: Measurement task for axis active	r	r	BOOL
\$AC_MEA[n] n: Probe 1 or 2	Status of measurement job (MEAS, MEAW) 0: Meas. job conditions not fulfilled 1: Measurement job conditions fulfilled	r		INT

r = read, w = write

Table 10-7 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$A_PROBE[n] n: Probe 1 or 2	Probe status 0: Probe not deflected 1: Probe deflected	r	r	BOOL
\$AA_MM[axis]	Measured value in MCS with MEAS	r	r	REAL
\$AA_MW[axis]	Measured value in WCS with MEAS	r	r	REAL
\$AA_MMi[axis]	Measured value in MCS with MEASA i: Trigger event 1...4	r	r	REAL
\$AA_MWi[axis]	Measured value in WCS with MEASA i: Trigger event 1...4	r	r	REAL
\$VA_IM[axis]	Measured encoder actual value in MCS	r	r	REAL
\$AA_ENC_ACTIVE[axis]	Validity of encoder actual values	r	r	BOOL
Travel to fixed stop				
\$AA_FXS[axis]	Status of travel to fixed stop 0: Axis is not at stop 1: Fixed stop has been approached successfully (axis is inside monitoring window) 2: Approach to fixed stop has failed (axis is not positioned at stop)	r	r	INT
Path distances				
\$AC_PATHN	Normalized path parameter (0: start of block, 1: end of block)		r	REAL
\$AC_PLTBB	Path distance from start of block in the MCS		r	REAL
\$AC_PLTEB	Path to end of block in the MCS		r	REAL
\$AC_DTBW	Distance from start of block in the WCS		r	REAL
\$AC_DTBB	Distance from start of block in the MCS		r	REAL
\$AC_DTEW	Distance from end of block in the WCS		r	REAL
\$AC_DTEB	Distance from end of block in the MCS		r	REAL
\$AC_DELT	Delete distance to go for path after DELDTG in WCS	r	r	REAL
Axial paths (valid for positioning and synchronous axes)				
\$AA_DTBW[axis]	Axial path from block start in WCS		r	REAL
\$AA_DTBB[axis]	Axial path from block start in MCS		r	REAL
\$AA_DTEB[axis]	Axial path to end of motion in MCS		r	REAL

r = read, w = write

Table 10-7 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AA_DTEW[axis]	Axial path to end of motion in WCS		r	REAL
\$AA_DELT[axis]	Axial distance to go after DELDTG in WCS	r	r	REAL
Positions				
\$AA_IW[axis]	Actual position of axis in the WCS	r	r	REAL
\$AA_IM[axis]	Actual position of axis in the MCS (interpolator setpoints)	r	r	REAL
Software end position				
\$AA_SOFTENDP[X]	Software end position, positive direction	r		REAL
\$AA_SOFTENDN[X]	Software end position, negative direction	r		REAL
Oscillation				
\$SA_OSCILL_REVERSE_POS1[axis]	Position of reversal point 1	r	r	REAL
\$SA_OSCILL_REVERSE_POS2[axis]	Position of reversal point 2	r	r	REAL
Path velocities				
\$AC_VACTB	Path velocity in MCS		r	REAL
\$AC_VACTW	Path velocity in WCS		r	REAL
\$AC_VC	Additive path feed override (must be rewritten in every IPO cycle or the value is set to 0)		r / w	REAL
\$AC_OVR	Path override factor (must be rewritten in every IPO cycle or the value is set to 100 %)		r / w	REAL
Axial velocities (valid for positioning axes)				
\$AA_VACTB[axis]	Axis velocity, setpoint in MCS		r	REAL
\$AA_VACTW[axis]	Axis velocity, setpoint in WCS		r	REAL
\$VA_VACTM[axis]	Axis velocity, actual value in MCS		r	REAL
\$AA_VC[axis]	Additive axial feedrate override		r / w	REAL
\$AA_OVR[axis]	Axial override factor (must be rewritten in every IPO cycle or value is set to 100 %)		r / w	REAL
Acceleration				
\$AC_PATHACC	Programmable acceleration for external events	r / w	r / w	REAL

r = read, w = write

Table 10-7 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
Master value coupling				
\$AA_LEAD_SP[axis]	Position of simulated master value	r	r / w	REAL
\$AA_LEAD_SV[axis]	Velocity of simulated master value	r	r / w	REAL
\$AA_LEAD_P[axis]	Position of real master value (corresponds to encoder actual value)	r	r	REAL
\$AA_LEAD_V[axis]	Velocity of real master value	r	r	REAL
\$AA_LEAD_P_TURN[axis]	Modulo position	r	r	REAL
\$AA_SYNC[axis]	Coupling status of slave axis 0: Not synchronized 1: Synchronism coarse 2: Synchronism fine 3: Synchronism coarse and fine	r	r	INT
\$AA_IN_SYNC[axis]	Synchronization process of the following axis 0: No synchronization 1: Synchronization running	r	r	INT
\$AA_COUP_ACT[axis]	Type of axis coupling of slave axis 0: Not coupled 3: Axis is corrected tangentially 4: res. 8: Slave axis 16: Master axis	r	r	INT
\$SA_LEAD_OFFSET_IN_POS[FA]	Master axis position offset	r / w		REAL
\$SA_LEAD_SCALE_IN_POS[FA]	Scaling for master axis position	r / w		REAL
\$SA_LEAD_OFFSET_OUT_POS[FA]	Slave axis position offset	r / w		REAL
\$SA_LEAD_SCALE_OUT_POS[FA]	Scaling for slave axis position	r / w		REAL
\$P_CTABDEF	Program section for curve table definition 0: No curve table definition 1: Curve table definition	r		BOOL
Overlaid motion				
\$AA_OFF[axis]	Overlaid motion		r / w	REAL
\$AA_OFF_VAL[axis]	Total of pathes involved in the overlaid motion	r	r	REAL

r = read, w = write

Table 10-7 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AA_OFF_LIMIT[axis]	Limit for overlaid movement 0: Not reached 1: Reached in positive direction 2: Reached in negative direction	r	r	INT
CPU variables (user DB "FMx", DBW22 and DBW34)				
\$A_DBW[0] \$A_DBW[2]	Data word from CPU, FM can read Data word to CPU, FM can write (freely programmable by user)		r w	INT
Trace				
\$AA_SCTRACE[axis]	Generation of an IPO event (trigger event)	r / w	r / w	BOOL
States				
\$AC_STAT	Current FM status 0: reset 1: interrupted 2: Active		r	INT
\$AC_PROG	Current status of NC program 0: Program reset/aborted 1: Program stopped 2: Program running 3: Program waiting		r	INT
\$AC_IPO_BUF	Number of preprocessed blocks	r	r	INT
\$AC_SYNA_MEM	Number of available elements for synchronized actions	r	r	INT
\$AA_STAT[axis]	Axis status 0: No status available 1: Traverse motion active 2: Axis has reached end of IPO 3: Axis in position (coarse target range) 4: Axis in position (fine target range)		r	INT

r = read, w = write

Table 10-7 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AA_TYP[axis]	Axis type 0: Axis in a different channel 1: Channel axis (path or positioning motion) 2: Neutral axis 3: CPU axis 4: Reciprocating axis 5: Neutral axis currently traversed in JOG mode 6: Master value-coupled slave axis 7: Coupled motion of slave axis 8: Command axis (MOV, POS from synchronized action)	r	r	INT
\$AA_FXS[axis]	Status of travel to fixed stop 0: Fixed stop not reached 1: Fixed stop reached 2: Error in approach	r	r	INT
\$AA_MOTENDA[axis]	1: Block change "Exact stop fine" 2: Block change "Exact stop coarse" 3: Block change "IPO end" 4: Block change "Braking ramp"	r	r	INT
\$AC_PRESET[X]	Last preset value defined	r		REAL
\$MC_CHAN_NAME	"CHAN1"..."CHAN4"	r	r	STRING
\$P_ACTID[ID No.]	Status of the synchronized action 0: not active 1: active	r	r	INT
\$P_PROG_EVENT	Request call event 1: Start 2: End of program 3: Reset 4: FM restart	r	r	INT
Programming				
\$P_F	Last programmed path feed F	r		REAL
\$P_FA[X]	Last programmed positioning axis feed	r		REAL
\$P_EP[X]	Last programmed setpoint (end point)	r		REAL
\$P_GG[n]	Current G function of a G group, n... name of G group	r		INT
\$P_AXNn	Current axis name of the geometry axis n=1 abscissa n=2 ordinate n=3 applicate	r		INT

r = read, w = write

Table 10-7 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$PI	Circle constant PI, fixed value PI= 3,1415927	r		REAL
\$PI	Circle constant P1, permanent value PI= 3,1415927	r		REAL
Electronic gear				
\$AA_EG_TYPE[FA,LA]	Kind of coupling 0: Actual-value coupling 1: Setpoint coupling	r	r	INT
\$AA_EG_NUMERA[FA,LA]	Numerator of coupling factor Number of curve table if denominator = 0	r	r	REAL
\$AA_EG_DENOM[FA,LA]	Numerator of coupling factor	r	r	REAL
\$AA_EG_SYN[FA,LA]	Synchronized position for the specified master axes	r	r	REAL
\$AA_EG_SYNFA[FA,LA]	Synchronized position for the specified following axis	r	r	REAL
\$AA_EG_BC[FA]	Block change mode when activating the EG for the specified following axis	r	r	STRING
\$P_EG_NUM_LA[FA]	Number of the master axes defined for the specified following axis	r	r	INT
\$AA_EG_AX[n, FA]	Axis name of the nth following axis	r	r	AXIS
\$AA_EG_ACTIVE[FA,LA]	Status of the electronic gear with reference to the specified master axis and following axis 0: disabled 1: enabled	r	r	BOOL
\$VA_EG_SYNCDIFF[FA]	Amount of the synchronism difference with reference to the specified following axis	r	r	REAL
\$VA_EG_SYNCDIFF_S[FA]	Synchronism difference with reference to the specified following axis	r	r	REAL
FIFO memory				
\$AC_FIFO[n...] n = 1...4	System variable for access to FIFO range	r / w	r / w	REAL

r = read, w = write

10.21 FIFO variable (\$AC_FIFO1[...] to \$AC_FIFO10[...])

General

Data sequences to be connected can be stored in one FIFO range. Possible applications are continuous measuring (MEAC) or product tracing with continuous processes.

The FIFO variables will be created in the range of the R parameters, thus constituting a special form of an R parameter access.

Define start, number and length of the FIFO ranges via parameterization (see Section 9.1.4, Configuration parameters).

This function is provided as of software release 5.2.

Structure of the FIFO area

Fig. 10-57 shows the structure of the FIFO area. The special functions for the FIFO management are realized by the elements \$AC_FIFO_n[0] to \$AC_FIFO_n[5].

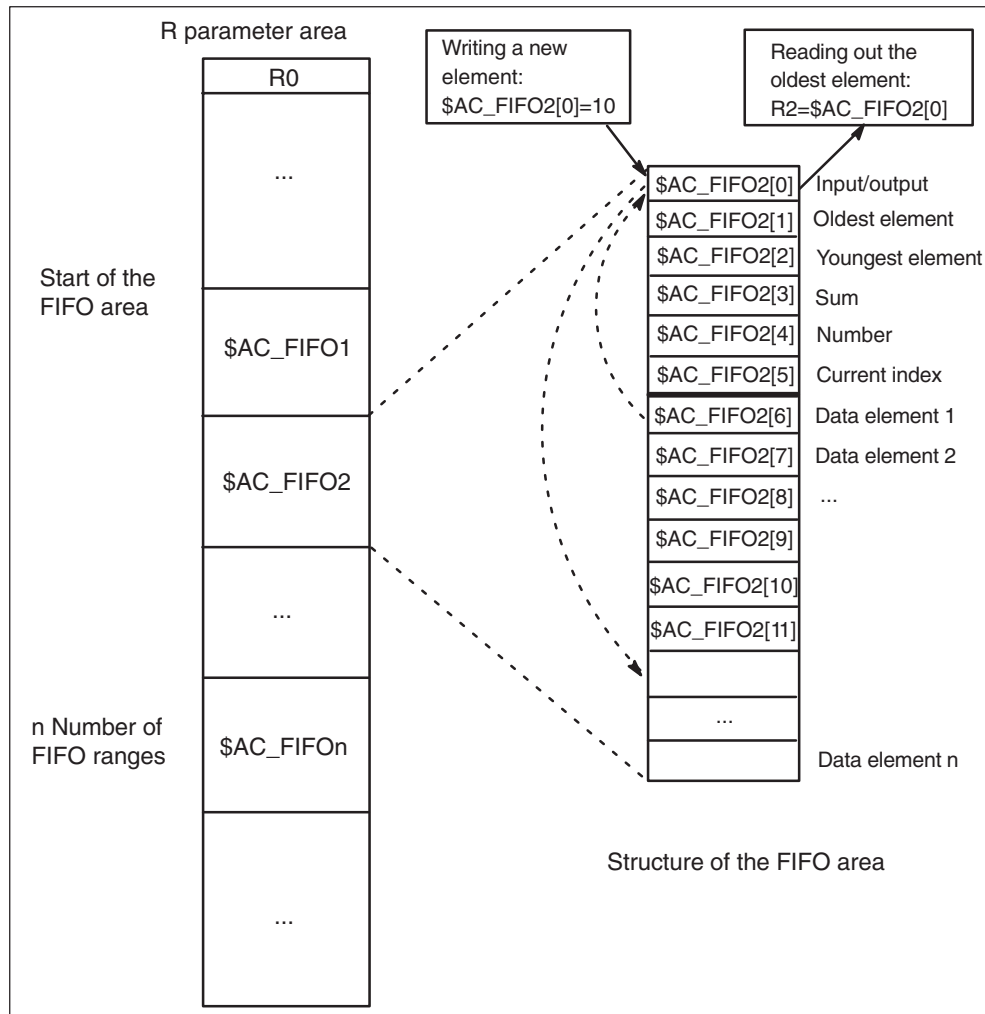


Fig. 10-57 Structure of the FIFO area

Function of the FIFO elements

\$AC_FIFO[0]	Write:	A new value is stored in the FIFO (maximum number = number of FIFO elements)
	Read:	The oldest element is read and removed
\$AC_FIFO[1]	Read:	Value of the oldest element
	Write:	not permitted
\$AC_FIFO[2]	Read:	Value of the youngest element
	Write:	not permitted
\$AC_FIFO[3]	Read:	Sum of all elements
		Note: Parameter "FIFO area sum generation" must be activated.
	Write:	not permitted
\$AC_FIFO[4]	Read:	Number of entered elements
	Write:	only \$AC_FIFO[4]=0 i.e. Reset FIFO area
\$AC_FIFO[5]	Read:	Index to the next FIFO element to be written
	Write:	not permitted
\$AC_FIFO[6 and higher]	Read:	Value (data element)
	Write:	not permitted

The parameter "Number of FIFO elements" defines the number of data elements.
The number of required R parameters is calculated as follows:

Number of R parameters =
number of FIFO ranges * (number of FIFO elements + 6)

If not sufficient R parameters exist, the error message 20170 (FIFO has exceeded range for R parameters) is issued after FM restart. In this case, you will have to reduce the FIFO range or increase the number of the R parameters.

The parameters "Number of FIFO elements" and "FIFO range sum generation" are the same for all FIFO ranges.

Programming example

The parts lying on the conveyor line are to be picked up by the gripper and be piled. The line axis runs at constant speed and does not possess a measuring system. A sensor is installed to acquire the position of the parts on the line. During the acceptance, the gripper is to traverse synchronously with the line. Make sure that the gripper cycle is completed reliably before the next part is picked up by the gripper.

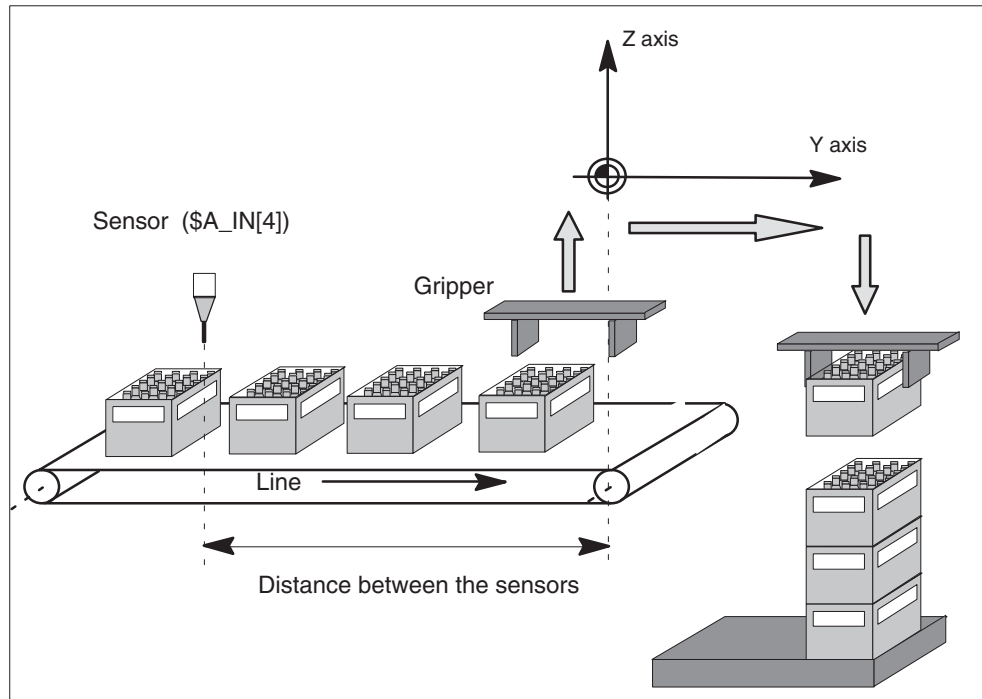


Fig. 10-58 Machining example

; Definitions

N10 DEF REAL PART_L = 300

N15 DEF REAL CON_L = 1510

N20 DEF REAL SENSOR_DISPL = 1850

N25 DEF REAL MIN_SYN = 10

N30 R5=0

N35 \$AC_MARKER[0]=0

N40 \$AC_MARKER[1]=0

N40 \$AC_MARKER[4]=0

; Part length [mm]

; Min. conveying path from the sensor

; Distance from the sensor to Y = 0

; Minimum synchronous travel

; Line speed [mm/s]

; Flag of the starting cycle

; Flag of the step sequence

; FIFO1 – clear memory

; Sum generation is parameterized

; Curve table

N50 CTABDEF (Y,X,1,0)

N55 X0 Y0

N60 X100 Y100

N65 CTABEND

; 1:1 coupling

; Leading axis X is simulation axis

; Park position

N100 G0 Z50

N105 Y0

```

N110 PRESETON (X, 0)
N115 CANCEL ()

; Waiting for first workpiece
N120 WHENEVER $A_IN[4]== FALSE DO RDISABLE
N125 EVERY $A_IN[4] == TRUE DO $AC_TIMER[1]=0 ; Start of 1st timer
N130 G4 F0.1 ; Dummy block

; Time measurement and calculation of line speed (cyclically)
N150 ID=1 EVERY $A_IN[4] == FALSE
DO $R5 = PART_L / $AC_TIMER[1] ; Calculate line speed
$AC_FIFO1[0] = PART_L ; Save part length
$AC_TIMER[1] = 0 ; Restart timer
$AC_MARKER[0] = 1 ; Start cyclic time measurement

N160 ID=2 EVERY ($A_IN[4] == TRUE) AND ( $AC_MARKER[0]==1 )
DO $AC_FIFO1[0] = $AC_TIMER[1] * $R5 ; Save part spacing
$AC_TIMER[1] = 0 ; Restart timer

; Step sequence
; 1. Wait for part in transfer position
N200 ID=3 EVERY ($AC_MARKER[1] == 0) AND ($AC_FIFO1[3] > CON_L )
DO PRESETON ( X,$AC_FIFO1[3]-SENSOR_DISPL) ; Set actual value X
$R3=$AC_FIFO1[0] $R3=$AC_FIFO1[0] ; Delete twice
$AC_MARKER[1]=1 ; Next step

; 2. Start simulated leading axis X and enable master-value coupling
N210 ID=4 EVERY $AC_MARKER[1]==1
DO MOV[X]=1 FA[X]=$R5*60 ; Start X axis
LEADON(Y,X,1) ; Master-value coupling ON
$AC_MARKER[1]=2 ; Next step

; 3. Enable gripper cycle after synchronism has been reached
N220 ID=5 EVERY ($AC_MARKER[1]==2) AND ($AA_IM[Y] > MIN_SYN)
AND ($AA_SYNC[Y] > 2)
DO $AC_MARKER[1]=3 ; Enable cycle

; 4. Gripper cycle
N290 G0 G64
N300 ZYKLUS:
N305 WHENEVER $AC_MARKER[1] <> 3 DO RDISABLE ; Wait for enable
N310 G4 F0.1 ; Dummy block
N315 Z0 M60 ; Pick up part
N320 WHEN TRUE DO LEADOF(Y,X) ; Master-value coupling OFF
N325 Z10 ; Lift part
N330 Y700 ; Transport part
N335 Z-20 M61 ; Deposit part
N340 Z10
N345 Y0 ; Park position
N350 WHEN TRUE DO MOV[X]=0 $AC_MARKER[1]=0 ; Restart step sequence
N355 G4 F0.1
N360 GOTOB ZYKLUS

N400 M2

```


Note:

To test the program, you can use the following synchronized actions:

```
WHEN TRUE DO $AC_TIMER[2]=0  
ID=10 EVERY $AC_TIMER[2] > 6 DO $A_OUT[4]=1 $AC_TIMER[3]=0  
ID=11 EVERY $AC_TIMER[3] > 9 DO $A_OUT[4]=0 $AC_TIMER[2]=0
```

Insert these synchronized actions after block N115 and connect the digital output 4 to the digital input 4 or replace \$A_IN[4] in the program by \$A_OUT[4].

10.22 User variables

General

In addition to the R parameters and system variables always existing, you can use your own user variables. The user variables must be defined before use and differ themselves in their range of validity.

Range of validity

Table 10-8 Range of validity of user variables

Variable	Meaning
LUD	Local user variables: <ul style="list-style-type: none"> are defined in the NC program only apply in the current program/program level become invalid with end of program or reset
GUD	Global user variables: <ul style="list-style-type: none"> are defined separately in a GUD block are valid in all program levels also after reset and power-on of the FM

Number and memory allocation of the user variables

Table 10-9 Variable types

Variable	Memory Type Assignment	Number	Memory for Management Values	
Arithmetic parameters	SRAM channel	100 (default value) max. 10 000 ¹⁾	–	800 bytes 80,000 bytes
LUD	DRAM channel	fixed 200	150 bytes per LUD	12 kB
GUD FM global	SRAM FM	10 (default value) max: ¹⁾	80 bytes per GUD	12 kB (default value) max: ¹⁾
GUD channel-global	SRAM channel	40 (default value) max: ^{1) 2)}	80 bytes per GUD	(FM global)

1) can be parameterized depending on the free SRAM available

2) The number of variable names is parameterized. This name will apply to all channels. The value assignment, however, is channel-dependent (max. 4 values per variable).

With LUD or GUD variables, the data type determines the memory required for storing the variable values:

INT	4 bytes
REAL	8 bytes
BOOL	1 bytes
CHAR	1 bytes
STRING	1 bytes per character, max. 200 characters
AXIS	4 bytes

Definition of the user variables

The definition of the user variables is carried out using the **DEF** statement and has to be carried out in a separate block. Only one variable type can be defined in a DEF block.

LUD variable

This variable type has to be defined in the beginning of the program.

The data are stored in DRAM and are only valid up to the end of the program or until a reset is carried out.

Notice

Do not use statements as variable names (see Table 10-12) or system variables (see Table 10-7).

GUD variable

GUD variables have to be defined in a separate GUD block via the start-up tool. The definition must include a range of validity:

```
DEF NCK                ; GUD exists once FM globally
DEF CHAN                ; GUD exists once for each channel
```

Another attribute is provided to generate preprocessing stop when reading or writing the GUD variables:

```
DEF NCK SYNRW          ; Preprocessing stop when reading/writing
DEF NCK SYNRR          ; Preprocessing stop when reading
```

GUD values are saved in the SRAM and are valid also after reset and power-on of the FM. The number of the GUD variables, and the storage range for the values can be set in the start-up tool.

Programming

DEF file type name ; Definition of a variable without value assignment
DEF data type name=value ; Definition of a variable with value assignment

DEF data type name[n,m] ; Field definition without value
; assignment
; n = field size of 1st dimension (row)
; m = field size of 2nd dimension
; (column)

DEF data type name[n,m]=(value,value ...); Field definition with value assignment

Name[n,m] = SET (value, expression, value, ...)
; Initialization with different values

Name[n,m] = REP (value or expression) ; Initialization with the same value

ISVAR(name) ; checks whether the variable name
; is known (software version 5
; and higher)

Possible variable types are: INT, REAL, BOOL, CHAR, STRING, AXIS.

The variable name can contain a maximum of 31 characters. The first two characters must be letters or underscores. The \$ character is not permitted. In the case of definitions without value assignments, the variable value is assigned zero by default.

Fields can be defined two-dimensionally as the maximum, string fields, however, only one-dimensionally. The first field element starts with the index [0,0], and the last field element has the index [n-1, m-1].

The value assignment in the definition of fields is carried out from the left to the right and from top to the bottom, from the view of the table structure. Any elements not specified are initialized with zero.

Examples for the definition of user variables

```
DEF REAL POSITION1      ; User variable of the type REAL with name
                       ; POSITION1, initial value = 0.0

DEF INT START_1=1, END=10; Several user variables of the type INT
                       ; with different initial values

DEF CHAR CHARACTER="A" ; User variable of the type CHAR
                       ; Initial value is the ASCII character "A"

DEF AXIS TABLE=X      ; User variable of the type AXIS

DEF REAL TAB_REAL[2,4] ; Field with name TAB_REAL,
                       ; Field elements of the type REAL
                       ; 2 lines with 4 columns each

DEF BOOL BIT_ARRY[5,5]=(TRUE, TRUE) ; Field definition with value
                                     ; assignment
                                     ; Element[0,0] = TRUE
                                     ; Element[0,1] = TRUE
                                     ; Elements[0,2] to [4,4] = FALSE
```

Application example of a LUD variable

```

; Program loop
N10 DEF INT COUNTER = 0
...
START:
N30 COUNTER=COUNTER+1           ; Increment counter variable
N30 G91 X5 Y5
N40 IF COUNTER<50 GOTOB START   ; Return to START until
                                ; COUNTER < 50
...

; Use of the data type AXIS
N10 DEF AXIS ABSZ=X, ORDI=Y
N20 DEF REAL POS_ABSZ
...
N40 POS[ABSZ]=100 FA[ABSZ]=2000 ; Positioning of the X axis
N50 POS[ORDI]=200 FA[ORDI]=1500 ; Positioning of the Y axis
N60 POS_ABSZ=$AA_IW[ABSZ]       ; Reading of the actual position of the
                                ; X axis
...

; Initialization and calculation with fields

N10 DEF INT I_M, I_N           ; INT variable (index and degree)
N20 DEF REAL R_A[360,2]       ; REAL field with 720 elements
N30 R10=0

; Writing a field with SIN and COS values in steps of 0...360 degrees
:
N40 R_A[I_N, I_M] = SIN(I_N)   ; Sine 0...360 degrees
N50 R_A[I_N, I_M+1] = COS(I_N) ; Cosine 0...360 degrees
N60 R10=R10+SQRT (POT(R_A[I_N, I_M]) + POT(R_A[I_N, I_M+1]))
N70 I_N=I_N+1
N80 IF I_N<360 GOTOB W_FELD

; Error evaluation: Are all elements calculated correctly?

IF R10 <> 360 GOTOF ERROR
; Correct, Mr. Pythagoras !
M0
M30
ERROR:
; ?
M0
M30

```

Application example of a GUD variable

Definition file for a GUD variable:

```
N10 DEF NCK REAL R_WERT_A           ; GUD variables FM globally

M30
```

Application variable: Checking for variable

```
DEF INT VAR1
DEF BOOL IS_VAR=FALSE
N10 IS_VAR=ISVAR("VAR1")           IS_VAR is TRUE in this case
...

DEF REAL VARARRAY[10,10]
DEF BOOL IS_VAR=FALSE
N20 IS_VAR=ISVAR("VARARRAY[ , ]")  ; IS_VAR is TRUE in this case,
                                   ; variable is two-dimensional
IS_VAR=ISVAR("VARARRAY") ; IS_VAR is TRUE, variable exists

N40 S_VAR=ISVAR("VARARRAY[8,11]")  ; IS_VAR is FALSE, illegal array index

N50 IS_VAR=ISVAR("VARARRAY[8,8]")  ; IS_VAR ist FALSE, missing "]"

N60 IS_VAR=ISVAR("VARARRAY[ ,8]")  ; IS_VAR is TRUE, illegal Arrayindex

N70 S_VAR=ISVAR("VARARRAY[ 8 , ")  ; IS_VAR is TRUE
...

DEF BOOL IS_VAR=FALSE
N10 IS_VAR=ISVAR("$AA_IW")         ; IS_VAR is TRUE in this case
N20 IS_VAR=ISVAR("$AA_IW[X]")     ; IS_VAR is TRUE in this case
```

10.23 Program jumps (GOTOF, GOTOB, GOTO, GOTOC, LABEL, IF)

General

Programs are executed block by block, from the first written block to the last.

You can modify this order by including program jumps in a separate block.

Programming

GOTOF LABEL	; Unconditional jump forward
GOTOB LABEL	; Unconditional jump backward
GOTO	; Unconditional jump forward/reverse (software version 5 and higher)
GOTOC	; Unconditional jump forward / reverse / continue (software version 5 and higher)
IF condition GOTOF LABEL	; Conditional jump forward
IF condition GOTOB LABEL	; Conditional jump backward
IF Bedingung GOTO	; Conditional jump forward/reverse
IF Bedingung GOTOC	; Conditional jump forward/reverse/continue
LABEL	; Destination (in jump statement)
LABEL	; Jump destination (label in program)

Jump destinations (labels)

Jump destinations (labels) must be entered as user-defined names. A name can comprise a minimum of 2 and a maximum of 25 characters (letters, digits, underscore). The **first two** characters must be letters or underscores. A colon ":" must be inserted after the label name, in order to identify the name as a label in the user program.

Labels are always programmed at the beginning of the block, immediately after the block number (if one is used).

Labels must be unique within a program.

The statements in Table 10-12 must not be used.

Examples:

N10 LABEL1: G1 X20	; LABEL1 is a label
TR78943: G0 X10 Y20	; TR78943 is a label, no block number
GOTOB TOP	; Label here as destination without colon

Jump without direction specification GOTO

The jump destination is searched for first in the forward and then in the reverse direction.

If no jump destination is found, the error message 14080 "Jump destination not found" is issued.

Jump without direction specification GOTOC

The jump destination is searched for first in the forward and then in the reverse direction.

If the jump destination is not found, the **program is continued** with the next NC block.

Unconditional program jumps

Unconditional program jumps are always executed. Infinite loops or exit jumps can be programmed after conditional jumps, for example.

Example:

```
N10 G...           ; Starting point for infinite loop
N20 TOP:          ; Define jump destination
...              ; The blocks between N10 and N100 are
...              ; executed cyclically (infinite loop)
...              ; The program is terminated with RESET
N100 GOTOB TOP   ; Jump backwards
```

Conditional program jumps

Conditional program jumps are programmed with an IF statement. When the condition is true, the program jumps to the block with the specified label.

Example:

```
...
N20 IF R1<R2 GOTOF LABEL1 ; If condition is fulfilled, then jump to
                          ; block with LABEL1
N70 LABEL1: G1 ...
```

10.24 Control structures

General

In addition to the program jumps, you can use these statements to realize program loops and branches.

Programming

IF ELSE ENDIF	; Choice between 2 alternatives
LOOP ENDLOOP	; Endless loop
FOR ENDFOR	; Counter loop
WHILE ENDWHILE	; Loop with condition at loop start
REPEAT UNTIL	; Loop with condition at loop end
CASE	; Case differentiation

IF ELSE ENDIF – choice between two alternatives

The IF ELSE ENDIF statement serves to choose from two alternatives.

Programming

IF condition

1st alternative; is executed if the condition is fulfilled

ELSE

2nd alternative; is executed if the condition is not fulfilled

ENDIF

Note: ELSE branch can also be dropped.

Example:

```
N50 IF (R10 > 5)
N60 G0 X200           ; X axis will traverse if R10 > 5
N70 ELSE
N80 G0 Y100           ; Y axis will traverse if R10 <= 5
N90 ENDIF
```

LOOP ENDLOOP – endless loop

This statement can be used for endless loops. A jump back to the loop start is always carried out at the loop end. The NC program must be quitted with reset.

Programming**LOOP**

NC block sequence

ENDLOOP**Example:**

N30 LOOP

N40 G0 Y100

N50 G1 Y0 F500 ; Endless motion of the Y axis

N60 ENDLOOP

N100 M30

FOR ENDFOR – counter loop

The counter loop is used to repeat a specific number of passes. The number of passes defines a counter variable of the type INT. If the initial value is greater than the final value, the loop is not executed. Negative values are permissible.

Programming

FOR INT variable = initial value **TO** final value

NC block sequence

ENDFOR**Example:**

N10 DEF INT COUNTER

N30 FOR COUNTER = 0 TO 100 ; The counter loop is passed **101** times.

N40 G0 Y100

N50 G1 Y0 F500

N50 ENDFOR

END:

N80 M30

WHILE ENDWHILE – loop with condition at the loop start

The loop is passed as long as the condition at the loop start is fulfilled.

Programming

WHILE condition
NC block sequence
ENDWHILE

Example:

```
N30 WHILE ($A_IN[1]==TRUE) AND ($A_IN[2]==TRUE)
N40 G0 Y100
N50 G1 Y0 F500
N60 ENDWHILE

N80 M30
```

REPEAT UNTIL – loop with condition at the loop end

The loop is passed as long as the condition is not fulfilled at the loop end.

Programming

REPEAT
NC block sequence
UNTIL condition

Example:

```
N30 REPEAT
N40 G0 Y100
N50 G1 Y0 F500
N60 UNTIL ($A_IN[1]==TRUE) AND ($A_IN[2]==TRUE)

N80 M30
```

CASE – case differentiation

The CASE statement can be used to branch differently, depending on the value of a variable or an expression of the type INT. The default branch is passed if the variable or the expression does not assume any of the specified constants. If no default branch is programmed, the execution is continued with the next NC block following after the CASE statement. The whole statement must be programmed in an NC block without gaps.

Programming

CASE expression **OF** constant1 **GOTOF** LABEL1 ... **DEFAULT GOTOF** LABELn

CASE expression **OF** constant1 **GOTOB** LABEL1 ... **DEFAULT GOTOB** LABELn

Example:

```
N10 CASE (R10+R11) OF 1 GOTOF FALL1 2 GOTOF FALL2 3 GOTOF FALL3
```

```
; Default: do nothing
```

```
M30
```

```
FALL1:
```

```
N20 G0 Y100
```

```
N30 G1 Y0 F500
```

```
N40 M30
```

```
FALL2:
```

```
N50 G0 X100
```

```
N60 G1 X0 F1000
```

```
N70 M30
```

```
FALL3:
```

```
N80 G0 Z100
```

```
N90 G1 Z0 F1000
```

```
N100 M30
```

Supplementary conditions

Control structures always apply in the current program level. A nesting depth of max. 8 control structures is possible within each program level.

Example:

```

; Main program
N10 LOOP
N20   FOR
N30   WHILE
N40   L10 ; -----> ; Subroutine L10
N50   ENDWHILE
N60   ENDFOR
N70 ENDLOOP
N80 M30

N10 WHILE
N20   FOR
N30   FOR
N40   ...
N50   ENDFOR
N60   ENDFOR
N70 ENDWHILE
N80 M30

```

Control structures are processed interpretatively, i.e. as long as the condition is fulfilled, no blocks after the control structure are processed.

If possible do not use program jumps from control structures.

Definitions of user variables at the program start may not be part of a control structure.

Block skipping and labels are not permitted in blocks containing control structure statements.

10.25 Axis variable

General

Each axis is assigned a fixed (internal) axis name either via the parameterization or as a default value. Variables of the data type AXIS can be assigned an axis name. The statements AXNAME, ISAXIS and AX allow the conversion to the data type AXIS and the programming with axis variables.

Programming

DEF AXIS axis variable = X	; Definition of an axis variable and assignment of a (valid) axis name
AXNAME (STRING)	; Type conversion STRING by axis name
AXSTRING (AXIS)	; Type conversion axis name by STRING
ISAXIS (axis number)	; Test for permissible geometry axis no's 1...3
AX[axis variable]	; Axis programming of a variable axis name

Examples with axis variables

```

N10 DEF AXIS ABSZI                ; Definition of the axis variable
N20 DEF AXIS ORDI
N30 DEF STRING[4] Achsen = "XYZA" ; Axis name as a string variable

N40 IF ISAXIS (1) == FALSE GOTOF NO_ABSZI
                                ; Does the 1st geometry axis exist?
N50 ABSZI = $P_AXN1              ; Assignment of the axis name of the
                                ; 1st geometry axis to the variable
                                ; ABSZI
N60 G1 AX[ABSZI]=100 F200        ; G1 block with variable axis address
N70 POS[ABSZI]=50 FA[ABSZI]=300 ; POS motion with variable
                                ; Axis address

NO_ABSZI:
N80 IF ISAXIS(2) == FALSE GOTOF NO_ORDI
N90 ORDI = AXNAME(axes[1])      ; Assignment from string and
                                ; type conversion

N100 G1 AX[ORDI]=200 F400
N110 AX[AXNAME(axes[1])]=100

NO_ORDI:
; Positioning of the A axis
N120 POS[AXNAME(axes[3]) = 50 FA[AXNAME(axes[3])]=2000
...

```

10.26 String operations

General

User variables of the type STRING can incorporate an ASCII string.

String operations can be used, e.g. for generating variable program calls or for reading from or writing to an NC program.

Programming

DEF STRING[m] name	; Definition of a string variable; string length m
STRLEN (STRING)	; Length of a string
<<	; Linking and type conversion to STRING
ISNUMBER (STRING)	; Type conversion STRING to BOOL
NUMBER (STRING)	; Type conversion STRING to REAL
AXNAME (STRING)	; Type conversion STRING to AXIS
AXSTRING (AXIS)	; Type conversion AXIS to STRING
TOUPPER (STRING)	; Conversion lowercase to uppercase letters
TOLOWER (STRING)	; Conversion uppercase to lowercase letters
INDEX (STRING, CHAR)	; String search in the STRING, starting from the ; beginning of the string
RINDEX (STRING, CHAR)	; Character search in the STRING, starting from ; the string end
MINDEX (STRING, CHAR, CHAR, ...)	; Search for several characters in the ; STRING, starting from the beginning of ; the string
MATCH (STRING, STRING)	; String search in the STRING
STRING[M]	; Selection of a character with index M
SUBSTR (STRING, M)	; Selection of a partial string from index M up to the ; end of the string
SUBSTR (STRING, M, N)	; Selection of a partial string with length N from ; index M

String assignment

A STRING variable can be assigned ASCII characters directly. The characters must be put in brackets "...". If " or ' are to be parts of the string, these must be put into brackets '...'. The string length must be equal to or greater than the assigned character.

Individual ASCII characters, e.g. characters that cannot be represented, can also be assigned as binary or hexadecimal values via the field index. The maximum valid index is the end identifier (character 0) of the string variable.

Examples

String assignment:

```

N10 DEF STRING[25] STG="Axis      is traversing" ; Assignment as an
                                           ; ASCII character string
N20 STG[5] = 96                               ; Write single character ' to string
N30 STG[6] = "X"                             ; Write single character X to string
N40 STG[7] = 96                               ; STG now: "Axis 'X' is traversing"
N50 STG[8] = 0                               ; Shift end identifier
                                           ; STG now: "Axis 'X'"

```

String operations:

```

N10 DEF STRING[45] JOB = "Traverse the X axis to 123.4, then to 10"
N20 DEF STRING[45] RESULT
N30 DEF AXIS axis
;
N40 axis = AXNAME (SUBSTR (JOB, 13 ,1))

```

; Select position values from JOB and type conversion to REAL

```

N50 G0 AX[axis]= NUMBER (SUBSTR (JOB, 23, 5)) ; Rapid traverse rate
N60 POS[axis]= NUMBER (SUBSTR (JOB, 38, 2)) ; Positioning motion

```

; Create string via string linking and type conversion

```

N70 RESULT = "Position "<<AXSTRING(axis)<<" is: "<<$AA_IM[axis]

IF SUBSTR(JOB, 38, 2) <> SUBSTR(RESULT, 15, 2) GOTOF ERROR

N90 RESULT = RESULT<<" , this is correct"
N100 M30

ERROR:
N110 RESULT = RESULT<<" , this is wrong"
N120 M30

```

10.27 Reading, writing and deleting a file

General

These statements can be used for generating and evaluating NC programs. Possible applications are, e.g. log files for measurement results. The size of an NC program created using the WRITE statement may not exceed 100 kbytes.

Programming

WRITE (error, file name, text)

error	Return value, variable of the type INT 0 No error 1 Path not permitted 2 Path not found 3 File not found 4 Wrong file type 10 File is full 11 File is in use 12 No resources free 13 No access rights 20 Other errors
filename	The file to which the data are to be written is a variable of the type STRING. The file name can be specified with path and file extension. If no file extension is specified, .mpf will be added. Without specifying a path, the file is stored in the current directory, i.e. in the directory of the selected program. Max. file name length: 25 characters Max. path name length: 128 characters Example: TESTFILE TESTFILE_MPF /_N_MPF_DIR/_N_TESTFILE_MPF/ When specifying the path, the domain identifier _N_ must be specified. If the file does not yet exist, it will be created.
Text	Text to be written; variable of the type STRING; max. 200 characters, (an LF is additionally added internally)

READ (error, file name, line, number, result)

error	Return value; variable of the type INT. 0 No error 1 Path not permitted 2 Path not found 3 File not found 4 Wrong file type 13 Insufficient access right 21 Line does not exist 22 Field length of the result variable "result" too small 23 Range of lines too large; parameter "number" goes beyond file end
file name	File from which data are to be read; variable of the type STRING For further information, see WRITE statement
line	Range of lines from which data are to be read; variable of the type INT. 0 The number of lines before the end of the program, which are specified with the parameter "number" is read. >0 Number of the first line to be read
number	Number of lines to be read; variable of the type INT.
result	Variable of the type STRING in which the read text is stored (maximum length 255 characters) The control characters LF or CR LF will not be accepted from the read file. Read lines are cut without an error message if the line is longer than the string length of the target variable "result". Files of the type _BIN, _EXE, _OBJ, _LIB, _BOT, _TRC, _ACC, _CYC, _NCK cannot be read.

DELETE (error, file name)

error	Return value; variable of the type INT. 0 No error 1 Path not permitted 2 Path not found 3 File not found 4 Wrong file type 11 File is in use 12 No resources free 20 Other errors
file name	File that is to be deleted; variable of the type STRING For further information, see WRITE statement.

result = ISFILE (file name)

file name File for which the test is carried out; variable of the type STRING
 For further information, see WRITE statement

result Variable of the type BOOL.
 TRUE File exists
 FALSE File does not exist

Software version 5 and higher:

FILEDATE (error, filename, _date)

error Return value, variable of type INT (see WRITE statement)

filename File to be checked; variable of type STRING
 For further information, see WRITE statement

_date Variable of the type STRING, 8 characters

FILETIME (error, filename, _time)

error Return value, variable of type INT (see WRITE statement)

filename File to be checked; variable of type STRING
 For further information, see WRITE statement

_time Variable of the type STRING, 8 characters

FILESIZE (error, filename, _size)

error Return value, variable of type INT (see WRITE statement)

filename File to be checked; variable of type STRING
 For further information, see WRITE statement

_size Variable of the type INT

Examples

```

; A program for time measurement TestProg1 is to be created.
; TestProg1, in turn, will create a program LOG where
; the measured time is entered.
; Note: To obtain more clarity, no evaluation of the
; return value ERROR has been carried out.

; Variable definition
N10 DEF INT ERROR
N20 DEF STRING[20] ProgName="TestProgr1.mpf"
N30 DEF STRING[50] SN10="N10 DEF STRING[50] STARTZ=' ' ' Start time: ' ' ' "
N40 DEF STRING[50] SN20="N20 DEF STRING[50] ENDEZ=' ' ' End time: ' ' ' "
N50 DEF STRING[50] SN30="N30 DEF INT ERROR_2
N60 DEF STRING[50] SN40="N40 G0 G90 G64 X0 Y0 Z0 F100"
N70 DEF STRING[100] SN50="G1 G91 X1 Y1 Z1"
N80 DEF STRING[20] Prog
N90 DEF INT IDX=0

; Create program name
N100 IDX=MATCH(ProgName, ".mpf")
N110 Prog=SUBSTR(ProgName,0, IDX)

; N120 DELETE (ERROR, Prog)          ; Delete program (if any)

; Write to program:
; Write program header
WRITE (ERROR, Prog, "; Write test to file")
WRITE (ERROR, Prog, "; Program name: "<<Prog)
WRITE (ERROR, Prog, "; date "<<$A_DAY<<."<<$A_MONTH<<."<<$A_YEAR)
WRITE (ERROR, Prog, "; time: "<<$A_HOUR<<."<<$A_MINUTE<<."<<$A_SECOND)

; Write NC blocks from predefined STRING variable
WRITE (ERROR, Prog, SN10 )
WRITE (ERROR, Prog, SN20 )
WRITE (ERROR, Prog, SN30 )
WRITE (ERROR, Prog, SN40 )

; Write block for starting the time measurement
WRITE (ERROR, Prog, "N50 STARTZ=STARTZ<<$A_HOUR<<."<<$A_MI-
NUTE<<."<<$A_SECOND          ; Start of time measurement

; Create ten G91 blocks with consecutive block number
N130 R10 = 0
N140 WHILE R10 < 10
N150 WRITE (ERROR, Prog, "N"<<R10+60<<SN50 )
N160 R10 = R10+1
N170 ENDWHILE

```

```
; Write block for end of time measurement
WRITE (ERROR, Prog, "N80 ENDEZ=ENDEZ<<$A_HOUR<<".<<$A_MI-
NUTE<<".<<$A_SECOND    ; End of time measurement

; Blocks for creating the log file
WRITE (ERROR, Prog, "N100 DELETE (ERROR_2, ""LOG"" )" )
WRITE (ERROR, Prog, "N110 WRITE (ERROR_2, ""LOG"", "" result of time mea-
surement: "" )" )
WRITE (ERROR, Prog, "N100 DELETE (ERROR_2, ""LOG"", STARTZ) )
WRITE (ERROR, Prog, "N100 DELETE (ERROR_2, ""LOG"", ENDEZ) )

; Call created program
CALL Prog

; Delete created program
DELET (ERROR, Prog)

; Measurement results are contained in the file LOG

M30
```

10.28 Program coordination (INIT, START, WAITE , WAITM, WAITMC, SETM, CLEARM)

General

Normally, the program is executed in the individual channels independently without mutual influencing.

If a coordination of the program sequences is required, you can use the following statements.

The statements require a separate NC block each.

Programming

INIT (Channel no., "Program")	; Program selection
START (Channel no., channel no., ...)	; Program start
SETM (label1, label2, ...)	; Set synchronism marker
CLEARM (label1, label2, ...)	; Delete synchronism marker
WAITM (label1, channel no., channel no., ...)	; Wait for synchronism marker
WAITMC (label1, channel no., channel no., ...)	; Conditioned waiting for ; synchronism marker
WAITE (channel no., channel no., ...)	; Wait for program end

Parameters

Channel no.	; Channel no's 1...4
Program	; Program name
Marker	; Number of the synchronism marker; 10 synchronism markers ; per channel

INIT Program selection

Program selection for a specified channel. Mode and channel status of the appropriate channels must permit program selection. Program selection for the own channel is not possible.

The program name can be specified either with or without path specification and with or without domain identifier.

Examples:

Program selection for the second channel; program name ProgK2.

INIT (2, "/_N_MPF_DIR/_N_ProgK2_MPF") ; with path specification and domain
; identifier

INIT (2, "_N_ProgK2_MPF") ; with domain identifier (_N_)

INIT (2, "ProgK2") ; without path specification and domain
; identifier

Start Program start

Program start for one or several channels. Mode and channel status of the appropriate channels must permit a program start. Program start for the own channel is not possible.

Example:

```
; Program runs in channel 1
...
N10 INIT(2, "ProgK2")      ; Program selection for the 2nd channel
N20 INIT(3, "ProgK3")      ; Program selection for the 3rd channel
N30 START(2,3)             ; Program start for channel 2 and 3
...
```

SETM/CLEARM Set / clear a synchronism marker

Sets or clears a synchronism marker in the own channel.

SETM or CLEARM has no influence on the program execution in the own channel. The synchronism marker is set or cleared with block output. The status of the synchronism marker is kept after reset and start.

SETM or CLEARM can also be programmed as an action in a synchronized action (Section 10.32). A maximum of 10 synchronism markers are possible per channel.

A set synchronism marker cannot be set once more; multiple clearing is possible.

CLEARM() without specifying the synchronism marker will clear all synchronism markers in the channel.

An executable block, e.g. G4, must be programmed between SETM and WAIT... statements.

WAITM Wait for synchronism marker

Waiting for a synchronism marker from the specified channels. WAITM will create exact stop in the previous block in the own channel.

The program run is interrupted if the synchronism marker from the specified channels is not yet set.

WAITMC Conditional waiting for synchronism marker

Conditional waiting for a synchronism marker from the specified channels.

Exact stop in the previous block and interruption of the program execution will only be generated if the synchronism markers of the specified channels are not yet set.

WAITE Wait for end of program

Waiting for the end of the program in the specified channels.

Programming example

```

; PROG_K1.MPF
; Channel 1: Control program for channels 2, 3, 4

; Program selection in the channels

N10 INIT (2, "Prog_K2")
N20 INIT (3, "Prog_K3")
N30 INIT (4, "Prog_K4")

ZYKLUS:

N50 CLEARM()           ; Clear all synchronism markers

N60 G0 X22
N50 START (2)          ; Program start in channel 2
N60 WAITM (1,2)        ; Waiting for synchronism marker 1 from channel 2

N70 X33
N80 START (3)          ; Program start in channel 3
N100 WAITM (1,3)       ; Waiting for synchronism marker 1 from channel 3

N110 X44
N120 START (4)         ; Program start in channel 4
N130 WAITM (1,4)       ; Waiting for synchronism marker 1 from channel 4

N140 X0

N150 SETM (3)          ; Set synchronism marker 3 (end of cycle)
N160 G4 F0.1           ; Executable block
N120 WAITE (2,3,4)     ; Wait for end of program in channels 2, 3, 4

GOTOB CYCLE

M30

; Channel 2
; PROG_K2.MPF

N10 CLEARM()           ; Clear set synchronism markers (if any)
N20 G1 Y22 F1000
N30 SETM(1)            ; Set synchronism marker 1
N40 G0 Y0
N50 WAITM(3,1)         ; Wait for synchronism marker 3 from channel 1
N50 CLEARM()           ; Clear all synchronism markers in the channel
N60 M30

```

```
; Channel 3
; PROG_K3.MPF

N10 CLEARM() ; Clear set synchronism marker (if any)
N20 G0 Z33
N30 SETM(1) ; Set synchronism marker 1
N40 G0 Z0
N50 WAITM(3,1) ; Wait for synchronism marker 3 from channel 1
N60 CLEARM() ; Clear all synchronism markers in the channel
N70 M30
```

```
; Channel 4
; PROG_K4.MPF

N10 CLEARM() ; Clear set synchronism markers (if any)

N20 POS[A]=44 FA[A]=10000
N30 SETM(1) ; Set synchronism marker 1
N30 POS[A]=0 FA[A]=500
N50 WAITM(3,1) ; Wait for synchronism marker 3 from channel 1
N60 CLEARM() ; Clear all synchronism markers in the channel
N70 M30
```

SETM / CLEARM in synchronized actions

allows the coordination between synchronized action and the execution of the NC program in the own channel. SETM or CLEARM must be programmed in the synchronized action. In the NC program, exclusively the number of the own channel must be specified in the WAITM or WAITMC statement.

Programming example

```
. . .
N10 CLEARM()
N20 G0 X0 Y0 Z0

N30 ID=1 WHEN $AA_IM[Y]> 50 DO SETM(9) ; Set synchronism marker 9 if
; position of Y axis > 50 mm

N30 WEHN $AA_IM[X]> 20 DO POS[Y]=100 FA[Y]=2000
N50 G1 X50 F800

N60 WAITMC (9,1) ; Wait for synchr. marker 9 from the own channel (channel 1)
N70 X0

M30
. . .
```

10.29 Subroutine technique

General

There is no fundamental difference between a main program and a subroutine.

Frequently recurring program sequences are often stored in subroutines. The main program can then call up and run the subroutine at the required point.

The structure of a subroutine is identical to that of a main program. As in the case of main programs, an end of program identifier is inserted in the last block of the programmed sequence. In this context, this indicates a return to the calling program level.

Programming

UP_NAME...	; Subroutine call without param. transfer
UP_NAME(PARA1, PARA2, ...)	; Subroutine call with parameter transfer
PROC	; Subroutine definition for parameter transfer
EXTERN	; External declaration for parameter transfer
VAR	; Parameter transfer with return
CALL	; Indirect subroutine call
MCALL	; Modal subroutine call
SAVE	; Saving of modal functions
P...	; Program repetition
M2	; End of subroutine with CPU output
RET	; End of subroutine without CPU output

Subroutine call

Subroutines are called by their name in the main program or another subroutine. The call statement must be programmed in a separate block.

Example:

```
N10 L12      ; Call subroutine L12
...
N200 L12     ; 2nd call of subroutine L12
...
N466 GRUND  ; Call of subroutine named GRUND
```

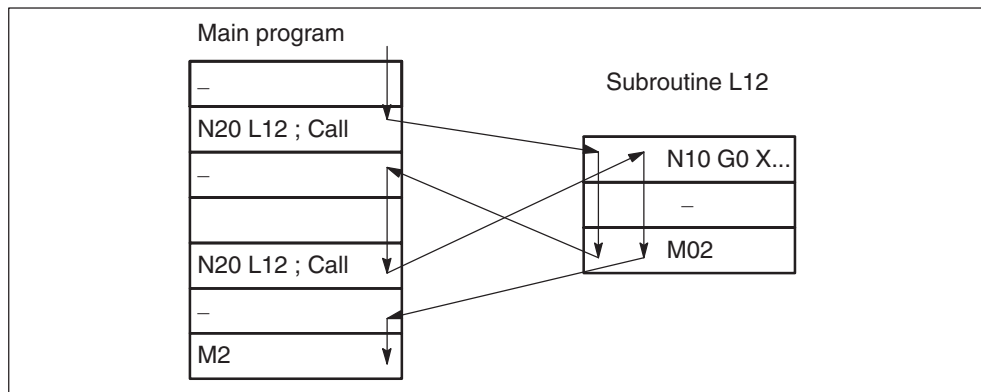


Fig. 10-59 Example of a program run with double subroutine call

Nesting depth

A main program or subroutine can call a further subroutine. This subroutine can call a further subroutine, etc. A total of 12 program levels, including the main program level, are available for such nested calls. That means: A maximum of 11 subroutines can be called from a main program.

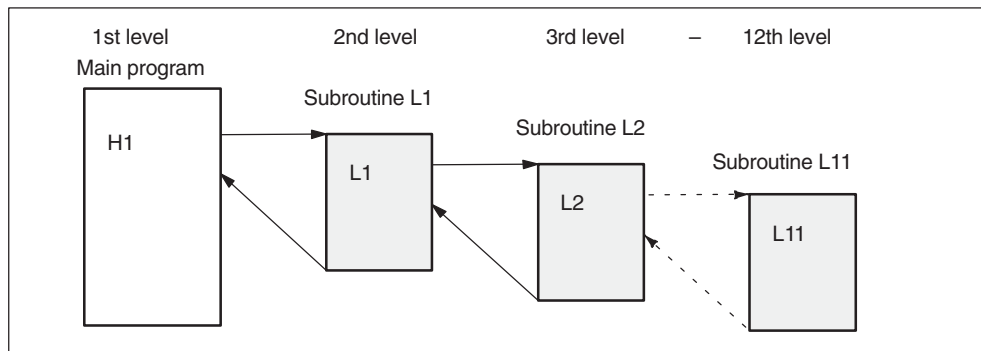


Fig. 10-60 Nesting depth

You can also call subroutines in ASUBs. You must keep an appropriate number of levels free for their execution.

Subroutine call without parameter transfer

For subroutines, address word L... can be used. 31 decimals (integer only) can be used for the value.

Please note: Leading zeros after address L are important.

In addition, a name can be freely selected, observing the following conventions:

- The first two characters must be letters.
- Then letters, digits and underscore are permitted (no blanks or tabulators).
- The maximum length of the name is 25 characters.

Example:

```

...
; Call of two different subroutines
N10 L123                ; Call of L123.SPF
N20 L0123               ; Call of L0123.SPF
...
N50 UP_TEIL1           ; Call of UP_TEIL1.SPF
...

```

Subroutine call with parameter transfer

When calling a subroutine, parameters can be transferred from the main program to the subroutine. In the main program, the parameter transfer can be explained with an EXTERN statement, and in the subroutine, with the PROC statement.

Both the EXTERN and the PROC statement must be programmed in a separate block.

A maximum of 127 parameters can be transferred.

As the transfer parameter, constants or LUD variables of different data types are possible.

Using the VAR statement, it can be additionally agreed for the LUD variable whether in the subroutine modified parameters are returned to the main program.

In the subroutine call, parameters can be omitted; in this case, the transfer value is assigned zero by default. Parameters of the type AXIS and VAR parameters must be transferred completely.

Example:

```

; Main program
N10 DEF INT PAR_H1                ; Definition of an INT variable

;Subroutine explanation:
N20 EXTERN UP_FIX_PAR ( REAL, INT, INT) ; Without parameter transfer
N30 EXTERN UP_VAR_PAR (VAR INT, INT)   ; With parameter transfer
...
N50 PAR_H1=10
N60 UP_FIX_PAR (123.45, PAR_H1, 90)    ; Subroutine call
...                                   ; PAR_H1 = 10
N70 PAR_H1=PAR_H1+10                  ; PAR_H1 = 20
...
N80 UP_VAR_PAR (PAR_H1, 50)           ; Subroutine call
...                                   ; PAR_H1 = 70
N100 M2
; Subroutine UP_FIX_PAR
N10 PROC UP_FIX_PAR (REAL PAR_L1, INT PAR_L2, INT PAR_L3)
...
N20 PAR_L2 = PAR_L2+PAR_L3            ; PAR_L2=100
...
N100 RET

; Subroutine UP_VAR_PAR, PAR_L1 is to be returned
N10 PROC UP_VAR_PAR(VAR INT PAR_L1, INT PAR_L2)
...
N20 PAR_L1 = PAR_L1+PAR_L2           ; PAR_L1=70
...
N30 M2

```

Indirect subroutine call CALL

An indirect subroutine call is possible using the statement CALL.
The subroutine name must be transferred in a variable of the type STRING.
Indirect subroutine call with parameter transfer is not possible.

Example:

```

...
N10 DEF STRING[30] UP_NAME
N20 UP_NAME = "TEIL1"                ; without specifying the domain
...
N30 CALL UP_NAME
...
N40 UP_NAME = "_N_TEIL2_SPF"         ; also possible with specifying
                                      ; the domain
N50 CALL UP_NAME
...

```

Modal subroutine call MCALL

With this statement, the subroutine is recalled automatically after each block containing a path motion and is executed.

Only one MCALL call can be active at a time. Any parameter transfer, as well as the definition and initialization of variables in the subroutine are only carried out during the first call.

The function is disabled with MCALL without specifying the subroutine.

Example:

```
...
N10 G0 X0 Y0
N20 MCALL UP_POS_Z ; Enable modal subroutine call
N30 X10 ; UP_POS_Z is carried out after each motion block
N40 Y10 ;
N50 X20
N60 MCALL ; Disable modal subroutine call
...
```

Program repetition P...

If a subroutine needs to be executed several times in succession, then the number of required passes must be programmed after the subroutine name (in address P) in the block containing the call. A maximum of 9999 passes are possible (P1 to P9999). **P** does not have to be programmed for a single pass.

Example:

```
N10 L123 P3 ; Call L123 with 3 passes
...
N420 L567 ; Call L567 with 1 pass
```

Saving of modal functions SAVE

The statement SAVE will reset modal G functions modified in the subroutine and zero offsets at the end of the subroutine to their initial status.

Save must be written in the subroutine together with the PROC statement.

Example:

```
; Main program
N10 G0 X0
N20 G1 G90 F500 X100
N30 UP_TEIL_A
N40 X33 ; X axis traverses to X=33 with F100
...
M2

; Subroutine UP_TEIL_A with SAVE statement
PROC UP_TEIL_A SAVE
N10 G91 G1 X22 F100
...
RET
```

End of subroutine

M2 or RET can be programmed as the subroutine end:

- **M2**

The subroutine is terminated with an exact stop and execution jumps back to the calling program. M2 is output to the CPU.

- **RET**

Same effect as M2, except that G64 continuous-path mode is not interrupted. RET must be programmed in a separate block. RET is not output to the CPU.

It is possible to alter modally active G functions or R parameters, which are also employed in the calling program, in a subroutine (e.g. G90 to G91). Make sure that all modally active functions and R parameters are set as required for the subsequent program run after the jump back to the calling program.

10.30 Asynchronous subroutines (ASUB)

General

Asynchronous subroutines are special routines which are started by events (signals) in the machining process. In this case, an NC block which is being executed is interrupted. It is possible to resume the NC program at a later stage at the point of interruption.

The FM 357-2 has 8 on-board inputs (inputs 0 to 7) which can be used to trigger an interruption of the running program and start an interrupt routine (ASUB).

An ASUB can also be started by the CPU. Only interrupt no. 8 may be used.

ASUB programming

PROC NAME SAVE

PROC ; Define an ASUB
 NAME ; Name of ASUB
 SAVE ; Restore interruption position and the current machining
 ; status
 REPOS L ; Reposition at interruption point in
 ; main program/subroutine

Call programming

SETINT(n) PRIO=1 NAME

SETINT(n) ; Assign a digital input/interrupt no. (n = 1...8/8)
 ; n = 1...8 corresponds to HW input 0...7
 PRIO = m ; Define priority (m = 1...128, 1 is highest priority)
 NAME ; Name of ASUB
 DISABLE(n) ; Disable ASUB (n = no. of digital input)
 ENABLE(n) ; Activate ASUB (n = no. of digital input)
 CLRINT(n) ; Clear assignment between digital input and NC program

PROC

The name of an ASUB is defined with PROC. An ASUB is programmed in the same way as a subroutine.

Example:

```
PROC LIFT_Z           ; Subroutine name LIFT_Z
N10 G0 Z200          ; NC blocks
...
N20 M02              ; End of subroutine
```

SAVE

If the SAVE command has been used in the definition of an ASUB, the interruption position of the axes is saved automatically.

The current status, modally active G functions and zero offsets of the interrupted NC program take effect again as soon as the ASUB has ended.

This allows the program to be resumed later at the interruption point.

Example:

```
PROC LIFT_Z SAVE     ; SAVE saves the current
                    ; machining status
N10 G0 Z200          ; NC blocks
...
N20 M02              ; Stored machining status is restored again.
```

REPOSL

This function is only available for the FM 357-2 LX, H.

To reposition the axis on the interruption point again, a REPOSL statement must be programmed at the end of the ASUB.

Example:

```
PROC LIFT_Z SAVE
...
N20 REPOSL M02      ; Reposition on interruption point
```

SETINT(n)

Definition of which input must start which ASUB. This statement gives a normal subroutine the status of an ASUB.

If a new ASUB is assigned to an occupied input, the old assignment is automatically deactivated.

Example:

```
N20 SETINT(3) PRIO=1 LIFT_Z ; Assign input 3 to "LIFT_Z"
...
```

PRIO

If your NC program contains several SETINT statements, you must assign a processing priority to the ASUBs. PRIO=1 has highest priority.

The ASUBs are executed successively in the order of the priority when several inputs are active simultaneously.

If new signals are detected during ASUB execution, the associated ASUBs are executed subsequently in order of their priority.

Example:

```
N20 SETINT(3) PRIO=2 LIFT_Z ; "LIFT_Z" with priority 2
...
```

DISABLE(n) / ENABLE(n)

By using the DISABLE command, you can protect NC program sections prior to program interruption. The assignment defined by SETINT is retained, however there is no response to the 0/1 edge change of the interrupt signal. The ENABLE command is used to cancel the DISABLE command. The ASUB is not started until the next 0/1 edge change of the interrupt signal.

Example:

```
N20 SETINT(3) PRIO=2 LIFT_Z ;
N30 ... ; ASUB LIFT_Z possible
N40 ...
N50 DISABLE(3)
N60 ... ; ASUB LIFT_Z disabled
N70 ...
N80 ENABLE (3)
N90 ... ; ASUB LIFT_Z possible
...
```

CLRINT(n)

With this statement or end of program, the assignment between an input and an ASUB is cancelled.

Example:

```

N10 SETINT(3) PRIO=2 LIFT_Z
N20 SETINT(4) PRIO=1 LIFT_X      ;
N30 ...                          ; ASUB LIFT_Z possible
N40 ...
N50 CLRINT(3)
N60 ...                          ; ASUB LIFT_Z canceled
N70 M02                          ; ASUB LIFT_X canceled
    
```

Program levels

There are a total of 12 program levels available. Whichever subroutine levels are not required by ASUBs are freely available to the NC programmer.

Of the 12 program levels, four should be reserved for ASUBs.

Processing sequence

The following diagram shows the fundamental sequence in which an ASUB is processed

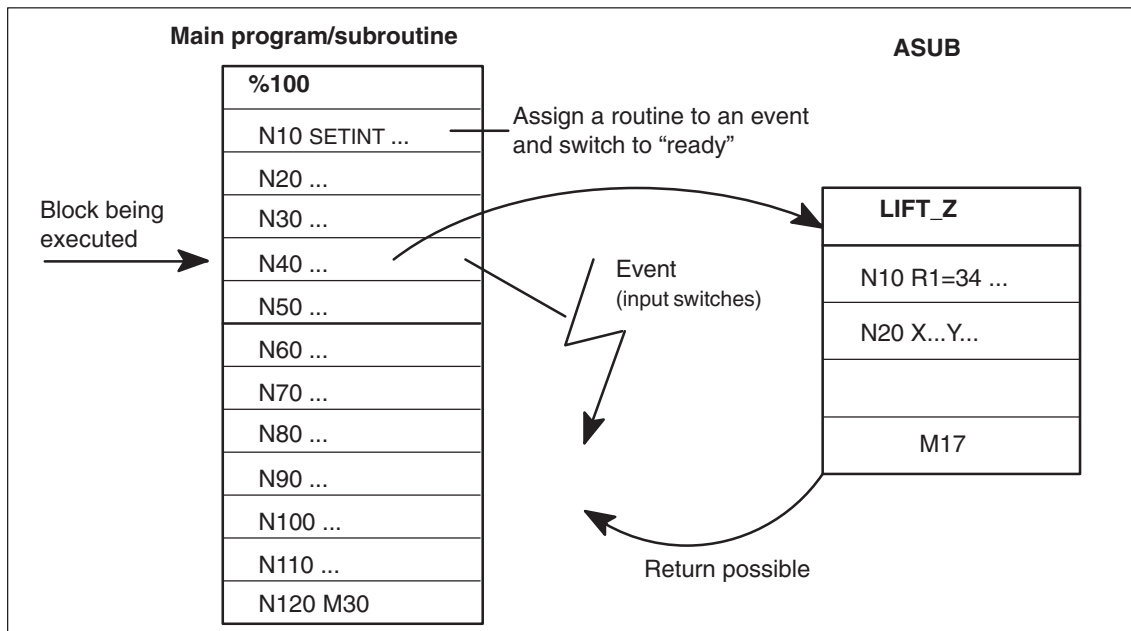


Fig. 10-61 Working with ASUBs

10.31 Activating machine data (NEWCONF)

General

The statement NEWCONF is used to activate machine data of the activation stage NEW_CONFIG.

When the function NEWCONF is executed, an implicit preprocessing stop is carried out.

This statement must be used in conjunction with changing machine data from the NC program.

Programming

NEWCONF ; Activate machine data of the activation stage NEW_CONFIG

In the parameterization tool, machine data area, amongst other features, the activation stage of the machine data is displayed in the table view.

Example:

N10 ...

N20 \$MA_MAX_AX_ACCEL[1]= 2 ; Max. acceleration of the 1st axis

N30 NEWCONF

10.32 Synchronized actions

General

Synchronized actions provide the user with the opportunity of initiating actions independently of the NC block processing operation. The point in time at which these actions are activated can be defined by a condition. Synchronized actions are executed in the interpolation cycle (IPO cycle).

The scope of available functions has significantly increased as compared to SW version 1.2.

Programming

A synchronized action comprises the following elements:

- ID number (validity)
- Action duration, condition
- Action

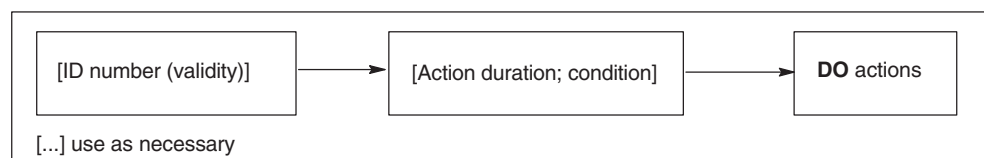


Fig. 10-62 Structure of motion synchronous actions

A maximum of 320 storage elements are available. A synchronized action needs at least 4 of these elements.

A synchronized action must be programmed **on its own** in a block and takes effect in or from the next output block (e.g. block containing G01, G02, G04, auxiliary function output).

Validity

The following synchronized actions are available:

- **without ID number**

The synchronized action takes effect only in the next executable NC block in Automatic mode.

- **ID = n (modal synchronized action) n = 1...255**

The synchronized action acts modally from the next executable block in the active NC program. If you program the same ID number twice, the second synchronized action overwrites the first.

- **IDS = n (static synchronized action) n = 1...255**

This synchronized action takes effect from the next executable block and is modally active beyond the active NC program in every operating mode. If you program the same IDS number twice, the second synchronized action overwrites the first. This action is not affected by the end of program.

Static synchronized actions in any operating mode are available for the FM 357-2LX only.

Response at the end of the program

- **without ID number and ID = n (modal synchronized action) n = 1...255**

If an action is active at the end of the program, M2 or M30 is output, and the status "Program is running" remains present until the action is completed. If an active MOV or POS motion is started from an action, the FM message "Wait: for positioning axis" is output.

- **IDS = n (static synchronized action) n = 1...255**

With the end of program, M2 or M30 and the status "Program canceled" are output, irrespective of an active action.

Response in case of Reset (user DB, "FMx", DBX108.7+n)

- **without ID number and ID = n (modalsynchronized action) n = 1...255**

These synchronized actions will be deleted.

- **IDS = n (static synchronized action) n = 1...255**

These synchronized actions will be reset.

Processing sequence

Modal and static synchronized actions are processed in the order of their ID number, i.e. ID=1 before ID=2. After modal and static synchronized actions have been processed, the non-modal actions are processed in the order in which they were programmed.

Scanning/execution frequency

These statements determine how often the condition is scanned and the associated actions executed.

- **No scanning/execution frequency**

The action is **always executed cyclically**.

- **WHEN**

If this condition is true, the action is executed **once**. The synchronized action is then ended.

- **WHENEVER**

While the condition is true, the action is executed **cyclically**.

- **FROM**

If the condition is fulfilled **once**, the action is executed **cyclically**.

- **EVERY**

Every time the condition is fulfilled, the action is executed **once**.

Condition

The execution of an action can be made dependent on a condition (logical expression). The conditions are checked in the IPO cycle.

Structure of a condition: Comparison <Boolean operator> comparison

Comparison: Expression <compare operator> expression

Expression: Operand <Operator> operand ...

Boolean operators: E.g. NOT

Compare operator: E.g. ==

Operand: System variable or value

System variable: E.g. \$AA_IW[X] (actual value of X axis)

The available functionality is summarized in Tables 10-10 and 10-11.

Notice

The left-hand side of a comparison is **reread** in every IPO cycle.

The right-hand side is formed once while the block is being preprocessed.

If the condition on the right-hand side must also be read cyclically in the IPO cycle, an **additional \$ sign** must be inserted before the system variable.

Example:

Comparison between cycle actual value of X axis and expression calculated during block preprocessing:

```
N10 ... $AA_IW[X]>R5+100
```

Comparison between cyclic actual value of X axis and cyclic actual value of Y axis:

```
N10 ... $AA_IW[X]>$AA_IW[Y]
```

Logic operations involving comparisons:

```
N10 ... ($AA_IW[X]>100) OR ($AA_IW[X]<COS ($AA_IW[Y]))
```

For further details, please see "Arithmetic operations in synchronized actions".

DO action

When the condition is fulfilled, the actions (max. 16) programmed after **DO** are executed.

System variables can also be read and written in the action component.

Example:

Write value from MARKER1 to digital output 11:

```
... DO $A_OUT[11]=$AC_MARKER[1]
```

Change the velocity of the X axis as a function of the Y axis actual position:

```
... DO $AA_OVR[X]=$R10*$AA_IM[Y]-$R11
```

CANCEL(n)

You can cancel modal or static synchronized actions with this statement. A currently active action is executed to the end (e.g. positioning motion). **CANCEL()** is a normal statement and cannot be written as an action.

System variables $\$P_ACTID[n]$ supply the status (active/not active) of a synchronized action, n is the ID No.

Actions within synchronized actions

General

Actions which are started from synchronized actions must also be ended in synchronized actions. Vice versa, functions which are started in an NC block must also be ended in an NC block.

Example:

```
N10 TRAILON(Y,X)
N20 WHEN TRUE DO TRAILON(Z,A)
N30 H22
...
;right:
N100 TRAILOF (Y,X)
N110 WHEN TRUE DO TRAILOF(Z,A)
N120 M21
...
;wrong:
N100 TRAILOF(Z,A)
N110 WHEN TRUE DO TRAILOF(Y,X)
N120 M21
```

M and H functions

Up to 5 M functions and 3 H functions can be output in a machining block as synchronous commands.

When a condition is fulfilled, the auxiliary functions are output immediately to the CPU in the IPO cycle. The output timing set via machine data is irrelevant in this case.

The CPU acknowledges an auxiliary function after a complete CPU user cycle. The block change is not affected by the acknowledgement.

An auxiliary function may **not** be output cyclically, i.e. it can be programmed only with vocabulary word "WHEN" or "EVERY" or as a non-modal function.

Predefined M commands are not permissible.

Example: Output of M functions as a function of an actual position

```
N10 WHEN $AA_IW[X]>100 DO M70 M72
N15 G1 X200 F5000
```

If the actual value in the WCS of the X axis exceeds 100 mm, M functions M70 and M72 are output once.

RDISABLE Programmed read-in disable

This statement interrupts block processing if the associated condition is fulfilled. The system processes only the programmed motion synchronous actions, preprocessing of subsequent blocks continues.

If the condition for the RDISABLE statement is no longer fulfilled, the read-in disable is cancelled. At the beginning of the block containing RDISABLE, an exact stop is initiated irrespective of whether or not the read-in disable is active.

Example: Quick program start

```
N10 WHENEVER $A_IN[10]==FALSE DO RDISABLE
N15 G0 X100
```

The block after N15 is not processed as long as the condition for RDISABLE is fulfilled.

On the 0/1 edge at digital input 10, the block after N15 and all subsequent blocks are enabled for execution. The synchronized action is thus ended.

DELDTG Delete distance to go with preprocessing stop for path axes
DELDTG(axis) Delete distance to go with preprocessing stop for positioning axes

The DELDTG statement causes a preprocessing stop in the next output block. If the condition for DELDTG is fulfilled, the distance to go is deleted and the preprocessing stop cancelled.

Modal functions such as continuous-path mode or rounding are not permitted and/or interrupted.

The path or axial distance to go to the block end can be read in system variables \$AC_DELT and \$AA_DELT[axis] after the distance to go has been deleted.

DELDTG and DELDTG(axis) may be programmed only with statements “WHEN” or “EVERY” and as non-modal commands (without ID number).

Example: Delete distance to go as a function of actual position of axes Y and X

```
N10 G0 X0 Y100
N20 WHEN $AA_IW[X]>$$AA_IW[Y] DO DELDTG(X)
N30 POS[X]=100 FA[X]=5000 POS[Y]=0 FA[X]=5000
```

The actual values of the X and Y axes are read and evaluated in the IPO cycle. If the actual value of the X axis exceeds the actual value of the Y axis, the X axis is stopped and its distance to go deleted.

POS[axis] Positioning motion to end position
MOV[axis] Positioning motion without end position

These actions can be programmed to position axes asynchronously to the NC program. The positioning motion itself does not affect the NC program.

An axis may not be moved from the NC program and a synchronized action at the same time. It may be moved by these two sources in succession, but delays may occur while the axis is changed over.

The axial feedrate must be programmed after statement FA[axis].

Active software limits are active. Working area limitations set in the NC program (WALIMON/WAILMOF) are **not** active.

POS[axis] = position

The axis traverses towards a preset end position. The end position is specified as an absolute or relative value (see Section 10.2.3).

You can enter a new position “on the fly” while the axis is moving.

Active zero offsets and tool offsets are applied.

To enable the axis, WAITP(axis) must be programmed once before POS[axis].

Example: On-the-fly input of a new end position

```
N10 ID=1 EVERY $A_IN[9]==TRUE DO POS[Y]=100 FA[Y]=2000
N20 ID=2 EVERY $A_IN[10]==TRUE DO POS[Y]=200
```

When digital input 9 switches from 0 to 1, the Y axis commences its positioning motion to end position 100. If input 10 switches from 0 to 1, a new end position 200 for Y is set on the fly.

MOV[axis] = value

An axis is traversed infinitely in the programmed direction. An end position can be preset on the fly or the axis stopped.

Value = 1: Axis motion in positive direction
 Value = -1: Axis motion in negative direction
 Value = 0: Stop axis motion

MOV[axis] = 0 must be used only for an axis:

- that is moved by means of MOV[axis] = 1/-1 or POS[axis] = ...
- or that is enabled by WAITP(axis).

Example: On-the-fly changeover between MOV and POS

```
N10 ID=1 WHEN $AA_STAT[X]<>1 DO MOV[X]=1 FA[X]=1000
N20 ID=2 WHEN $A_IN[10] == 1 DO POS[X]=100
```

The X axis starts to move in the positive direction (ID=1). If input 10 switches to 1, the axis is positioned at 100 while it is still in motion. These actions are performed only once.

PRESETON (MA, IW) Set actual value

MA – machine axis
 IW – actual value

PRESETON can be programmed to reset the control zero in the machine coordinate system, i.e. a new value is assigned to the current axis position. The function can be executed while the axis is moving.

PRESETON can be executed from synchronized actions for

- axes which are being positioned from synchronized actions (POS, MOV)
- modulo rotary axes which are being traversed by the NC program

Example: Set actual value while axis is moving

```
N10 ID=1 EVERY $A_IN[9]==TRUE DO POS[X]=100 FA[X]=2000
N20 ID=1 EVERY ($A_IN[10]==TRUE) AND ($AA_STAT[X]==1)
      DO $AC_PARAM[1]=$AA_IW[X]+5 PRESETON(X1, $AC_PARAM[1])
```

On the 0/1 edge change at digital input 9, the X axis commences its positioning movement. If the X axis is moving, its current actual position is shifted by +5 mm every time digital input 10 switches from 0 to 1.

ACC[axis] ;Programmable acceleration

Using the programmable acceleration, the axis acceleration set via the parameterization can be changed to a synchronized action. To do so, the motion must also be programmed as a synchronized action. In this case, the acceleration active at the beginning of the motion becomes active.

\$AC_PATHACC ;Programmable acceleration for external events

If the path acceleration has been reduced using ACC, very long braking paths can result, e.g. in the case of external events, such as “Stop” or “Override = 0”. The system variable \$AC_PATHACC can be used to increase the path acceleration in braking or acceleration processes initiated by external events in synchronized actions.

Values less than the effective path acceleration are ignored.

The maximum value is limited to the double of the axis acceleration.

Subroutines as actions

The function is available for the FM 357-2LX.

You can call a subroutine as an action in static or modal synchronized actions. This subroutine, however, may contain only those functions which may also be programmed as individual actions. Several subroutines may be started and active simultaneously.

The blocks are processed sequentially in the IPO cycle. Simple actions such as the setting of a digital input require only one IPO cycle while positioning motions will require several cycles. Only one axis motion may be programmed in each block.

Once a subroutine has started, it will be processed to the end irrespective of the associated condition. At the program end, the program can be restarted if the associated condition is fulfilled.

Example: Several positioning motions in subroutines

```
ID=1 EVERY $A_IN[9]==TRUE DO POS_X
ID=2 EVERY $AA_IW[X]>=100 DO POS_Y
ID =3 WHENEVER ABS($AA_IW[X]-$AA_IW[Y])<20 DO $AA_OVR[Y]=50
```

POS_X

```
N10 POS[X]=100 FA[X]=1000
N20 M55
N30 POS[X]=0
N40 M2
```

POS_Y

```
N10 POS[Y]=50 FA[Y]=2000
N20 M56
N30 POS[Y]=100
N40 M2
```

Whenever input 9 switches from 0 to 1, subprogram POS_X is started (ID=1). When the actual position of the X axis reaches or exceeds 100, POS_Y (ID=2) is started. If the distance between the X and Y axes drops below 20 (safety clearance), ID=3 reduces the feedrate of the Y axis to 50 %.

TRAILON (slave axis, master axis, coupling factor) ; Activate coupled motion
TRAILOF (slave axis, master axis) ; Deactivate coupled motion

The master axis may already be in motion when the coupled motion function is activated. In this case, the slave axis is accelerated to the setpoint velocity.

It is possible to switch over on the fly between a positioning motion and axis motions resulting from coupled motion on the condition that both motions are actions from synchronized actions.

For further details about the coupled motion function, please refer to Section 9.16.1.

Example: Activating/deactivating coupled motion on the fly

```
N10 WHEN $AA_STAT[X]<>1 DO MOV[X]=1 FA[X]=1000
N20 ID=2 EVERY $AA_IW[X]>100 DO TRAILON(Y,X,1) POS[Z]=0 FA[Z]=100
N30 ID=3 EVERY $A_IN[10]==TRUE DO POS[Z]=50
N40 ID=4 EVERY ($AA_IW[X]>200)AND($AA_COUP_ACT[Y]==0) DO POS[Y]=0
N50 ID=5 EVERY $AA_IW[X]>200 DO TRAILOF(Y,X)
N60 ID=6 EVERY $A_IN[9]==1 DO PRESETON (X1,0)
```

...

The X axis (conveyor belt) is traversing as an infinite axis in the positive direction. A sensor at digital input 9 switches if a part is detected on the conveyor belt. The actual position of the X axis is then set to 0 (ID=6). When position X100 is reached in relation to the new zero point, the Y axis is coupled to the X axis and axis Z traverses to gripper position 0 (ID=2). The coupled Z and Y axes traverse in parallel to the X axis. When the gripper is holding the part, input 10 switches to 1 and the Z axis is then positioned at 50 (ID=3). The coupling is dissolved at position X200 (ID=5) and the Y axis traverses back to position 0 (ID=4).

Note:

To be able to traverse an axis coupled via a synchronized action in the NC program again, first the function TRAILOF must be called.

It must be ensured that TRAILOF has been completed before the relevant axis is requested.

This is not the case in the following example:

```
N50 WHEN TRUE DO TRAILOF(Y, X)
N60 Y100
```

In this case, the axis will not be enabled in time, since the modally active synchronized action becomes active synchronously with TRAILOF.

To avoid such situations, it is recommended to proceed as follows:

...

```
N50 WHEN TRUE DO TRAILOF(X, Y)
N55 WAITP(Y)
N60 Y100
```

LEADON (slave axis, master axis, curve table) ; Activation of coupling
LEADOF (slave axis, master axis) ; Deactivate coupling

Master value couplings are activated and deactivated from synchronized actions independently of the NC program and is thus not restricted by block limits. The control initiates a synchronization operation (see Section 9.16.3) to set up the coupling.

It is possible to switch over on the fly between positioning motions and movements resulting from an axis coupling started in a synchronized action.

The interrelationship between the master and slave values defined in the curve table can be used for calculations in the condition and action components.

For further details about the master value coupling, please refer to Section 9.16.3.

CTABDEF() ; Start curve table definition
CTABEND() ; End curve table definition
CTAB() ; Read out slave value for a master value
CTABINV() ; Read out master value for a slave

System variable **\$AA_SYNCH**[axis] indicates the synchronization status of the slave axis.

Example: See MEAWA

MEAWA[axis]=(mode, trigger event_1...4) ; Axial measurement without
deletion of distance to go

The function is available for the FM 357-2LX.

For details about programming and operating mode of the Measurement function, please see Sections 10.10 and 9.17.

While the measuring function in the NC program is a non-modal function, the measurement function from synchronized actions can be freely activated and deactivated. Using a static synchronized action, for example, you can also take measurements in JOG mode.

Only one measurement job may be programmed for each axis. A measurement task started from the NC program (not from a synchronized action) cannot be affected by a synchronized action.

The measurement result is stored in system variables.

\$AA_MM1...4[axis] ; Measured value of trigger event 1...4
; in machine coordinate system

Example: Master value coupling and measurement from synchronized actions

```

N10 CTABDEF(Y,X,1,0)      ; Start curve table
N20 G1 X0 Y0              ; Starting point: LW 0, FW 0
N30 X20 Y10              ; LW 0...20 , FW 0...10
N40 X40 Y40              ; LW 20...40 , FW 10...40
N50 X60 Y70              ; LW 40...60 , FW 40...70
N60 X80 Y80              ; LW 60...80 , FW 70...80
N70 CTABEND              ; End curve table
                          ; LW – master value, FW – slave value
N80 $AC_PARAM[1]=0       ; PRESETON value Y axis
N90 $AC_MARKER[1]=0      ; Marker

; Master value coupling
N100 WHEN $AA_STAT[X]<>1 DO PRESETON(X1,-20) MOV[X]=1 FA[X]=10000
N110 ID=1 EVERY $AA_IW[X]>=100 DO PRESETON(X1,-20)
N120 ID=2 EVERY $AA_IW[X]>=0 DO LEADON(Y,X,1)
N130 ID=3 EVERY $AA_IW[X]>=80 DO LEADOF(Y,X)
      PRESETON(Y1,$AC_PARAM[1]) M50

; Measurement
N150 ID=4 EVERY ($AA_MEAAct[Y]==0)AND($AC_MARKER[1]==1)
      DO $AC_MARKER[1]=0 $AC_PARAM[1]=50-$AA_MM1[Y]
N140 ID=5 EVERY $AC_MARKER[1] == 0
      DO MEAWA[Y]=(2,1) $AC_MARKER[1]=1

```

The X axis is moving a conveyor belt continuously. The actual value is reset cyclically to -20 at position 100 (ID=1).

The master value coupling is activated in the X0 to X80 range (ID=2 and ID=3).

The slave axis Y then moves as specified in the curve table defined between N10 and N70.

The Y axis is transporting a strip of foil into which the part arriving on the conveyor belt must be welded at position X80. M50 (ID=3) starts the welding cycle which is controlled by the CPU.

Notches in the foil initiate the measurement in Y by means of a sensor (ID=5).

The difference between the measured value and the expected position of the notch (Y50) is calculated when the Y axis actual position is reset (ID=3).

X should be a modulo axis for high-speed transport tasks. There is then no need to set the actual value in the IPO cycle (ID=1) while the axis is moving.

LOCK (ID No., ID No., ...) ; Disable synchronized action
UNLOCK (ID No., ID No., ...) ; Enable synchronized action
RESET (ID No., ID No., ...) ; Reset synchronized action

A synchronized action is disabled with LOCK. An action currently in progress or the active block in the subroutine are completed to the end.

UNLOCK cancels the disable command, processing of the associated actions continues again depending on the relevant conditions.

RESET resets a synchronized action. The actions or the subroutine are aborted. The synchronized action is then treated like a new statement.

Interface signal "Disable synchronized action" can be used by the CPU to disable synchronized actions between ID No. 1 and 8:

- User DB, "FMx", DBX580.0...7 for channel 1
- User DB, "FMx", DBX586.0...7 for channel 2
- User DB, "FMx", DBX592.0...7 for channel 3
- User DB, "FMx", DBX598.0...7 for channel 4

Timer variable \$AC_TIMER[n]

The system variable \$AC_TIMER[n] can be used to start actions after a defined waiting time. 32 timer variables (indices 1 to 32) are provided. Timer variables are of the data type REAL, unit of measurement seconds.

Resetting and starting a timer

Assigning a timer an initial value greater than or equal to zero will either reset or start a timer.

Stopping a timer

A timer is stopped by assigning -1.
 End of program or reset will stop and reset the timer.

Reading a timer

The current time value can be read both with the timer running and stopped.
 After the timer has been stopped, the last current time value is kept and can be continued to be read.

Example:

0.7 seconds after setting the digital input 5, the X axis is traversed to position 100.

```
WHEN $A_IN[5]==TRUE DO $AC_TIMER[1]=0 ; Reset and start timer
WHEN $AC_TIMER[1] >=0.7 DO POS[X]=100 $AC_TIMER[1]=-1
                                ; Start POS and stop timer
```

Arithmetic operations in synchronized actions

Complex calculations can be performed in the condition and statement components of synchronized actions (see Tables 10-10 and 10-11).

The calculations are performed in the IPO cycle. Each operand requires one element. System variable \$AC_SYNA_MEM indicates the number of free elements, a maximum of 320 elements is available.

Only system variables of one and the same data type can be used in the expression; no implied type conversion is carried out.

Example:

```
DO $R12 = $AC_PARAM[1] ; Permitted REAL, REAL  
DO $R12 = $AC_MARKER[2] ; Not permitted REAL, INT
```

Parenthesizing of expressions is permitted, the “division and multiplication before addition and subtraction” rule applies. Indexing is possible, system variables may be used as indices.

Type conversion

For converting the type from REAL to INT and vice versa, you can use the following statements:

```
INT RTOI (REAL) ; Conversion from REAL to INT  
REAL ITOR (INT) ; Conversion from INT to REAL
```

Example:

```
$R10 = 1234.567  
WHEN TRUE DO $AC_MARKER[1] = RTOI($R10) ; $AC_MARKER[1]=1234
```

The following operators may be used in synchronized actions.

Table 10-10 Operators in synchronized actions

Operator	Meaning
Basic arithmetic operations	
+	Addition
-	Subtraction
*	Multiplication
/	Division
Functions	
SIN()	Sinus
COS()	Cosine
TAN()	Tangent
SQRT()	Square root
POT()	Square
ABS()	Absolute value
TRUNC()	Integer component (truncate)
Compare operators	
==	Equal
<>	Not equal to
>	Greater than
<	Less than
> =	Greater or equal to
< =	Less than or equal to
Boolean operators	
NOT	NOT
AND	AND
OR	OR
XOR	Exclusive OR
Bit-serial operators	
B_NOT	Bit-serially negated
B_AND	Bit-serial AND
B_OR	Bit-serial OR
B_XOR	Bit-serial exclusive OR

You may use the following system variables for synchronized actions.

Table 10-11 System variables

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
User variables				
\$Rn	Arithmetic parameter in static memory	r / w	r / w	REAL
\$AC_MARKER[n] n = 0...7	Marker variable, counter	r / w	r / w	INT
\$AC_PARAM[n] n = 0...49	Arithmetic parameter in dynamic memory	r / w	r / w	REAL
Digital inputs/outputs				
\$A_IN[n]	Digital input	r	r	BOOL
\$A_OUT[n]	Digital output	r / w	r / w	BOOL
\$A_INA[n]	Analog input	r	r	REAL
\$A_OUTA[n]	Analog output	r / w	r / w	REAL
Timers				
\$A_YEAR	Current system time year	r	r	INT
\$A_MONTH	Current system time month	r	r	INT
\$A_DAY	Current system time day	r	r	INT
\$A_HOUR	Current system time hour	r	r	INT
\$A_MINUTE	Current system time minute	r	r	INT
\$A_SECOND	Current system time second	r	r	INT
\$A_MSECOND	Current system time millisecond	r	r	INT
\$AC_TIME	Time from start of block in seconds	r	r	REAL
\$AC_TIMEC	Time from block beginning in IPO cycles	r	r	REAL
\$SAC_TIMER[n]	Timer variable; n number of the timer variables (1...32)	r/w	r/w	REAL
Measurement				
\$AA_MEAAct[axis]	Status of axial measurement 0: Measurement task for axis terminated/not active 1: Measurement task for axis active	r	r	BOOL
\$AC_MEA[n] n: Probe 1 or 2	Status of measurement job (MEAS, MEAW) 0: Meas. job conditions not fulfilled 1: Measurement job conditions fulfilled	r		INT
\$A_PROBE[n] n: Probe 1 or 2	Probe status 0: Probe not deflected 1: Probe deflected	r	r	BOOL

r = read, w = write

Table 10-11 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AA_MM[axis]	Measured value in MCS with MEAS	r	r	REAL
\$AA_MW[axis]	Measured value in WCS with MEAS	r	r	REAL
\$AA_MMi[axis]	Measured value in MCS with MEASA i: Trigger event 1...4	r	r	REAL
\$AA_MWi[axis]	Measured value in WCS with MEASA i: Trigger event 1...4	r	r	REAL
\$VA_IM[axis]	Measured encoder actual value in MCS	r	r	REAL
\$AA_ENC_ACTIVE[axis]	Validity of encoder actual values	r	r	BOOL
Travel to fixed stop				
\$AA_FXS[axis]	Status of travel to fixed stop 0: Axis is not at stop 1: Fixed stop has been approached successfully (axis is inside monitoring window) 2: Approach to fixed stop has failed (axis is not positioned at stop)	r	r	INT
Path distances				
\$AC_PATHN	Normalized path parameter (0: start of block, 1: end of block)		r	REAL
\$AC_PLTBB	Path distance from start of block in the MCS		r	REAL
\$AC_PLTEB	Path to end of block in the MCS		r	REAL
\$AC_DTBW	Distance from start of block in the WCS		r	REAL
\$AC_DTBB	Distance from start of block in the MCS		r	REAL
\$AC_DTEW	Distance from end of block in the WCS		r	REAL
\$AC_DTEB	Distance from end of block in the MCS		r	REAL
\$AC_DELT	Delete distance to go for path after DELDTG in WCS	r	r	REAL
Axial paths (valid for positioning and synchronous axes)				
\$AA_DTBW[axis]	Axial path from block beginning in WCS		r	REAL
\$AA_DTBB[axis]	Axial path from block beginning in MCS		r	REAL
\$AA_DTEB[axis]	Axial path to end of motion in MCS		r	REAL
\$AA_DTEW[axis]	Axial path to end of motion in WCS		r	REAL
\$AA_DELT[axis]	Axial distance to go after DELDTG in WCS	r	r	REAL
Positions				
\$AA_IW[axis]	Actual position of axis in the WCS	r	r	REAL

r = read, w = write

Table 10-11 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AA_IM[axis]	Actual position of axis in the MCS (interpolator setpoints)	r	r	REAL
Software end position				
\$AA_SOFTENDP[X]	Software end position, positive direction	r		REAL
\$AA_SOFTENDN[X]	Software end position, negative direction	r		REAL
Oscillation				
\$SA_OSCILL_REVERSE_POS1[axis]	Position of reversal point 1	r	r	REAL
\$SA_OSCILL_REVERSE_POS2[axis]	Position of reversal point 2	r	r	REAL
Path velocities				
\$AC_VACTB	Path velocity in MCS		r	REAL
\$AC_VACTW	Path velocity in WCS		r	REAL
\$AC_VC	Additive path feed override (must be rewritten in every IPO cycle or the value is set to 0)		r / w	REAL
\$AC_OVR	Path override factor (must be rewritten in every IPO cycle or the value is set to 100%)		r / w	REAL
Axial velocities (valid for positioning axes)				
\$AA_VACTB[axis]	Axis velocity, setpoint in MCS		r	REAL
\$AA_VACTW[axis]	Axis velocity, setpoint in WCS		r	REAL
\$VA_VACTM[axis]	Axis velocity, actual value in MCS		r	REAL
\$AA_VC[axis]	Additive axial feedrate override		r / w	REAL
\$AA_OVR[axis]	Axial override factor (must be rewritten in every IPO cycle or value is set to 100 %)		r / w	REAL
Acceleration				
\$AC_PATHACC	Programmable acceleration for external events	r / w	r / w	REAL
Master value coupling				
\$AA_LEAD_SP[axis]	Position of simulated master value	r / w	r / w	REAL
\$AA_LEAD_SV[axis]	Velocity of simulated master value	r	r	REAL
\$AA_LEAD_P[axis]	Position of real master value (corresponds to encoder actual value)	r	r	REAL
\$AA_LEAD_V[axis]	Velocity of real master value	r	r	REAL
\$AA_LEAD_P_TURN[axis]	Modulo position	r	r	REAL

r = read, w = write

Table 10-11 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AA_SYNC[axis]	Coupling status of slave axis 0: Not synchronized 1: Synchronism coarse 2: Synchronism fine 3: Synchronism coarse and fine	r	r	INT
\$AA_IN_SYNC[axis]	Synchronization process of the following axis 0: No synchronization 1: Synchronization running	r	r	INT
\$AA_COUP_ACT[axis]	Type of axis coupling of slave axis 0: Not coupled 3: Axis is corrected tangentially 4: res. 8: Slave axis 16: Master axis	r	r	INT
\$SA_LEAD_OFF-SET_IN_POS[FA]	Master axis position offset	r / w		REAL
\$SA_LEAD_SCALE_IN_POS [FA]	Scaling for master axis position	r / w		REAL
\$SA_LEAD_OFF-SET_OUT_POS[FA]	Slave axis position offset	r / w		REAL
\$SA_LEAD_SCALE_OUT_POS[FA]	Scaling for slave axis position	r / w		REAL
\$P_CTABDEF	Program section for curve table definition 0: No curve table definition 1: Curve table definition	r		BOOL
Overlaid motion				
\$AA_OFF[axis]	Overlaid motion		r / w	REAL
\$AA_OFF_VAL[Achse]	Total paths of the overlaid motions	r	r	REAL
\$AA_OFF_LIMIT[axis]	Limit for overlaid movement 0: Not reached 1: Reached in positive direction 2: Reached in negative direction	r	r	INT
CPU variables (user DB "FMx", DBW22 and DBW34)				
\$A_DBW[0] \$A_DBW[2]	Data word from CPU, FM can read Data word to CPU, FM can write (freely programmable by user)		r w	INT
Trace				
\$AA_SCTRACE[axis]	Generation of an IPO event (trigger event)	r / w	r / w	BOOL

r = read, w = write

Table 10-11 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
States				
\$AC_STAT	Current FM status 0: reset/ 1: interrupted 2: Active		r	INT
\$AC_PROG	Current status of NC program 0: Program reset/aborted 1: Program stopped 2: Program running 3: Program waiting		r	INT
\$AC_IPO_BUF	Number of preprocessed blocks	r	r	INT
\$AC_SYNA_MEM	Number of available elements for synchronized actions	r	r	INT
\$AA_STAT[axis]	Axis status 0: No status available 1: Traverse motion active 2: Axis has reached end of IPO 3: Axis in position (coarse target range) 4: Axis in position (fine target range)		r	INT
\$AA_TYP[axis]	Axis type 0: Axis in a different channel 1: Channel axis (path or positioning motion) 2: Neutral axis 3: CPU axis 4: Reciprocating axis 5: Neutral axis currently traversed in JOG mode 6: Master value-coupled slave axis 7: Coupled motion of slave axis 8: Command axis (MOV, POS from synchronized action)	r	r	INT
\$AA_FXS[axis]	Status of travel to fixed stop 0: Fixed stop not reached 1: Fixed stop reached 2: Error in approach	r	r	INT
\$AC_PRESET[X]	Last preset value defined	r		REAL
\$P_ACTID[ID No.]	Status of synchronized action 0: not active 1: active	r	r	INT
\$AA_MOTENDA[axis]	1: Block change "Exact stop fine" 2: Block change "Exact stop coarse" 3: Block change "IPO end" 4: Block change "Braking ramp"	r	r	INT

r = read, w = write

Table 10-11 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AC_PRESET[X]	Last specified PRESET value	r		REAL
\$MC_CHAN_NAME	"CHAN1"... "CHAN4"	r	r	STRING
\$P_ACTID[ID-Nr.]	Status of the synchronized action 0: not active 1: active	r	r	INT
\$P_PROG_EVENT	Request call event 1: Start 2: End of program 3: Reset 4: FM restart	r	r	INT
Programming:				
\$P_F	Last programmed path feed F	r		REAL
\$P_FA[X]	Last programmed positioning axis feed	r		REAL
\$P_EP[X]	Last programmed setpoint (end point)	r		REAL
\$P_GG[n]	Current G function of a G group, n... name of G group	r		INT
\$PI	Circle constant P1, permanent value PI= 3,1415927	r		REAL
Electronic gear				
\$AA_EG_TYPE[FA,LA]	Kind of coupling 0: Actual-value coupling 1: Setpoint coupling	r	r	INT
\$AA_EG_NUMERA[FA,LA]	Numerator of coupling factor Number of curve table if denominator = 0	r	r	REAL
\$AA_EG_DENOM[FA,LA]	Numerator of coupling factor	r	r	REAL
\$AA_EG_SYN[FA,LA]	Synchronized position for the specified master axes	r	r	REAL
\$AA_EG_SYNFA[FA,LA]	Synchronized position for the specified following axis	r	r	REAL
\$AA_EG_BC[FA]	Block change mode when activating the EG for the specified following axis	r	r	STRING
\$P_EG_NUM_LA[FA]	Number of the master axes defined for the specified following axis	r	r	INT
\$AA_EG_AX[n, FA]	Axis name of the nth following axis	r	r	AXIS

r = read, w = write

Table 10-11 System variables, continued

System variable	Meaning	Access to NC programs	Access to synchronized actions	Type
\$AA_EG_ACTIVE[FA,LA]	Status of the electronic gear with reference to the specified master axis and following axis 0: disabled 1: enabled	r	r	BOOL
\$VA_EG_SYNCDIFF[FA]	Amount of the synchronism difference with reference to the specified following axis	r	r	REAL
\$VA_EG_SYNCDIFF_S[FA]	Synchronism difference with reference to the specified following axis	r	r	REAL
FIFO memory				
\$AC_FIFO[n...]	System variable for access to FIFO range	r / w	r / w	REAL

r = read, w = write

Execution of a synchronized action

Synchronized actions are executed in the IPO cycle as the relevant block is being processed. If several synchronized actions are simultaneously active, computing time required in the IPO cycle increases. If the permissible time is exceeded, the program is aborted and an error message output (error no. 4240). The following Figure illustrates the principle of synchronized actions.

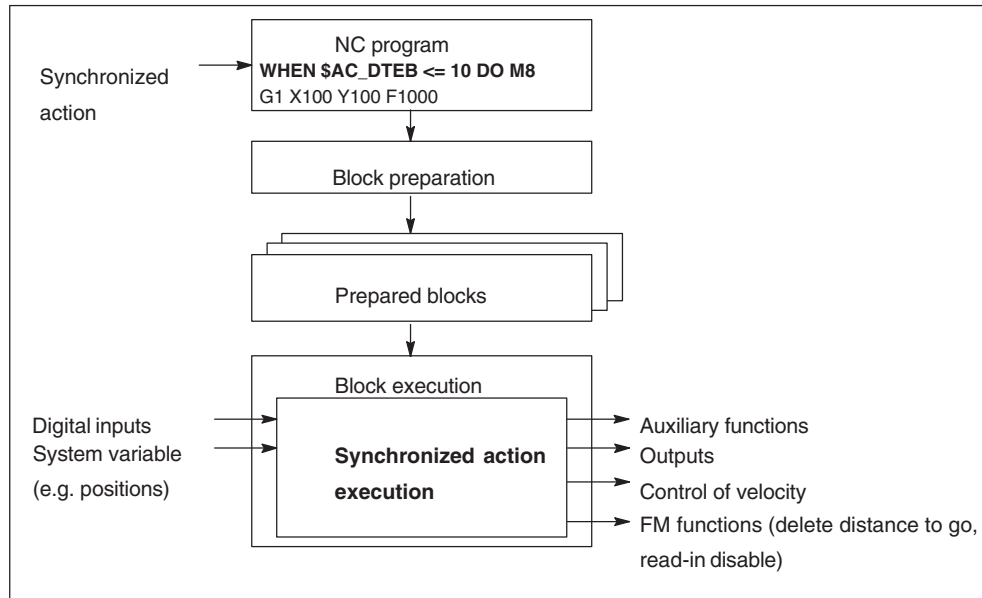


Fig. 10-63 Execution of a synchronous action

Further application examples

Fast start/stop of a single axis via a digital input

```
N10 ID=1 WHENEVER $A_IN[11] == FALSE DO $AA_OVR[X] = 0
N20 POS[X]=200 FA[X]=5000
```

The modal synchronized action in N10 results in the X axis being stopped every time the signal at digital input 11 switches from 1/0 (override = 0).

With the 0/1 signal change, the override is set internally to 100%, the axis continues to move.

Example of programming sequence for several synchronized actions

```
N10 WHENEVER $AA_IW[X] > 60 DO $AC_OVR = 30
N20 WHENEVER $AA_IW[X] > 80 DO $AC_OVR = 40
N30 ID= 2 WHENEVER $AA_IW[X] > 20 DO $AC_OVR = 20
N40 ID= 1 DO $AC_OVR = 10
N50 G1 X200 F1000
```

Synchronized actions are processed in the following sequence: N40 → N30 → N10 → N20. Depending on the actual position of the X axis, the velocity (via override) is increased:

IW < 20:	F=100
IW > 20:	F=200
IW > 60:	F=300
IW > 80	F=400

(The last written value remains active.)

10.33 Oscillation

General

The Oscillation function implements an axis motion between two reversal points that is independent of the NC program. After the oscillation motion has been activated, the remaining axis can be traversed as desired. The following NC statements can be programmed to define an oscillation motion or to change one that is already active.

The function is available for the FM 357-2LX.

Programming

OSCTRL[axis]	; Control statement
OSP1[axis]	; Position of reversal point 1
OSP2[axis]	; Position of reversal point 2
OST1[axis]	; Stop time at reversal point 1
OST2[axis]	; Stop time at reversal point 2
FA[axis]	; Feedrate of oscillation axis
OSNSC[axis]	; Number of residual strokes
OSE[axis]	; End position
OS[axis]=n	; Oscillation On/OFF
	n = 1 Oscillation On
	n = 0 Oscillation Off

Several oscillation axes can be active at the same time. Oscillation motions are always executed as a G1 motion.

Control statement OSCTRL[axis]=(SET, UNSET)

One of the functions of this statement is to define the oscillation motion response on deactivation.

SET values set and UNSET values delete individual control statements. Several control statements can be linked by +.

In response to Oscillation OFF (OS[axis]=0), the oscillation motion is ended by the approach to a reversal point (value: 0...3). The residual strokes (if programmed) can then be executed and an end position approached.

SET/UNSET values:

- 0: After Oscillation Off approach next reversal point (default)
- 1: After Oscillation Off approach reversal point 1
- 2: After Oscillation Off approach reversal point 2
- 3: After Oscillation Off do not approach any reversal point (if no residual strokes)
- 4: Approach end position on completion of residual strokes
- 8: After deletion of distance to go, execute residual strokes and approach end position if programmed
- 16: After deletion of distance to go approach reversal point acc. to 0...3
- 32: Feedrate change is not effective until next reversal point
- 64: Rotary axis is traversed via shortest route

Example:

OSCTRL[X]=(1+4+16, 8+32+64)

The oscillation motion is ended at reversal point 1. The residual strokes are then executed and the end position approached. If the distance to go is deleted, the oscillation axis approaches reversal point 1. Control statements 8, 32 and 64 are re-set if they were set previously in the program.

Position of reversal points OSP1[axis]/OSP2[axis]

The reversal point positions can be programmed as absolute or relative values.

- Absolute setting: OSP1[axis]=value
- Relative setting: OSP1[axis]=IC(value)
(position = reversal point 1 + value)

A relative position refers back to a previously programmed reversal point. Active offsets are applied.

Stop time at reversal point 1/2 OST1[axis]/OST2[axis]=value

This statement defines the axis behaviour at the reversal points.

Value:

- 2: Reverse without exact stop
- 1: Reverse with exact stop coarse target range
- 0: Reverse with exact stop fine target range
- >0: Stop time in seconds after exact stop fine target range

Number of residual strokes OSNSC[axis]

This statement defines the number of residual strokes to be executed at the end of the oscillation motion. A residual stroke is the movement to another reversal point and back.

End position OSE[axis]

This position is approached on deactivation of the oscillation motion (and execution of residual strokes if programmed) if control statement 4 or 8 is active for deletion of distance to go.

When an end position is programmed, OSCTRL[axis]=4 is generated internally.

Oscillation On/Off OS[axis]

The axis must be enabled for oscillation with WAITP(axis) before Oscillation On (OS[axis]=1).

Likewise, the axis must be enabled for other movements with WAITP(axis) after Oscillation Off (OS[axis]=0).

A program cannot be ended until the oscillation motion itself has ended.

Any active working area limitation is active. Protection zones are **not** monitored.

Control from NC program

An active oscillation motion can be controlled block-synchronously, i.e. as an NC block is processed, by the statements listed above.

Any change to the stop time or position of a reversal point does not take effect until the reversal point is approached again. The effectiveness of a change to the oscillation feedrate FA[axis] can be set with control statement OSCTRL[axis]=(32).

Programming examples

```

N10 G0 X0 Y0 Z0
N20 Z100
N30 WAITP(Z) ; Enable Z for oscillation

N50 OSP1[Z]=50 OSP2[Z]=100 OSE[Z]=150 ; Define oscillation motion
N60 OST1[Z]=0 OST2[Z]=5
N70 OSNSC[Z]=0 OSCTRL[Z]=(1, 0) FA[Z]=200
N80 OS[Z]=1 ; Oscillation On
... ; Any NC program
N100 OS[Z]=0 ; Oscillation Off
N110 WAITP(Z) ; Enable Z for another motion
N120 G0 Z0
N130 M2

```

The Z axis must oscillate between 50 and 100. It reverses at point 1 with exact stop fine and dwells for 5 s at point 2 after it has reached exact stop fine. In response to Oscillation Off, the Z axis traverses to the first reversal point and then to end position 150.

Synchronization of oscillation motion

The oscillation movement can be synchronized with any other movement. For further details, please refer to the functionality of synchronized actions described in Section 10.32.

Special system variables display the reversal points of the oscillation motion:

\$SA_OSCILL_REVERSE_POS1[axis]	Position of reversal point 1
\$SA_OSCILL_REVERSE_POS2[axis]	Position of reversal point 2

Programming example

```

N20 G0 Z100
N30 WAITP(Z) ; Enable Z for oscillation
N50 OSP1[Z]=50 OSP2[Z]=100 ; Define oscillation motion
N60 OST1[Z]=1 OST2[Z]=1
N70 OSCTRL[Z]=(1, 0) FA[Z]=200
N80 OS[Z]=1 ; Oscillation On
N90 ID=1 EVERY $AA_IW[Z]>= $SA_OSCILL_REVERSE_POS2[Z]
      DO POS[X]=IC(10) FA[X]=200
N100 ID=2 WHENEVER $AA_STAT[X]==1 DO $AA_OVR[Z]=0
N105 M0 ; Oscillation motion
N110 OS[Z]=0 ; Oscillation Off
N120 WAITP(Z) ; Enable Z for another motion
N125 G0 Z0
N130 M2

```

The Z axis oscillates between 50 and 100. Every time it reaches reversal point 2 (Z100), the X axis moves a distance of 10mm (ID=1). While the X axis is moving, the oscillation axis is stopped (ID=2)

10.34 Master value coupling

General

This function allows the position of a slave axis to be coupled to the position of a master axis. The functional interrelationship and the scope of definition of the coupling are defined in a curve table.

For further information about this function, please refer to Section 9.16.3.

Programming

CTABDEF(FA, LA, CTAB No, TYP)	; Begin curve table definition
CTABEND	; End curve table definition
CTABDEL(CTAB No)	; Delete a curve table
CTABDEL(CTAB-Nr, CTAB-Nr)	; Delete curve table range
	; (software version 5 and higher)
CTABDEL()	; Delete all curve tables
	; (software version 5 and higher)
CTAB(LW, CTAB-Nr, GRAD)	; Read out slave value for a master
CTABINV(FW, LAB, CTAB No, GRAD)	; Read out master value for a slave
LEADON(FA, LA, CTAB No)	; Activate coupling
LEADOF(FA, LA)	; Deactivate coupling
FA	; Slave axis
LA	; Master axis
FW	; Slave value (position)
LW	; Master value (position)
CTAB-Nr	; Number of curve table
TYP	; Curve table characteristics
	; 0: Curve table is not periodic
	; 1: Curve table is periodic (master value)
	; 2: Curve table is periodic (master and slave value)
LAB	; Range of expectation of master axis (if no unambiguous master
	; value can be determined for a slave value)
GRAD	; Gradient (output value)

Definition of curve table CTABDEF, CTABEND

Curve tables are defined in the NC program. The curve table starts with statement CTABDEF and ends with CTABEND. The motion statements for the master and slave axes each generate a curve segment. The selected interpolation mode (linear, circular, spline interpolation) determines the characteristic of the curve segment. The curve table characteristic is the same as the geometry of a “normally” programmed contour, all statements which affect the geometry (offset, tool offsets) are active.

After processing, the curve table is stored in the NC program memory.

The CTAB No. is used to select the relevant curve table when a master value coupling is activated (LEADON). A curve table stored in the NC program memory can be applied for any master or slave axis.

The scope of definition of the curve table is marked by the first and last value pairs of the master and slave axis positions.

The following statements may not be used:

- Stop preprocessor (STOPRE)
- Motion statement for one axis only
- Motion reversal of master axis
(assignment between master and slave axis positions no longer unambiguous)

Modal statements and R parameters outside the curve table are not affected by statements in the curve table.

Parameter TYP defines whether a curve table outputs periodic or nonperiodic slave values.

Periodic curve table TYPE 1:

The range of definition of the master axis is evaluated as a modulo value. A modulo conversion is performed for the continuous master value, the slave value is output periodically. The slave value must be identical at the beginning and end of the definition range to avoid step changes.

Note:

If the following axis is a modulo rotary axis and the last curve value is to be located at 360 degrees, e.g. the last pair of values must be written as follows:

```
...  
N90 X80 Y350  
N100 X100 Y=IC(10)
```

Periodic curve table TYPE 2:

The range of definition of the master axis is evaluated as a modulo value. A modulo conversion is performed for the continuous master value, the slave value is output periodically. The slave value must not be identical at the beginning and end of the definition range.

Nonperiodic curve table:

The curve table supplies values for the slave axis only within the definition range. The upper and lower limits are output as slave values outside the definition range.

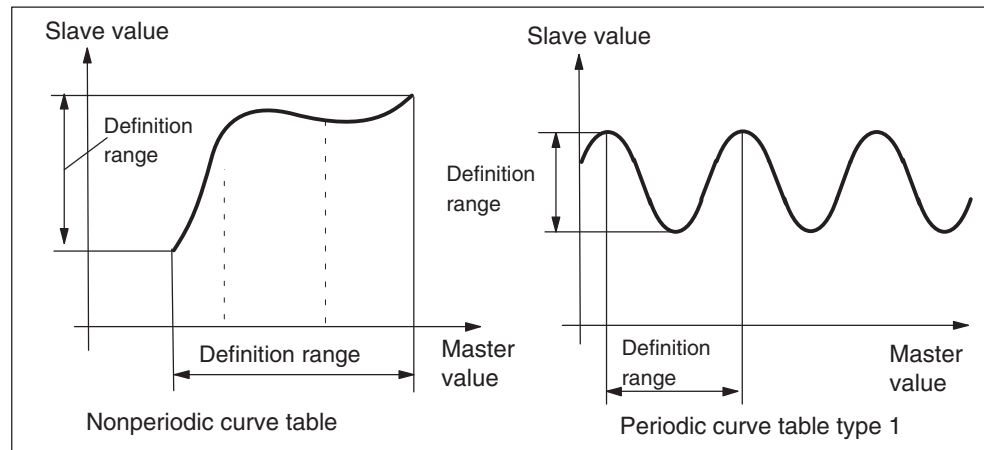


Fig. 10-64 Example of periodic and nonperiodic curve tables

Notice

If curve tables used in conjunction with modulo rotary axes are to act beyond the first revolution, a "Periodic curve table TYPE 1 or TYPE 2" must be selected for these axes.

Periodic curve tables are internally represented absolutely as a modulo rotary axis, i.e. if the following axis is located in the first modulo range of a periodic curve table (viewed absolutely) and is not coupled, it can no longer be coupled in the next modulo range.

The conversion to the absolute position is carried out after activating LEADON, irrespective of whether or not it was coupled.

Reading table values CTAB and CTABIN

The slave value for a master value can be read with CTAB, either directly from the NC program or in synchronized actions.

Example:

```
N05 DEF REAL GRAD           ; Definition of the real variable "GRAD"
N10 R20 = CTAB(100, 1, GRAD) ; The slave value for master value 100 of curve
                             ; table 1 is stored in R20. The gradient at this
                             ; point is entered in variable GRAD
```

The master value for a slave value can be read with CTABINV. An approximate value for the expected master value must be specified since the assignment between slave and master is not always unambiguous.

Example:

```
N10 R30 = CTABINV(50, 20, 2, GRAD) ; The master value for slave value 50
                                     ; and for expected master value 20 is
                                     ; entered in R30
```

Deleting a curve table CTABDEL(CTAB No)

This statement is used to delete the table saved with the CTAB no.

A curve table can be overwritten by the CTABDEF statement; no warning is output.

Deleting a curve table CTABDEL(CTAB-Nr N, CTAB No M)

This statement is used to delete the tables saved with CTAB No. through CTAB No. M. If an error occurs when deleting, e.g. a table is active during coupling (LEADON), the process of deletion is canceled.

Deletion of a curve table CTABDEL()

This statement deletes the curve table stored under CTAB No.

Example

```

N100 CTABDEF(Y,X,3,0) ; Begin definition of a nonperiodic curve table with the
                        ; number 3
N105 ASPLINE          ; Spline interpolation
N110 X5 Y0           ; 1st motion statement, defines start values and
                        ; 1st interpolation point: Master value: 5; Slave value: 0
N120 X20 Y0          ; 2nd interpolation point: Master value: 5...20;
                        ; Slave value: Start value...0
N130 X100 Y6         ; 3rd interp. point: Master val.: 20...100; Slave val.: 0...6
N140 X150 Y6         ; 4th interp. point: Master val.: 100...150; Slave val.: 6...6
N150 X180 Y0         ; 5th interp. point: Master val.: 150...180; Slave val.: 6...0
N200 CTABEND        ; End of definition; the curve table is generated in its in-
                        ; ternal representation as a max. 3rd degree polynomial;
                        ; the method by which the curve with the specified interp.
                        ; points is calculated will depend on the modally selected
                        ; interp. method (in this example: Spline interp.); the NC
                        ; program status before the definition was started is
                        ; restored.

```

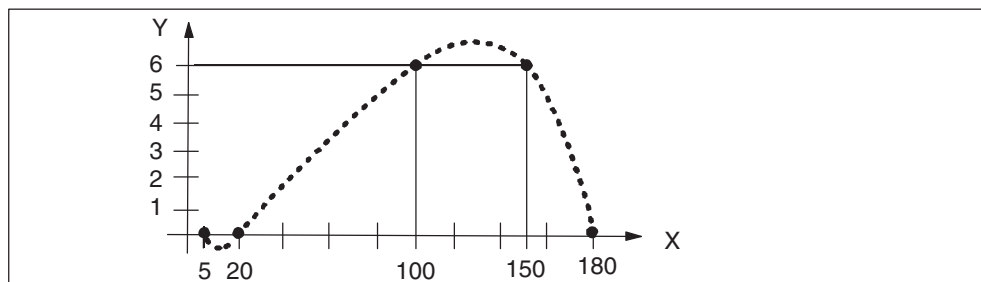


Fig. 10-65 Example of curve table definition

Activating and deactivating a master value coupling LEADON / LEADOF

A master value coupling must be activated with statement LEADON. After a synchronization run, the slave axis is moved solely via the master value coupling.

LEADOF deactivates the coupling.

Example:

```

...
N30 LEADON(Y, X, 1) ; Activate master value coupling, Y is slave axis,
                    ; X is master axis, curve table 1 is active
N40 G1 X200 F200    ; The slave axis may not be programmed until
                    ; the coupling is deactivated again
...
N50 LEADOF(Y, X)   ; Deactivate master value coupling
N60 X0 Y20         ; Slave axis Y can now be moved again
                    ; under the control of the NC program.
...

```

Master value couplings can also be activated and deactivated from synchronized actions (see Section 10.32).

System variables for master value coupling

You can read or write data for a master value coupling via the following system variables:

Read:

\$AA_LEAD_V[LA]	; Velocity of master axis
\$AA_LEAD_P[LA]	; Position of slave axis
\$AA_LEAD_P_TURN[LA]	; Modulo position of master axis with periodic ; Curve table
\$AA_SYNC[LA]	; Synchronism status ; 0: Not synchronized ; 1: Coarse synchronism ; 2: Fine synchronism ; 3: Coarse and fine synchronism

Read/write:

\$AA_LEAD_SV[LA]	; Velocity per IPO cycle with simulated master ; value
\$AA_LEAD_SP[LA]	; Position in machine coordinate system with ; Simulated master value

Simulated master value

If “simulated master value” is parameterized as the type of coupling, the master value must be specified via the system variable \$AA_LEAD_SP[LA] cyclically in the IPO cycle of a synchronization.

Example 1:

The A axis is parameterized as an external master and is to be the master axis of a master value coupling. A simulated master value will be created using the following synchronized action:

```
...
N20 whenever $SA_LEAD_TYPE[A]==2 DO $AA_LEAD_SP[A]=$AA_IM[A]
...
```

The programming of the system variable \$AA_LEAD_SP[...] in synchronized actions allows, e.g. a correction or smoothing of the actual value provided from the external master.

Example 2:

```
...
whenever $SA_LEAD_TYPE[A]==2 DO
    $AA_LEAD_SP[A]=$AA_IM[A]+ ($V A_IM[A] - $AC_PARAM[2])*3

whenever $SA_LEAD_TYPE[LAX]==2 DO $AC_PARAM[2]= $VA_IM[A]
...
```

The actual value provided from master A is corrected by 3 * path per IPO cycle.

10.35 Electronic gear

General

The function “Electronic gear” (EG) is intended to couple the motion of a following axis to the motion of four master axes as the maximum. Gear cascading is permitted, i.e. a following axis can be master axis of a series-connected electronic gear. A following axis can continue to be traversed using the NC program; the portion of the EG motion is overlaid in this case.

The functional interrelation between master axis and following axis is defined either by a coupling factor or a curve table. Portions of motions of several master axes act to the following axis motion additively.

The motions of the following axis can be derived either from the setpoint or from the actual value of the master axis.

The function “Electronic gear” **cannot** be programmed using synchronized actions.

This function is available as from software version 5.

Programming

EGDEF(FA, LAi, TYPi, ...)	; Define EG group
EGON(FA, SW, LAi, Zi, Ni, ...)	; Activate EG
EGONSYN(FA, SW, SP_FA, LAi, SP_LAi, Zi, Ni, ...)	; Activate EG with specification ; of the synchronized position
EGONSYNE(FA, SW, SP_FA, AM, LAi, SP_LAi, Zi, Ni, ...)	; Activate EG with specification of ; synchronized position and approach ; mode for modulo master and ; following axes
EGOFS(FA)	; Deactivate all EGs of a ; following axis
EGOFS(FA, LAi, ...)	; Deactivate individual EGs
EGDEL(FA)	; Delete the definition of an ; EG group

Parameters:

i Number of master axes: i = 1...4

Type Setpoint or actual-value coupling:

Type = 0 Actual-value of the master axis

Type = 1 Setpoint of the master axis

FA Following axis

LA Master axis

Z Numerator of coupling factor

N Denominator of coupling factor

SW Block change mode: (string parameters in inverted commas !)

“NOC”: Block change is carried out immediately

“FINE”: Block change is carried out at fine synchronization (default)

“COARE”: Block change is carried out at coarse synchronization

“IPOSTOP”: Block change is carried out at setpoint synchronization

AM Approach mode: (string parameters in inverted commas !)

“ACN”: Absolute dimension, negative approach direction

“ACP”: Absolute dimension, positive approach direction

“DCT”: Absolute dimension, time-optimized approach

“DCP”: Absolute dimension, path-optimized approach

“NTGT”: Next tooth gap, time-optimized approach (default)

“NTGP”: Next tooth gap, path-optimized approach

SP_FA Synchronized position of the following axis

SP_LA Synchronized position of the master axis

The statements with reference to the EG must be programmed in a separate block of the NC program.

All statements except EGON, EGONSYN and EGONSYNE create an internal preprocessing stop.

Define EG – EGDEF

The statement EGDEF assigns a following axis at least one and a maximum of 4 following axes.

The type of coupling can be different for each master axis.

A coupling may not yet be defined for the following axis.

EGDEF is the precondition to enable the EG.

Activate EG – EGON

With this command, the EG is turned on immediately. The current position of master and following axes are to be considered the synchronized EG positions.

Activate EG – EGONSYN

The programmed synchronized positions for master and following axes define the point at which the EG reaches the synchronous condition.

If the EG is not yet synchronized at the time when the statement EGONSYN/E is provided, the following axis is traversed via motion overlay to the synchronized position.

If the EG contains modulo axes, their positions are reduced modulo. This guarantees that in any case the next possible synchronized position is approached, i.e. a “relative” synchronization is provided.

Note:

If master and following axes are modulo axes, the FM calculates a “tooth gap” from the coupling factor and synchronizes then to the **next** “tooth gap”.

Example: Coupling factor = 2/10 → tooth spacing = $(360 \text{ deg} / 10) * 2 = 72 \text{ deg}$.

This behavior corresponds to the approach mode NTGT or NTGP with EGONSYNE. If this behavior is not desired, you EGONSYNE with the appropriate approach mode.

The synchronized position is approached within an optimum time, without taking into account the maximum velocity of the following axis. This can result in long synchronization times or no synchronization is reached at all. In such cases, you should also use EGONSYNE.

Activate EG – EGONSYNE

This function can be used if master **and** following axes are modulo axes. In addition to the synchronized position, an approach mode can be programmed.

The direction or path/time-optimized approach of the following axis to the synchronized position can be selected via the approach mode.

Example:

EGDEF (Y,X,1)

EGONSYNE (Y, “FINE”, SP_FA, “AM”, X, SP_LA, 2, 10) ; coupling 2/10

Synchronized position SP_		Position upstream of EGONSYNE		Approach mode AM			
LA	FA	LA	FA	DCT/DCP ¹⁾	ACP	ACN	NTGT/NTGP ²⁾
0	110	0	150	110 (−40)	110 (+320)	110 (−40)	182 (+32)
0	110	0	350	110 (+120)	110 (+120)	110 (−240)	326 (−24)
0	130	0	0	130 (+130)	130 (+130)	130 (−230)	346 (−14)
0	130	0	30	130 (+100)	130 (+100)	130 (−260)	58 (+28)
0	130	0	190	130 (−60)	130 (+300)	130 (−60)	202 (+12)
0	190	0	0	190 (−170)	190 (+190)	190 (−170)	334 (−26)
0	230	0	0	230 (−130)	230 (+230)	230 (−130)	14 (+14)

1) Assumption: The following axis is at standstill at the moment when the control system is turned on.
2) Gearwheel

There is only a difference between path and time-optimized approach if the following axis is moving at the time when the control system is turned on.

Example:

Influence of the synchronized position of the master axis, position of the master axis upstream of EGONSYN/E and of the coupling factor 2/10 to the synchronized position of the following axis:

Synchr. position SP_		Position upstream of EGONSYNE		Difference programmed SP_FA approached SP_FA	Approach mode AM		
LA	FA	LA	FA		DCT/DCP	ACP	ACN
0	110	0	150	0	110 (-40)	110 (+320)	110 (-40)
30	110	0	150	-30/10	107 (-43)	107 (+317)	107 (-43)
0	110	30	150	30 * 2 /10	116 (-34)	116 (+326)	116 (-34)
30	110	30	150	30+2/10-30 /10	113 (-37)	113 (+323)	113 (-37)

EG and gearwheel

If the following axis moves a gearwheel, the tooth spacing or tooth gap can be taken into account using the NTGT and NTGP (next tooth gap) approach modes. It is tried to approach the next tooth gap as the synchronized position. A tooth spacing of 72 deg (2/10 coupling factor) is assumed in the table.

Deactivate EG – EGOFS

The EG can be deactivated either selectively or completely.

EGOFS(FA)

This command deactivates the EG completely; the following axis will brake up to standstill.

EGOFS(FA, LA1, ...)

The portion of the specified master axes is deactivated selectively. The following axis continues to traverse, depending on the remaining master axes.

Delete EG – EGDEL

An EG defined with EGDEF can be deleted with EGDEL.

System variable to the EG

You can export information with regard to the coupling using the following system variables:

- \$AA_EG_TYPE[FA,LA] ; Coupling type
; 0: Actual-value coupling, 1: Setpoint coupling
- \$AA_EG_NUMERA[FA,LA] ; Numerator of coupling factor
; Number of curve table if numerator = 0
- \$AA_EG_DENOM[FA,LA] ; Denominator of coupling factor
- \$AA_EG_SYN[FA,LA] ; Synchronized position for the specified
; master axis

\$AA_EG_SYNFA[FA,LA]	; Synchronized position for the specified ; following axis
\$AA_EG_BC[FA]	; Block change mode when activating the EG ; for the specified following axis
\$P_EG_NUM_LA[FA]	; Number of master axes defined for the specified ; following axis
\$AA_EG_AX[n, FA]	; Axis name of the nth master axis
\$AA_EG_ACTIVE[FA,LA]	; EG status with reference to the specified master ; and following axes ; 0: disabled, 1: enabled
\$VA_EG_SYNCDIFF[FA]	; Amount of the synchronism difference with ; reference to the specified following axis
\$VA_EG_SYNCDIFF_S[FA]	; Synchronism difference with reference to the ; specified following axis

Programming example

A modulo following axis is to be coupled to two modulo master axes. The coupling factor is 1/2 for both master axes.

```

; Starting position
N20 EGOFS(Z)           ; Disable EG (if active)
N30 EGDEL(Z)          ; Delete EG (if defined)

N10 G0 X0 Y0 Z0       ; Approach starting position without EG

; EG definition
N40 EGDEF(Z,X,0,Y,0)  ; Master axis: X, Y, following axis Z
                       ; actual-value coupling

; Enable EG
; Interface signal "Enable following axis superimposition" (user DB "AXy",
; DBX8.4+m) !
N50 EGONSYNE (Z, "FINE", 100, "ACP", X, 50, 1,2, Y, 100, 1, 2)
; Approached Z pos. = 100 - (50/2) - (100/2) = 25

N60 X=IC(300) Y=IC(300) ; Z = 25 + (300/2) + (300/2) = 325

N70 X=IC(180) Y=IC(-180) ; Z = 325 + (180/2) - (180/2) = 325
                       ; Z does not carry out any motion

N80 EGOFS (Z,Y)       ; Disable Y axis selectively

N90 X0 Y0             ; Z = 325 - (120/2) = 265

N100 X100 Z=IC(-25)  ; Z = 265 + (100/2) - 25 = 290

N110 M30

```

10.36 Axis replacement

General

Using the function “Axis replacement”, it is possible to enable an axis in one channel and to assign it to a different channel.

If you program an axis which has been assigned to another channel or to the CPU using the “Axis replacement function”!, the NC program is stopped.

The FM message “Wait: Axis replacement, axis is in a different channel” or “Wait: Axis replacement, axis is CPU axis” appears.

This function is available as per from software version 5.

Programming

RELEASE (axis, axis, ...)	; Bring axis to the neutral condition
GET (axis, axis, ...)	; Accept axis from the neutral condition into the ; current channel
GETD (axis, axis, ...)	; Accept the axis directly from a channel into the ; current channel

Channel assignment

The axis must be assigned to several channels including a master channel via the parameterization. The master channel will define the default assignment for the axis after activating the module.

RELEASE

The axis is enabled, i.e. it changes from the channel axis status to the status of a neutral axis. RELEASE will create an NC block of its own. If the axis is already in the neutral state, RELEASE will not be carried out once again.

Special features:

- The axis must not be traversed, e.g. using POSA in the NC block or POS in synchronized actions.
- With tangential control, all axes of the group must be enabled.
- With motion coupling (coupled motion, master value coupling, electronic gear), only the master axis can be enabled.
- A RELEASE to a Gantry master axis will also result in a RELEASE for the Gantry following axis.

GET

With GET, the axis is accepted from the neutral state into the current channel.

If the axis was assigned to a different channel or to the CPU, synchronization is required. An internal STOPRE is created, and the current axis position is accepted. The NC block sequence is interrupted until the axis replacement is carried out completely.

An axis accepted with GET or GETD will also remain in this channel after Reset. The axis can only be assigned to another channel by carrying out axis replacement once again or to the master channel by **Power On**.

Special features:

- If the axis is already in the channel, GET will not be carried out; in this case, e.g. a G64 block sequence will not be interrupted.
- A neutral axis will turn into a channel axis without synchronization if it was enabled with RELEASE from this channel first and it was meanwhile assigned to no other channel or to the CPU.
- In the status “neutral axis”, an axis can also be programmed in the channel without GET; an internal (implied) GET is created.

GETD

With GETD, the axis is fetched directly from another channel; no RELEASE is required in advance.

GETD produces an interruption of the program execution in the issuing channel; REORG is triggered.

If necessary you can use the program coordination functions to synchronize the channels to one another (see Section 10.28).

Programming example

The X and Y axes are assigned to channels 1 and 2. Master channel for both axes is channel 1. The two axes should be changed between the channels.

Channel 1 should control the whole sequence.

Channel 1 additionally contains the Z axis, channel 2 the A axis.

```
; MASTER_K1.MPF           ; Channel 1: Control program
```

```
N10 CLEARM()
```

```
N20 INIT (2, "SLAVE_K2")  ; Program selection for channel 2
```

```
N30 START (2)             ; Start in the 2nd channel
```

```
N50 WAITM(1,2)           ; Wait for synchronism mark 1, channel 2
```

```
N60 GET(X1, Y1)          ; Channel 2 has enabled the axis, i.e. "fetch"
```

```
N50 G90 F1000 G64 X111 Y111 Z11
```

```

...
N100 SETM (2)           ; Channel 2 is reporting that X and Y are no longer
                        ; needed

N120 Z111
N130 Z1
...

N200 WAITM (3,2)
N210 CLEARM()

M30

; SLAVE_K2.MPF           ; Channel 2

N10 CLEARM ()
N20 RELEASE(X1, Y1)    ; Enable
N30 SETM(1)            ; and report

N40 POS[A]=200         ; An executable block must be programmed
                        ; between SETM and WAITM

N50 WAITM (2,1)        ; Wait for synchronism mark 2, channel 1
N60 GETD (X1, Y1)     ; Fetch axes directly

N50 G1 F1000 X222 Y222 A22
...

N100 RELEASE (X1, Y1)  ; Enable
N110 SETM(3)           ; and report

N200 A222

N250 M30

```

10.37 Speed feedforward control (FFWON, FFWOF)

General

The speed feedforward control function applies an additional velocity setpoint to the input of the speed controller, thus reducing the velocity-dependent following errors to zero.

This function makes it possible to increase path accuracy.

Programming

FFWON ; Activate feedforward control
FFWOF ; Deactivate feedforward control

The setting in parameter “Speed feedforward control” defines which axes are to be traversed with feedforward control applied. Parameters “Time constant current control loop” and “Weighting factor” must be set to achieve an optimum speed feedforward control setting (see Section 9.3, Position control).

Example:

```
N10 G0 X0 Y0  
N20 FFWON  
N30 G1 X100 Y200 F2000
```

The axes traverse with active feedforward control from N20.

10.38 Overview of statements

Table 10-12 Overview of statements

Statement	Meaning	Information/value range	Section
ABS()	Absolute value	Parameter calculation	10.18
AC	Absolute dimension, axis-specific	Non-modal	10.2.3
ACC[]	Programmable acceleration	0...200 %	10.32
ACN	Absolute dimensions for rotary axes, in negative direction	Non-modal	10.2.4
ACP	Absolute dimensions for rotary axes, in positive direction	Non-modal	10.2.4
ADIS	Rounding clearance for path feed		10.6.2
ADISPOS	Rounding clearance for rapid traverse		10.6.2
AMIRROR	Programmable mirror additive	Group 3, non-modal	10.3.2
AND	AND	Logical operator	10.18
AP	Polar angle	$\pm 0.00001 \dots 360^\circ$	10.2.5
AR	Angle of rotation		10.5.12
AROT	Programmable rotation additive	Group 3, non-modal	10.3.2
ASPLINE	Akima spline	Group 1, modal	10.5.10
ATRANS	Programmable zero offset additive	Group 3, non-modal	10.3.2
AX [axis variable]	Axis programming, variable axis name		10.25
AXNAME (STRING)	STRING type conversion to axis name		10.25 10.26
AXSTRING (AXIS)	Axis name type conversion to STRING		10.25 10.26
BAUTO	Start of spline curve, no command	Group 19, modal	10.5.10
BNAT	Start of spline curve, zero curvature	Group 19, modal	10.5.10
BRISKA()	Brisk acceleration for positioning axes		10.6.3
BRISK	Brisk acceleration for path axes	Group 21, modal	10.6.3
BSPLINE	B spline	Group 1, modal	10.5.10
BTAN	Start of spline curve, tangential transition	Group 19, modal	10.5.10
B_AND	AND by bits	Bitserial logical operator	10.18
B_NOT	Negated by bits	Bitserial logical operator	10.18
B_OR	OR by bits	Bitserial logical operator	10.18
B_XOR	Bitserial exclusive OR	Bitserial logical operator	10.18
CANCEL()	Deletion of modal or static synchronized actions	Synchronized action	10.32
CLRINT()	Clear assignment between digital input and NC program		10.30

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
CALL	Indirect subroutine call		10.29
CASE	Case differentiation	Control structure	10.24
CHF	Chamfer; value corresponds to the chamfer length		10.5.13
CHR	Chamfer; value corresponds to the distance from chamfer start to the programmed end-of-block position		10.5.13
CLEARM (marker1, marker2,...)	Clear synchronism marker		10.28
CLRINT()	Clear assignment of digital input to NC program		10.30
COARSEA[]	Block change at "Exact stop coarse"		10.5.8
COS()	Cosine	Degrees	10.18
CR	Circle radius		10.5.9 10.5.12
CSPLINE	Cubic spline	Group 1, modal	10.5.10
CTABDEF()	Beginning of curve table definition	Synchronized action	10.34 10.32
CTABDEL()	Delete curve table	Synchronized action	10.34 10.32
CTABEND()	End of curve table definition	Synchronized action	10.34 10.32
CTAB()	Read out slave value for a master value	Synchronized action	10.34 10.32
CTABINV()	Read out master value for a slave value	Synchronized action	10.34 10.32
DC	Absolute dimensions for rotary axes, shortest path	Non-modal	10.2.4
DELDTG	Delete distance to go with preprocessing stop for path axes	Synchronized action	10.32
DELDTG()	Delete distance to go with preprocessing stop for positioning axes	Synchronized action	10.32
DELETE (error, filename)	Delete file		10.28
DEF AXIS axis variable = X	Definition of an axis variable and assignment of a (valid) axis name		10.25
DEF STRING[m] name	Definition of a string variable; string length m		10.26
DISABLE()	Deactivate ASUB		10.30
DISPLOF	Deactivate NC block display for PROG_EVENT.SPF		9.8

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
DIV	Division (INT or REAL) (type INT)/(type INT) = (type INT)	Arithmetic operator	10.18
DO	Action component	Synchronized action	10.32
DRIVEA()	Drive acceleration for positioning axes		10.6.3
DRIVE	Drive acceleration for path axes	Group 21, modal	10.6.3
EAUTO	End of spline curve, no command	Group 20, modal	10.5.10
EGDEF()	Define EG group		9.16.4
EGDEL()	Delete an EG group		9.16.4
EGOFS()	Deactivate all EGs of a following axis		9.16.4
EGON()	Activate EG		9.16.4
EGONSYN()	Activate EG with specification of the synchronized position		9.16.4
EGONSYNE()	Activate EG with specification of the synchronized position and the approach mode for modulo master and following axes		9.16.4
ENABLE()	Activate ASUB		10.30
ENAT	End of spline curve, zero curvature	Group 20, modal	10.5.10
ETAN	End of spline curve, tangential transition	Group 20, modal	10.5.10
EVERY	Scanning/execution frequency	Synchronized action	10.32
EXECUTE	End of definition of a protection zone		10.17
EXTERN	External declaration for parameter transfer		10.29
F	Path feed	Path velocity in mm/min, inch/min, deg/min $0.001 \leq F \leq 999\,999.999$ (metric) $0.001 \leq F \leq 399\,999.999$ (inches)	10.5.1
FA	Feedrate for positioning axes	$0.001 \leq F \leq 999\,999.999$ (metric) $0.001 \leq F \leq 399\,999.999$ (inches)	10.5.1
	Feedrate of oscillation axis		10.33
FCUB	Cubic feedrate (spline)		10.5.2
FFWON	Activate feed control		10.37
FFWOF	Deactivate feed control		10.37
FGREF[]	Reference radius for rotary axes		10.5.3
FGROUP()	Path group		10.5.3
FINE[]	Block change at "Exact stop fine"		10.5.8
FL	Limit feed for synchronized axes	Modal	10.5.1
FNORM	Constant feedrate		10.5.2

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
FLIN	Linear feedrate		10.5.2
FOR ENDFOR	Counter loop	Control structure	10.24
FRC	Feedrate chamfer/rounding, non-modal		10.5.13
FRCM	Feedrate chamfer/rounding, modal		10.5.13
FROM	Scanning/execution frequency	Synchronized action	10.32
FXS[]	Select/deselect travel to fixed stop	Modal	10.11
FXST[]	Clamping torque	Modal	10.11
FXSW[]	Monitoring window	Modal	10.11
GET()	Accept axis from the neutral status into the current channel		9.21
GETD()	Accept axis directly from a channel into the current channel		9.21
GOTOB	Jump backwards		10.23
GOTOF	Jump forwards		10.23
G0	Linear interpolation with rapid traverse	Group 1, modal	10.5.4
G1	Linear interpolation with feed	Group 1, modal	10.5.6
G2	Circular interpolation clockwise	Group 1, modal	10.5.9
G3	Circular interpolation counterclockwise	Group 1, modal	10.5.9
G4	Dwell	Group 2, non-modal	10.7
G9	Exact stop	Group 11, non-modal	10.6.1
G25	Minimum working area limitation	Group 3, non-modal	10.13
G26	Maximum working area limitation	Group 3, non-modal	10.13
G17	Plane selection	Group 6, modal	10.2.7
G18	Plane selection	Group 6, modal	10.2.7
G19	Plane selection	Group 6, modal	10.2.7
G500	Settable zero offset off	Group 8, modal	10.3.1
G53	All zero offsets off	Group 9, non-modal	10.3.1
G54	1st settable zero offset	Group 8, modal	10.3.1
G55	2nd settable zero offset	Group 8, modal	10.3.1
G56	3rd settable zero offset	Group 8, modal	10.3.1
G57	4th settable zero offset	Group 8, modal	10.3.1
G60	Exact stop	Group 10, modal	10.6.1
G601	Block change on target range fine	Group 12, modal	10.6.1
G602	Block change on target range coarse	Group 12, modal	10.6.1
G603	Block change if "IPO end" not reached	Group 12, modal	10.6.1
G64	Continuous-path mode	Group 10, modal	10.6.2

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
G641	Continuous-path mode with programmed rounding clearance	Group 10, modal	10.6.2
G70	Measurement in inches	Group 13, modal	10.2.6
G71	Measurement in meters	Group 13, modal	10.2.6
G90	Absolute dimensions	Group 14, modal	10.2.3
G91	Incremental dimensions	Group 14, modal	10.2.3
G110	Polar dimension with reference to the last programmed position	Group 3, non-modal	10.2.5
G111	Pole dimension with reference to workpiece zero	Group 3, non-modal	10.2.5
G112	Polar dimension with reference to the last valid pole	Group 3, non-modal	10.2.5
H	H function	0... 99	10.15
I	Interpolation parameter	1st geometry axis	10.5.9
IC	Incremental dimension, axis-specific	Non-modal	10.2.3
ID	Number of synchronized action	Modal	10.32
IDS	Number of a static synchronized action	Modal	10.32
IF	Conditional program jumps		10.23
IF ELSE END IF	Choice between 2 alternatives		10.24
INDEX (STRING, CHAR)	Character search in the STRING, starting from the beginning of the string		10.26
INIT (channel no., "program")	Program selection		10.28
INVCW	Involute interpolation CW		10.5.12
INVCWW	Involute interpolation CCW		10.5.12
IPOBRKA[]	Block change at % value of the braking ramp		10.5.8
IPOENDA[]	Block change at "IPO end"		10.5.8
ISAXIS (axis number)	Test for permissible geometry axis no's. 1...3		10.25
ISNUMBER (STRING)	Type conversion STRING to BOOL		10.26
ISVAR()	checks whether the variable name is known		10.22
ITOR	Conversion from INT to REAL	Synchronized action	10.32
J	Interpolation parameter	2nd geometry axis	10.5.9
K	Interpolation parameter	3rd geometry axis	10.5.9
L	Subprogram name and call		10.29
LEADON()	Activation of coupling	Synchronized action	10.34 10.32
LEADOF()	Deactivation of coupling	Synchronized action	10.34 10.32

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
LOCK	Disable synchronized action	Synchronized action	10.32
LOOP END-LOOP	Endless loop	Control structure	10.24
M0	Stop at end of block	Fixed	10.14
M1	Conditional stop	Fixed	10.14
M2, M30	End of program	Fixed	10.14
M17, M3, M4, M5, M6, M40, M41...M45, M70	Disabled		10.14
M...	Unassigned M functions:	0...99 (except for permanent and disabled)	10.14
MATCH (STRING), STRING	String search in the STRING		10.26
MCALL	Modal subroutine call		10.29
MEAC[]	Axial measuring continuously (SW 5.2 and higher)		10.10.2
MEAS	Measurement with delete distance to go		10.10.1
MEASA	Axial measurement with deletion of distance to go		10.10.2
MEAW	Measurement without delete distance to go		10.10.1
MEAWA[]	Axial measurement without deletion of distance to go	Synchronized action	10.10.2 10.32
MINDEX (STRING, CHAR, CHAR, ...)	Search for several characters in the STRING, starting from the beginning of the string		10.26
MIRROR	Programmable mirror absolute	Group 3, non-modal	10.3.2
MOD	Modulo division (INT or REAL) provides the rest of an INT division	Arithmetic operator	10.18
MOV[]	Positioning motion without end position	Synchronized action	10.32
MSG	Output messages		10.1.3
N	Block number, subblock		10.1.3
NEWCONF	Activate machine data		10.31
NOT	NOT, negation	Logical operator	10.18
NPROTDEF	Definition of protection zones		10.17
NPROT	Activate protection zone		10.17
NUMBER (STRING)	Type conversion STRING to REAL		10.26
OR	OR	Logical operator	10.18
OS[]=n	Oscillation On/Off	n = 1, Oscillation On n = 0, Oscillation Off	10.33
OSCTRL[]	Control statement		10.33

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
OSE[]	End position		10.33
OSNSC[]	Number of residual strokes		10.33
OSP1[] OSP2[]	Position of reversal point 1 / 2		10.33
OST1[] OST2[]	Stop time at reversal point 1 / 2		10.33
P...	Program repetition		10.29
PRIO	Define priority	1...128	10.30
PROC	Define an ASUB		10.30
PL	Node distance for spline Length of parameter interval (polynomial interpolation)		10.5.10 10.5.11
PO[]	End points and polynomial coefficients		10.5.11
POLY	Polynomial interpolation		10.5.11
POS[]	Positioning movement with impact on block changeover Positioning movement to end position	Synchronized action	10.5.7 10.32
POSA[]	Positioning movement without impact on block changeover		10.5.7
POT()	Square		10.18
PRESETON	Set actual value	Synchronized action	10.4 10.32
PW	Point weight for spline		10.5.10
R	Arithmetic parameter	R0...R99	10.19
RDISABLE	Programmed read-in disable	Synchronized action	10.32
READ (error, file name, line, number, result)	Write file		10.28
RELEASE()	Enable axis		10.36
REPEAT UNTIL	Loop with condition at the loop end		10.24
REPOSL	Reposition at interruption point in main program/subroutine		10.30
RESET()	Reset synchronized action	Synchronized action	10.32
result = ISFILE (file name)			10.28
RET	End of subroutine without CPU output		10.29
RINDEX (STRING, CHAR)	Character search in the STRING, starting from the end of the string		10.26
RND	Rounding; value corresponds to the radius		10.5.13

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
RNDM	Rounding modal; value corresponds to the radius		10.5.13
ROT	Programmable rotation absolute	Group 3, non-modal	10.3.2
RP	Polar radius	Positive values in mm or inches	10.2.5
RPL	Angle of rotation in active plane	Group 3	10.3.2
RTLIOF	Rapid traverse with positioning motion		10.5.5
RTLION	Rapid traverse with linear interpolation		10.5.5
RTOI	Conversion from REAL to INT	Synchronized action	10.32
SD	Degree of B spline		10.5.10
SETINT()	Assign a digital input	1... 8	10.30
SAVE	Restore interruption position and current processing status		10.29 10.30
SETM (marker1, marker2, ...)	Set synchronism marker		10.28
SIN()	Sinus		10.18
SOFTA()	Soft acceleration for positioning axes		10.6.3
SOFT	Soft acceleration for path axes	Group 21, modal	10.6.3
START (channel no., channel no., ...)	Program start		10.28
STOPRE	Block preprocessor stop		10.12
STRLEN (STRING)	Linking and type conversion to STRING		10.26
STRING[M]	Selection of a character with index M		10.26
SQRT()	Square root		10.19
SUBSTR (STRING, M)	Selection of a substring from index M up to string end		10.26
SUBSTR (STRING, M, N)	Selection of a substring with length M from index M		10.26
T	Tool number	0... 49	10.16
TAN()	Tangent	Degrees	10.18
TANG(slave axis, master axis1, master axis2, coupling factor)	Definition of the tangential control		10.9
TANGOF (slave axis)	Disable tangential control		10.9
TANGON (slave axis, angle)	Enable tangential control		10.9

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
TLIFT (slave axis)	Permit intermediate block at contour corners		10.9
TOLOWER (STRING)	Conversion uppercase to lowercase letters		10.26
TOUPPER (STRING)	Conversion lowercase to uppercase letters		10.26
TRAILON	Define and activate coupled-axis grouping	Synchronized action	10.8 10.32
TRAILOF	Deactivate coupled-axis grouping	Synchronized action	10.8 10.32
TRANS	Programmable zero offset absolute	Group 3, non-modal	10.3.2
TRUNC()	Integer component (truncate)		10.18
UNLOCK()	Enable synchronized action	Synchronized action	10.32
UP_NAME...	Subroutine call without parameter transfer		10.29
UP_NAME(PARA 1, PARA2, ...)	Subroutine call with parameter transfer		10.29
VAR	Parameter transfer with return		10.29
VELOLIM[]	Programmable maximum velocity		10.6.4
WAITE (channel no., channel no., ...)	Wait for end of program		10.28
WAITM (marker1, channel no., ...)	Wait for synchronism marker		10.28
WAITMC (marker1, channel no., ...)	Conditional waiting for synchronism marker		10.28
WAITP()	Wait until position reached		10.5.7
WALIMON	Working area limitation on	Group 28, modal	10.13
WALIMOF	Working area limitation off	Group 28, modal	10.13
WHEN	Scanning/execution frequency	Synchronized action	10.32
WHENEVER	Scanning/execution frequency	Synchronized action	10.32
WHILE ENDWHILE	Loop with condition at loop beginning	Control structure	10.24
WRITE (error, file name, text)	Read file		10.28
XOR	Exclusive OR	Logical operator	10.18
\$A_	Current general data	System variable	10.20 10.32
\$AA_	Current axis-specific data	System variable	10.20 10.32

Table 10-12 Overview of statements, continued

Statement	Meaning	Information/value range	Section
\$AC_	Current general data	System variable	10.20 10.32
\$P_	Programmed data	System variable	10.20
\$Rn_		System variable	10.20 10.32
\$VA_		System variable	10.20 10.32
:	Block number, main block		10.1.3
/	Skip block		10.1.3
+	Addition	Arithmetic operator	10.18
-	Subtraction	Arithmetic operator	10.18
*	Multiplication	Arithmetic operator	10.18
/	Division	Arithmetic operator	10.18
=	Assignment	Arithmetic operator	10.18
= =	Equal to	Relation operator	10.18
< >	Not equal to	Relation operator	10.18
>	Greater than	Relation operator	10.18
<	Less than	Relation operator	10.18
> =	Greater or equal to	Relation operator	10.18
< =	Less than or equal to	Relation operator	10.18

Note: For a complete list of **all statements** (including the statements not available on the FM 357-2) please refer to the online help of the FM 357-2.



Application Example

11

Chapter overview

Section	Title	Page
11.1	On-the-fly curve table switching	11-1
11.2	Conveyor tracking	11-6

11.1 On-the-fly curve table switching

Curve tables in conjunction with master value coupling are used in many areas of the processing industry. In many cases, the continuous material flow must not be interrupted due to the process.

Any technologically conditioned changes in the curve table must be carried out while the process is running.

The following application example demonstrates the possibility of a curve table switching on the fly without interrupting the master value coupling, as well as of the motion of master and slave axis.

Description of functions

The curve table is stored in a separate subroutine CONTOUR.SPF and can be parameterized there. The parameters can be changed, e.g. via operation (OP) while the program is executed.

Another subroutine FLY_CTAB.SPF is provided to realize the on-the-fly switching between two curve tables. The program must be called after the geometry has been changed or until the master value coupling is activated for the first time.

The curve table is prepared in the subroutine from CONTOUR.SPF. Then, it is switched alternately from the active to the newly prepared curve table. The motions of master and slave axes are not interrupted during this process.

FLY_CTAB.SPF can be adapted to the specific conditions of the plant by way of parameterization.

Supplementary conditions

The synchronism of master and slave axes must be maintained when switching. This requires a table section of the same geometry in both tables (synchronism area).

If a switching process is not yet executed, the control system will wait before calling FLY_CATB.SPF again.

Program description

Contour subroutine

CONTOUR.SPF ; Contour subroutine; to be created by the user
; parameterized; can thus be modified via condition
; (e.g. OP)

; Example:

```
N10 X=R10 Y=R11
N20 X=R20 Y=R21
N30 X=R30 Y=R31
N100 M17
```

Subroutine for on-the-fly table switching

FLY_CTAB.SPF ; Program for on-the-fly table switching

; **Parameterization by user**

; 1. Curve table

N10 DEF INT CTAB_NR ; No. of first curve table

N15 DEF INT CTAB_TYP ; Type of coupling

; 2. Master axis position for switching the synchronism range

N20 DEF REAL LA_POS1 ; Lower position

N25 DEF REAL LA_POS2 ; Upper position

; 3. Definition of master and slave axes:

N30 DEF AXIS LAX

N35 DEF AXIS FAX

; 4. Name of contour subroutine

N40 DEF STRING[7] CONT_UP = "CONTOUR"

; 5. Internal variable and synchronized action

N45 DEF INT SYNACT_NR ; Free static synchronized action

N55 DEF INT M_CTAB_NEXT ; \$AC_MARKER No. table no.

N50 DEF INT M_IS_FLY ; \$AC_MARKER No. Set status

; with first call to zero!

```

; Wait – table switching carried out?
N60 WHENEVER $AC_MARKER[M_IS_FLY] > 0 DO RDISABLE
N65 G4 F0.001

; Define new table no.
N70 IF ($AC_MARKER[M_CTAB_NEXT] == CTAB_NR + 1)
N75 $AC_MARKER[M_CTAB_NEXT] = CTAB_NR
N80 else
N85 $AC_MARKER[M_CTAB_NEXT] = CTAB_NR + 1
N90 endif

; Table definition
N95 CTABDEF (FAX, LAX, $AC_MARKER[M_CTAB_NEXT], CTAB_TYP)
N100 CALL CONT_UP ; call of contour subroutine
N105 CTABEND

; Synchronized action: on-the-fly switching of the curve table
N110 $AC_MARKER[M_IS_FLY] = 1
N115 ID=SYNACT_NR WHEN
      ($AA_IM[LAX] >= LA_POS1) AND ($AA_IM[LAX] <= LA_POS2) DO
      LEADOF(FAX,LAX) LEADON(FAX,LAX, $AC_MARKER[M_CTAB_NEXT])
      $AC_MARKER[M_IS_FLY] = 0

N120 G4 F0.001

N125 M17

```

Example

Depending on R50, on-the-fly switching is to be carried out between three curve tables. Master axis A is a modulo axis traversing at high velocity. The slave axis X (linear axis) is coupled to the A axis via a periodical curve table.

Contour: CONTOUR.SPF

```

N05 CSPLINE
N10 A=R1 X=R2
N11 A=R3 X=R4
N12 A=R5 X=R6
N13 A=R7 X=R8
N14 A=R9 X=R10
N15 M17

```

Parameterization of FLY_CTAB.SPF

```

; Program for on-the-fly table swithcing

; Parameterization by user
; 1. Curve table
N10 DEF INT CTAB_NR = 1           ; No. of first curve table
N15 DEF INT CTAB_TYP = 1         ; Type of coupling

; 2. Master axis range for siwthcing the snychronism range
N20 DEF REAL LA_POS1 = 0         ; Lower position
N25 DEF REAL LA_POS2 = 50        ; Upper position

; 3. Definition of master and slave axes
N30 DEF AXIS LAX = A
N35 DEF AXIS FAX = X

; 4. Name of contour subroutine
N40 DEF STRING[7] CONT_UP = "CONTOUR"

; 5. Internal variable and synchronized action
N45 DEF INT SYNACT_NR = 4         ; Free static synchronized action
N50 DEF INT M_IS_FLY = 1         ; $AC_MARKER No. status
N55 DEF INT M_CTAB_NEXT = 2      ; $AC_MARKER No. table no.
...

```

Main program FLY_CYCLE.MPF

```

; Init
N10 $AC_MARKER[1] = 0
N15 R50=1 R51=0           ; Start with contour 1

; Contour 1
N20 R1=0 R2=0
N21 R3=50 R4=1
N22 R5=180 R6=3.5
N23 R7=310 R8=1
N24 R9=359.9 R10=0

; Initial position
N30 PRESETON(A1, 0)
N35 POS[X]=0 FA[X]=1000

; Endless motion of master axis
N40 DO MOV[A]=1 FA[A]=30000

FLY_CYC:

```

```

; Wait for change in R50
N50 WHENEVER $R50== $R51 DO RDISABLE
N55 G4 F0.001                ; output block

; Accept status in R51
N60 R51 = R50

; Change contour depending on R50
N75 R6 = 3.5                ; Contour 1
N80 IF R50 == 2
N81 R6 = 2.5                ; Contour 2
N82 ENDIF
N85 IF R50 == 3
N86 R6 = 1.5                ; Contour 3
N87 ENDIF

; Advance R50 automatically (example)
N90 IF R50 >=3
N91 R50=1
N92 ELSE
N93 R50 = R50 +1
N94 ENDIF

; Call of subroutine for on-the-fly table switching
N100 FLY_CTAB

N110 GOTOB FLY_CYC
N120 M30

```

Using the sample program, the slave axis carries out the following motion cycle:

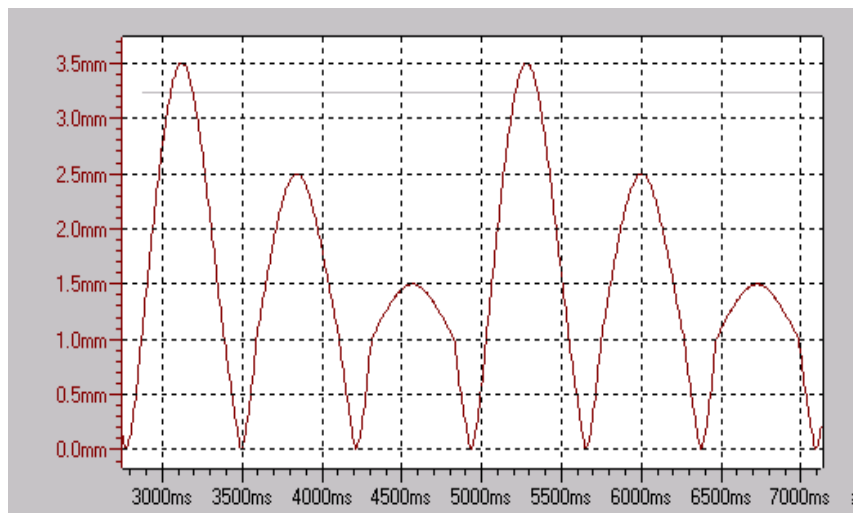


Fig. 11-1 Beispielprogramm Bewegungszyklus Folgeachse

11.2 Conveyor tracking

General

Conveyor Tracking is a method for processing moved workpieces.

The existing scope of functions of the FM357-2 allows to realize such machining tasks.

The following application example describes how to apply a figure during the transport motion.

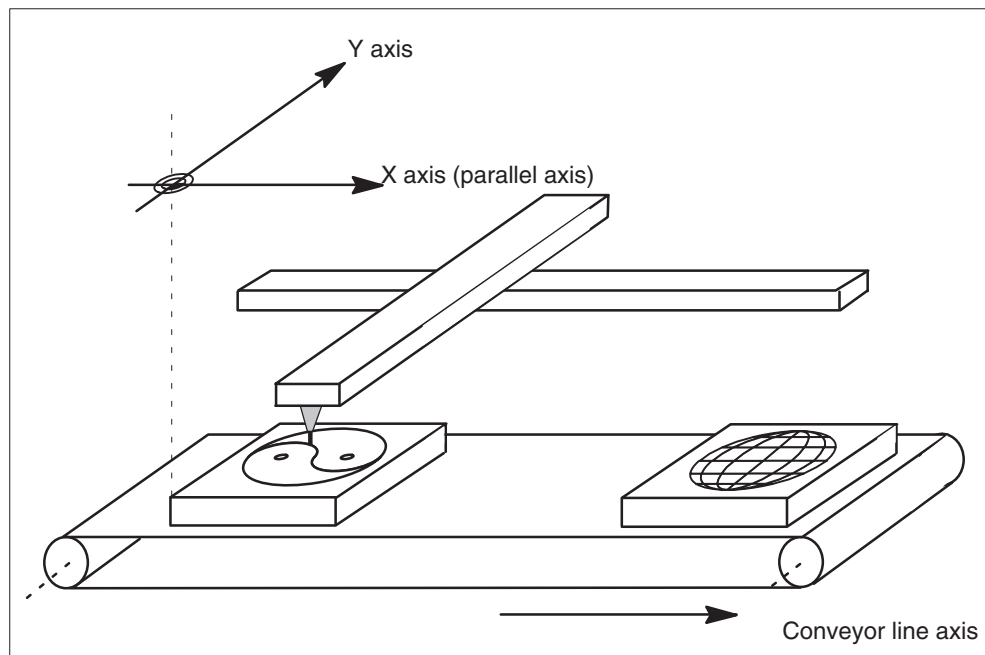


Fig. 11-2 Machining example

Principle of functioning

During machining, the parallel axis is coupled to the conveyor axis via an electronic gear using coupling factor 1:1.

The parallel axis as the following axis of the EG is additionally traversed via the NC program. This motion is programmed in the Cartesian coordinate system, since the motion component of the transport motion is compensated by the electronic gear (EG).

The workpiece position is acquired by a sensor connected to sensing probe input 1 or 2. Depending on the measured value determined, the actual value of the conveyor line axis is set for each workpiece anew. Thus, the reference to the workpiece is made in conjunction with the specification of the synchronized position when activating on the EG.

Conveyor line axis

Either

- a controlled axis of the FM or
- a single drive with encoder (servo axis in the follow-up mode)

can be used as a conveyor line axis.

Supplementary conditions

The following supplementary conditions result from the overlay of following axis motion and motion in the NC program:

- The status “Target range coarse or fine” will not be reached until the parallel axis is moved with the EG involved. Therefore, the G603 or G64 mode must be enabled in the NC program.
- The axial velocity component of the conveyor line motion will reduce the possible maximum velocity of the parallel axis for motions in the directions of the line motion.

Sample program

```

CONVEYOR.MPF          ; Main program
; Axse:
;   X Parallel axis to the conveyor line axis
;   Y Transverse axis to the conveyor line axis
;   A Conveyor line axis

; Parameterization

; Conveyor line axis
N10 DEF INT TR_TYP=0   ; Type of the conveyor line axis:
                       ; 0 external encoder (servo axis in the follow-up
                       ; mode)
                       ; 1 controlled FM axis

N30 DEF INT TR_VELO=1000 ; Velocity of the conveyor line axis (only type 1)

; Positions
N40 DEF REAL START_X=0 ; Start position of the X axis
N50 DEF REAL START_Y=0 ; Start position of the Y axis
N60 DEF REAL SYN_X=10  ; Synchronized position of the X axis
N70 DEF REAL SYN_A=10  ; Synchronized position of the A axis
N90 DEF REAL DIST_C=0  ; Distance between workpiece zero point and
                       ; measuring point

```

```

; Synchronized actions
N100 DEF INT SA_MEA1=1 ; 1st free static synchronized action "Measuring"
N110 DEF INT SA_MEA2=1 ; 2nd free static synchronized action "Measuring"

; Flags
N110 DEF INT M_ACT=1 ; AC_MARKER: Measuring job active
N120 DEF INT M_VAL=2 ; AC_MARKER: Measuring value valid
N130 DEF INT M_SET=3 ; AC_MARKER: Actual value of line axis set

; Machining program
N150 DEF STRING[6] PART_UP = TEIL_1

; Sensing probe number
N160 DEF INT MT_MR = 1

; Start initialization

; Flags
N200 $AC_MARKER[M_ACT] = 0
N210 $AC_MARKER[M_VAL] = 0
N220 $AC_MARKER[M_SET] = 0

; Deselect electronic gear (if activated)
N230 EGOFS(X)
N240 EGDEL(X)
N250 EGDEF(X,A,0)

; Activate static synchronized action; measuring with the falling edge
N200 ID=SA_MEA1
EVERY ($A_PROBE[MT_NR]==1) AND ($AC_MARKER[M_ACT]==0) DO
MEAWA[B]=(3,-MT_NR) $AC_MARKER[M_ACT]=1
N210 ID=SA_MEA2
EVERY ($AA_MEA[A]==0) AND ($AC_MARKER[M_ACT]==1) DO
$AC_MARKER[M_ACT]=0 $AC_MARKER[M_VAL]=1

; Conveyor line axis motion ON
N220 IF TR_TYP==1
N230 WHEN TRUE DO MOV[A]=1 FA[A]= TR_VELO
N240 ENDIF

ZYKLUS:

; Wait for measured value, set actual value of conveyor line axis, approach starting
position
N250 EVERY $AC_MARKER[M_VAL]==1 DO
PRESETON(B, ($AA_IM[A]-$AA_MM1[A]+DIST_C) )
$AC_MARKER[M_VAL]=0

```

```
N260 WHENEVER $AC_MARKER[M_SET]==0 DO RDISABLE
N270 G90 G0 G60 X=START_X Y=START_Y

; Activate electronic gear; coupling factor 1:1
N300 STOPRE
N310 WHEN TRUE DO $AC_MARKER[M_SET]=0
N320 EGONSYN(X, "FINE" , SYN_X, A, SYN_A, 1, 1)

; Call machining program
N400 G603
N410 CALL PART_UP

; Deactivate EG, go to cycle start
N500 EGOFS(X)
N510 GOTOB ZYKLUS

N600 M30
```

```
TEIL_1.SPF ; Machining program
```

```
; Simple contour consisting of several arcs
```

```
N10 G90 G1 X10 Y10 F3000
```

```
N20 G2 X30 Y10 I=10 J=0
```

```
N30 G3 X50 Y10 I=10 J=0
```

```
N40 G3 X50 Y10 I=-20 J=0
```

```
N50 RET
```



Troubleshooting

Section overview

Section	Title	Page
12.1	LED indicators	12-2
12.2	Error messages and their effect	12-6
12.3	Error list	12-8

General

The FM 357-2 multi-axis module has a diagnostic system for detecting:

- Faults on the module and the connected I/Os
- Faults which occur during operation of the module
- Faults which occur during communication with the CPU

Localization of faults

The following tools are available for localizing faults on the FM 357-2:

- Status and error display on LEDs
- Error messages sent to the CPU and to HMI units (human-machine interfaces)
- Error messages of the CPU via FC 5, FC 22

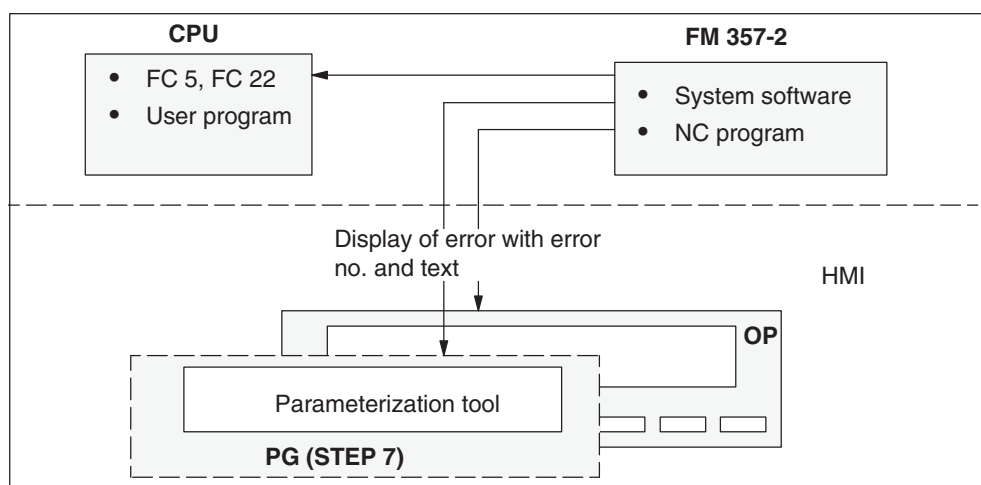


Fig. 12-1 Troubleshooting

Error messages

Error messages of the FM 357-2 are signalled to the user/CPU and identified by an error number and an error text.

You can read the error message with the error number and error text using the parameterization software or an OP (e.g. OP 17). The integrated help system provides information on how to remedy errors.

Please refer to the following manuals for error handling on the S7-300 system:

- Programming Manual *System Software for S7-300/400; Draft Program* (OB types, diagnostic alarm)
- Reference manual *System Software for S7-300/400; System and Standard Functions*
- User Manual *Basic Software for S7 and M7, STEP 7*

12.1 LED indicators

Status and error displays

The FM 357-2 uses the following status and error displays:

- **SF** – Group error
- **BAF** – Battery error
- **DC5V** – Logic supply
- **DIAG** – Diagnostics

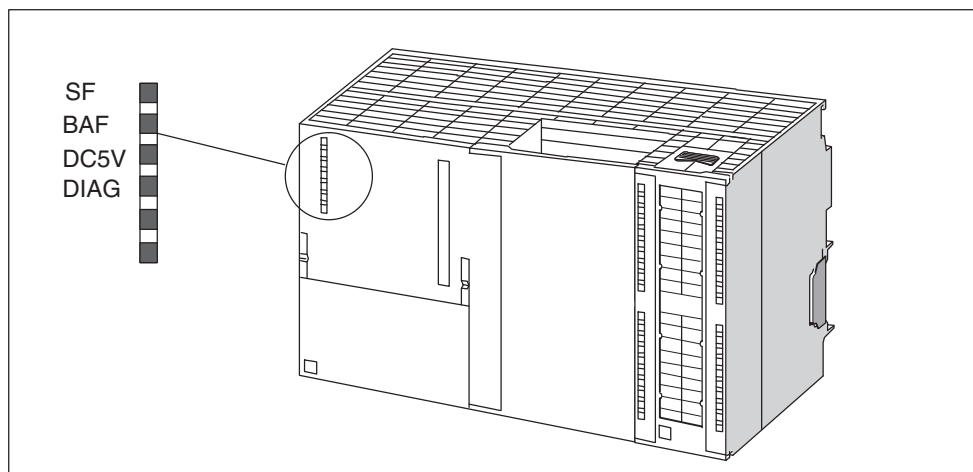


Fig. 12-2 Status and error displays of the FM 357-2

Meaning of status and error displays

The status and error displays are explained here in the order in which the LEDs are arranged on the FM 357-2.

Table 12-1 Status and error displays

Display	Meaning	Explanation
SF (red) LED ON LED flashing	Group error Incorrect license or no license Serious error during firmware update	This LED indicates an error condition in the FM 357-2. To eliminate the error you may need to: <ul style="list-style-type: none"> • Switch off/on • Recommission the system • Update the firmware • Replace the FM 357-2 <ul style="list-style-type: none"> • Insert a memory card with the correct license • Update the firmware from a memory card with license <ul style="list-style-type: none"> • Repeat the firmware update
BAF (red) LED – ON	Battery error	When the LED is lit continuously, it is possible that the data in the backup memory may be erased. After you change the battery you will need to start up the system again.
5 V DC (green) LED ON LED – OFF	Logic supply	This LED indicates that the logic supply is ready for operation. If not illuminated, this may indicate one of the following conditions: <ul style="list-style-type: none"> • Load current supply does not meet specifications, • Module incorrectly connected or • Module defective.
DIAG (yellow) LED flashing	Diagnostics	This LED indicates various diagnostic states If this LED flashes (0.5 Hz), it is indicating a battery warning. You need to change the battery.

The following LED display indicates that the FM 357-2 is operating without an error:

- LED SF: OFF
- LED BAF: OFF
- LED DC5V: ON
- LED DIAG: Fast regular flashing = FM sign of life (3 Hz)

Error displays

Table 12-2 gives you an overview of the LED error displays on the FM 357-2. Legend:

1	=	ON
0	=	OFF
n/1	=	Periodic flashing n times
x	=	No meaning for the error described

Table 12-2 Summary of LED error displays

SF	BAF	DC5V	DIAG ¹⁾	Meaning	Note
Hardware/power supply					
x	x	0	x	The internal 5 V power supply or the 24 V power supply has failed.	Replace the FM 357-2
License monitoring					
1/1	x	x	x	No valid system software license available (no memory card or memory card with invalid license plugged in)	Insert a memory card with the correct license
Encoder monitoring					
1	x	x	1/1	Encoder supply failure	Check encoder and connection
Battery monitoring					
x	1	x	x	Battery failure	Scanning: during operation Remedy: Replace battery
x	0	x	0.5 Hz	Battery warning, voltage below threshold	Scanning: during operation Remedy: Change battery as soon as possible
Power-up: The recommended remedy for this error group is to update the firmware again. If the error state still remains, replace the FM 357-2.					
0	x	x	0	Power-up	Status display
1	x	x	1	Startup error (DRAM error, part of the boot software defective)	
0	x	x	1	The integrity of the backup parameter data can no longer be assured (battery failure; the battery was removed when the FM was switched off)	Power-up with Defaults
1	x	x	4/1	Error during system software boot (checksum error; part of the software defective or missing)	Please contact the SIEMENS AG hotline
0	x	x	7/1	Error during loading of the communication software within the system software (parameterization by S7-300 missing)	

1) An error is indicated if the "DIAG" flashes briefly three times and then flashes in the manner described for each error.

Table 12-2 Summary of LED error displays, continued

SF	BAF	DC5V	DIAG ¹⁾	Meaning	Note
0	x	x	8/1	Error during loading of the communication software within the system software (CPU is in STOP mode; no synchronization with basic program; basic program not available)	
0	x	x	9/1	Error during FM power-up (no synchronization with CPU)	
Software and hardware monitoring:					
1	x	x	2/1	Hardware watchdog has responded and tripped a module reset	
1	x	x	3/1	Software watchdog has responded	
1	x	x	5/1	An internal system software error has caused a processor exception situation	Please contact the SIEMENS AG hotline
1	x	x	6/1	A timing error in the system software has caused a level overflow (not enough time to run servo or interpolator).	
1	x	x	11/1	Access error on data read (local P bus)	Hardware or software error on a module on the local P bus of the FM 357-2
1	x	x	13/1	General hardware error (local P bus)	
1	x	x	15/1	Access error on data write (local P bus)	
1	x	x	Cycl. flashing	No ready signal (no FM READY)	
Firmware update:					
2 Hz	x	x	x	Error during firmware update from memory card to internal memory of FM (internal memory not cleared or data written incorrectly)	System software fault in internal memory (repeat the update from memory card)
1	x	x	3/1	Error during firmware update from memory card to internal memory of FM (no memory card inserted, system software on the card defective)	
Data backup on memory card:					
1	x	x	1/1	Error during data backup to memory card (backup parameter data defective; no memory card or invalid memory card inserted; system software in internal memory defective; memory card cannot be cleared or data cannot be written to card)	

1) An error is indicated if the "DIAG" flashes briefly three times and then flashes in the manner described for each error.

12.2 Error messages and their effect

General

The following error messages are displayed to the user:

Plant shutdown errors (user program)

- “FM ready” (see Section 4.8) user DB “FMx”, DBX30.7

The signal is cancelled:

- in the case of error, see LED error display
- in the case of system error
- in the case of error, see error list Table 12-3, with the “Effect”:

No ready signal

The “Ready” state can be restored only after elimination of the error and FM Restart or by switching the FM 357-2 power supply off and then on again.

An “FM restart” can be initiated manually using “Parameterize FM 357-2” (in the “Troubleshooting” window) or an OP.

- “Communication ready” (user DB “FMx”, DBX10.0)

The signal is cancelled

- in the case of a communication fault between CPU and FM 357-2
- when the module has not yet powered up
- when the FM 357-2 has a fault, diagnostic alarm

The error number is stored in “Basic function error” (user DB “FMx”, DBW4) by FC 22 or FC 5. The communication ready state must be restored as described for “FM ready”.

- “Channel ready” (user DB “FMx”, DBX126.5+n)

The signal is cancelled in the event of:

- A serious error (e.g. hardware errors of active encoders)
- A serious axis error
- Incorrect parameterization on the FM

The “channel ready” state can be restored (except in the case of a hardware) by a reset; otherwise follow the steps described for “FM ready”.

A reset can be initiated via the interface or manually using “Parameterize FM 357-2” (in the “Startup” window) or an OP.

Channel error messages of the FM

The error number can be read out with FB 2 (Section 6.3.4).

- “FM error active” (user DB “FMx”, DBX31.0)
Group error message for channels 1 to 4 (channel error)
- “Channel error” (user DB “FMx”, DBX126.6+n)
An error has occurred in the channel. The “channel error, program stopped” indicates whether NC program execution has been interrupted or aborted.
- “Channel error, program stopped” (user DB “FMx”, DBX126.7+n)
An error has occurred in the channel. NC program execution has been interrupted or aborted (machining standstill).

The error can be acknowledged by:

- Reset, user DB (“FMx”, DBX108.7+n)
- CANCEL, manually via OP or “Parameterize FM 357-2” (in “Troubleshooting” window) the initiated error acknowledgement.
- CANCEL as PI service by means of FB 4

See also the error list in Table 12-3

Errors which occur during axis control from the CPU

“CPU axis error” (user DB “AXy”, DBX67.1+m). For the message and error number, see Section 6.5.1

Errors which occur during independent axis control from the CPU

“Independent CPU axis error” (user DB, “AXy”, DBX69.2+m). For the message and error number, see Section 12.3.

Further error messages

See output parameters of FB 2 (Section 6.3.4), FB 3 (Section 6.3.5) and FB 4 (Section 6.3.6).

12.3 Error lists

General

This section describes errors on the FM 357-2, their cause and effect, and how to remedy them.

The errors are subdivided into number ranges.

- System error: Number range 1 000 to 1 999
- Diagnostic error: Number range 2 000 to 9 999
- Channel error: Number range 10 000 to 19 999
- Axis error: Number range 20 000 to 29 999
- Functional error: Number range 30 000 to 99 999

System errors

The following errors are system errors which indicate **internal error states**. These errors should not occur if you follow the instructions in this manual.

If you do however encounter them, then please make a note **of the error number and the internal system error number it contains** (displayed only on OP 17, PG/PC) and contact the

SIEMENS AG Hotline see Preface

Error no.

1 000	1 004	1 012	1 016	1 160
1 001	1 005	1 013	1 017	
1 002	1 010	1 014	1 018	
1 003	1 011	1 015	1 019	

Error list of frequently occurring errors

Table 12-3 below lists the following errors:

- Diagnostic errors
- Channel errors
- Axis errors
- Functional error

The error text may contain variables. These are identified by the % symbol and a number.

Example: %1 = channel number, %2 = slot number, %3 = block number,
 %4 = axis name

Table 12-3 Error list

Error no.	Error message, error analysis and remedy	
Diagnostic errors		
1 100	No license or incorrect license	
Cause	No memory card or a memory card without valid firmware (license) is plugged in.	
Effect	<ul style="list-style-type: none"> • Cyclic flashing of red “SF”LED • All axes can only be operated in simulation mode 	
Elimination	Plug in a memory card with valid firmware (license)	
Acknowl.	FM restart	
2 000	Sign of life monitoring: CPU not active	
Cause	<p>The CPU must output a sign of life within a defined time span (100 ms) . If the sign of life is not detected, an error is output (see Section. 6.1.6).</p> <p>The sign of life is a counter value on the internal FM/CPU interface. The value is incremented by the CPU with the 10 ms time alarm. The FM 357-2 also checks cyclically whether the counter state has changed.</p>	
Effect	<ul style="list-style-type: none"> • No ready signal (no FM READY) • Start disable • Stop 	
Elimination	Determine the cause of the error on the CPU, and remedy (analyze USTACK. If the timeout was generated not by a CPU stop, but by a loop in the user program, there is no entry in the USTACK).	
Acknowl.	FM restart	
2 001	CPU has not started up	
Cause	The CPU must output a sign of life within 50 s after the power-up.	
Effect	<ul style="list-style-type: none"> • No ready signal (no FM READY) • Start disable • Stop 	
Elimination	Determine the cause of the error on the CPU (loop or Stop in user program), and remedy.	
Acknowl.	FM restart	
2 100	Battery warning threshold reached	
Cause	The battery voltage has reached the preliminary warning threshold of the monitoring system. This is 2.7 to 2.9 V (rated voltage of battery is 3.0 to 3.1 V at 950 mAh).	
Effect	Warning	
Elimination	The battery should be replaced within the next 6 weeks. A high power consumption on the backed-up RAM can subsequently cause the voltage to fall below the error limit of 2.4 to 2.6 V.	
Acknowl.	Clear error with the CANCEL key.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Diagnostic errors		
2 101	Battery alarm	
Cause	The monitoring system detected a low battery voltage (2.4 to 2.6 V) during cyclical operation.	
Effect	Warning	
Elimination	If the battery is changed without interrupting the power supply, there is no loss of data. You can thus continue production without initiating additional measures. (A buffer capacitor on the FM 357-2 maintains the supply voltage for at least 30 min – the battery can be replaced within this time even then the control is switched off).	
Acknowl.	Clear error with the CANCEL key.	
2 130	The power supply of encoder 5 V or 24 V has failed	
Cause	Failure of the power supply of the encoders (5 V/24 V).	
Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • FM switches to follow-up mode • The axes are no longer synchronized with the actual machine value (reference point). 	
Elimination	Check the encoders and cables for a short-circuit Check the power supply.	
Acknowl.	FM restart	
3 000	EMERGENCY STOP	
Cause	The emergency stop request is active on the FM/CPU interface.	
Effect	<ul style="list-style-type: none"> • No ready signal (no FM READY) • Start disable • Stop • FM switches to follow-up mode • Cancellation of controller enable signal to drive 	
Elimination	<ul style="list-style-type: none"> • Check whether an emergency stop cam was triggered or an emergency stop button was pressed. Check the user program. • Cancel the emergency stop and acknowledge the emergency stop over the FM/CPU interface. 	
Acknowl.	Clear the error with “Reset”.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Diagnostic errors		
4 060	Standard machine data loaded	
	Cause	Power-up with default values as a result of: <ul style="list-style-type: none"> • Operator action (e.g. startup switch) • Loss of retentive data • New software version • MD buffer voltage lost
	Effect	Warning
	Elimination	After you load the default MD, you must enter/load the individual MD for the specific plant.
	Acknowl.	Clear the error with "Reset".
4 070	Normalizing machine data has been altered	
	Cause	The controller operates internally with physical quantities (mm, degrees, s for paths, velocities, acceleration, etc.). The following MD causes these quantities to be converted: <ul style="list-style-type: none"> • MD "internal system of measurement" changed • MD "axis type" changed
	Effect	Warning
	Elimination	None
	Acknowl.	Clear error with the CANCEL key.
4 112	Servo cycle changed to %1 ms	
	Cause	%1 = String (new servo cycle) PROFIBUS-DP drive was parameterized. The FM must adapt the servo cycle to the DP cycle.
	Effect	Warning
	Remedy	–
	Acknowl.	Clear the error with "Reset".
4 113	System clock changed to %1 ms	
	Cause	%1 = String (new system clock) PROFIBUS-DP drive was parameterized. The FM must adapt the system clock (internal value) to the DP cycle.
	Effect	Warning
	Remedy	–
	Acknowl.	Clear the error with "Reset".
4 114	Error in DP clock of SDB1000	
	Cause	The DP clock of the externally loaded SDB is faulty or cannot be set.
	Effect	Warning
	Remedy	Eliminate the error in the SDB.
	Acknowl.	Switch off the FM and back on again.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Diagnostic errors		
4 240	Calculation time exceeded in the IPO or position controller plane, IP %1	
	Cause	%1 = internal address of the program point The current setting of IPO and servo cycle results in exceeding of the calculation time in one of the two planes.
	Effect	<ul style="list-style-type: none"> • No readiness message (no FM READY) • Start disable • Stop • The FM switches to follow-up mode.
	Elimination	<ul style="list-style-type: none"> • Increase the servo or IPO cycle. • Reduce the load in the servo or IPO plane (e.g. by reducing the number of active synchronized actions).
	Acknowl.	Switch of the FM and back on again.
4 290	Sign of life monitoring: Local P bus not alive	
	Cause	No sign of life on local P bus
	Effect	<ul style="list-style-type: none"> • No ready signal (no FM READY) • Start disable • Stop
	Elimination	Check the hardware and parameters
	Acknowl.	FM restart
4 291	Failure of module in local P bus slot %1, error codes: %2 %3 %4	
	Cause	%1 = slot number; %2, %3, %4 = error code The module in the specified slot has signalled a diagnostic alarm (for error message, see Programming Manual <i>System Software for S7-300/400; Draft Program</i>)
	Effect	<ul style="list-style-type: none"> • No ready signal (no FM READY) • Start disable • Stop
	Elimination	Check the hardware and S7-300 configuration
	Acknowl.	FM restart
6 020	Machine data changed – memory mapping changed	
	Cause	Memory-configuring data have been changed (e.g. number of R parameters, number of curve tables). The battery-backed memory was to be reconfigured. Loss of the retentive data
	Effect	Warning
	Elimination	–
	Acknowl.	Clear the error with “Reset”.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Diagnostic errors		
6 030	Limit of user memory has been adapted	
	Cause	Loading of the default MD, the controller determines the actual existing memory
	Effect	Warning
	Elimination	None
	Acknowl.	Clear the error with "Reset".
Channel error		
10 203	Channel %1 Start not possible without reference point	
	Cause	%1 = channel number Start was activated in MDI or Automatic mode, and at least one axis which has to be referenced has not reached its reference point.
	Effect	Stop NC block preparation
	Elimination	Reference the axis.
	Acknowl.	Clear the error with "Reset".
10 610	Channel %1 axis %2 not stopped	
	Cause	%1 = channel number; %2 = axis name An axis was positioned across several NC blocks with the POSA instruction. The programmed target position was not yet reached ("target range fine"), when the axis was reprogrammed. Example: N100 POSA[U]=100 : N125 X... Y... U... ; e.g.: U axis still traversing from N100!
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Check and correct the NC program, e.g. use the WAITP keyword to inhibit the block change until the positioning axes have also reached their target positions. Example: N100 POSA[U]=100 : N125 WAITP[U] N130 X... Y... U...
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 620	Channel %1 block %3 axis %2 at software limit switch %4	
	Cause	%1 = channel number; %2 = axis name; %3 = block number, label; %4 = string During the traversing movement, passing of the software limit switch in the direction indicated is detected.
	Effect	Start disable
	Elimination	Change the NC program or check the value of the interface signal "Position of the CPU axis" (user DB "AXy, DBB52+m).
	Acknowl.	Clear the error with "Reset".
10 621	Channel %1 axis %2 is resting on software limit switch %3	
	Cause	%1 = channel number; %2 = axis name; %3 = string The specified axis is resting on the specified software limit switch. An attempt was made to travel beyond the limit.
	Effect	No axis movement towards the limit
	Elimination	Move the axis within the permissible traversing range
	Acknowl.	Error remedy
10 631	Channel %1 axis %2 is resting on working area limitation %3	
	Cause	%1 = channel number; %2 = axis name; %3 = string The specified axis is resting on the specified working area limitation. An attempt was made to travel beyond the limit.
	Effect	No axis movement towards the limit
	Elimination	Move the axis within the permissible working area
	Acknowl.	Error remedy
10 650	Channel %1 Incorrect gantry machine data axis %2 error no. %3	
	Cause	%1 = channel number; %2 = axis name; %3 = error no. The parameter has been set to an incorrect value. For further information, please see error no. <ul style="list-style-type: none"> • Error no. = 1 → Either an incorrect gantry grouping has been entered or the synchronized axis name is incorrect. • Error no. = 2 → Leading axis has been specified more than once
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop
	Elimination	Correct machine data: "Leading axis", "Synchronized axis"
	Acknowl.	Clear error with "FM restart".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 651	Channel %1 Gantry grouping undefined %2	
	Cause	%1 = channel number; %2 = reason An undefined gantry grouping has been set in the parameter. The gantry grouping and rejection reason can be found in the transfer parameter. The transfer parameter comprises the following elements: %2 = error designation + gantry grouping (XX). <ul style="list-style-type: none"> • %2 = 10XX → No leading axis declared • %2 = 20XX → No synchronized axis declared e.g. error no. 1001 = No leading axis declared, grouping 1.
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop
	Elimination	Correct machine data: "Leading axis", "Synchronized axis"
	Acknowl.	Clear error with "FM restart".
10 652	Channel %1 axis %2 Gantry warning threshold exceeded	
	Cause	%1 = channel number; %2 = axis name The gantry synchronized axis has exceeded the warning limit set in parameter "Limit value for warning".
	Effect	Warning
	Elimination	<ul style="list-style-type: none"> • Check axis (uneven mechanical movement?) • Parameter is set incorrectly (limit value for warning). Changes to this parameter are effective after a Reset.
	Acknowl.	Error remedy
10 653	Channel %1 axis %2 Gantry trip limit exceeded	
	Cause	%1 = channel number; %2 = axis name The gantry synchronized axis has exceeded the error limit (actual value tolerance) set in parameter "Trip limit".
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	<ul style="list-style-type: none"> • Check axis (uneven mechanical movement?) • Parameter is set incorrectly (trip limit). If the parameter is changed, an FM restart is necessary.
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 654	Channel %1 Waiting for synchronization start of gantry group %2	
	Cause	%1 = channel number; %2 = gantry group This error message appears when the axes are ready to be synchronized. The gantry grouping can now be synchronized.
	Effect	Note
	Elimination	None
	Acknowl.	The message disappears after the grouping has been synchronized.
10 655	Channel %1 Synchronization of gantry group %2 in progress	
	Cause	%1 = channel number; %2 = gantry group None
	Effect	Note
	Elimination	None
	Acknowl.	The message disappears after the grouping has been synchronized.
10 700	Channel %1 block %2 protection zone %3 violated in Automatic or MDI mode	
	Cause	%1 = channel number; %2 = block number; %3 = protection zone number The workpiece-related protection zone has been violated.
	Effect	Stop
	Elimination	Correct the program or, if the axis can be moved freely, cross the protection zone with "Start".
	Acknowl.	Clear error with "Start" or "Reset".
10 702	Channel %1 Protection zone %2 violated in Jogging or Incremental relative mode	
	Cause	%1 = channel number; %2 = protection zone number The workpiece-related protection zone has been violated.
	Effect	Stop
	Elimination	Move within the permissible traversing range or, if the axis can be moved freely, cross the protection zone with "Start".
	Acknowl.	Error remedy
10 706	Channel %1 Protection zone %2 reached with axis %3 in Jogging or Incremental relative mode	
	Cause	%1 = channel number; %2 = protection zone number; %3 = axis name The specified axis has reached the workpiece-related protection zone.
	Effect	Stop
	Elimination	Move within the permissible traversing range or, if the axis can be moved freely, cross the protection zone after enabling by the CPU.
	Acknowl.	Error remedy

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 720	Channel %1 block %3 axis %2 software limit switch %4	
	Cause	%1 = channel number; %2 = axis name; %3 = block number, label; %4 = string The programmed path violates the software limit switches active for the axis. The error is activated when the NC program block is prepared.
	Effect	Stop
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
10 721	Channel %1 block %3 axis %2 software limit switch %4	
	Cause	%1 = channel number; %2 = axis name; %3 = block number, label; %4 = string The motion you want to perform will violate the axis for the software limit switch. The error is activated when preparing approach or residual blocks with REPOS.
	Effect	Stop
	Elimination	Correct the NC program.
	Acknowl.	Clear the error with "Reset".
10 730	Channel %1 block %3 axis %2 working area limitation %4	
	Cause	%1 = channel number; %2 = axis name; %3 = block number, label; %4 = string The programmed path violates the working area limitation active for the axis. The error is activated when the NC program block is prepared.
	Effect	Stop
	Elimination	<ul style="list-style-type: none"> • Correct the NC program • Change the working area limitation
	Acknowl.	Clear the error with "Reset".
10 731	Channel %1 block %3 axis %2 work area limitation %4	
	Cause	%1 = channel number; %2 = axis name; %3 = block number, label; %4 = string The motion you want to perform will violate the axis for the work area limitation. The error is activated when preparing approach or residual blocks with REPOS.
	Effect	·Stop
	Elimination	<ul style="list-style-type: none"> • Correct the NC program. • Change the work area limitation.
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 800	Channel %1 block %3 axis %2 is not a geometry axis	
Cause	%1 = channel number; %2 = axis name; %3 = block number, label A special axis has been programmed with an illegal interpolation type for special axes (e.g. G2/G3). A geometry axis was moved as a positioning axis. An offset with a rotation component was programmed for the axis in this state.	
Effect	Stop	
Elimination	Correct the NC program	
Acknowl.	Clear the error with "Reset".	
10 860	Channel %1 block %2 Feedrate not programmed	
Cause	%1 = channel number; %2 = block number, label An interpolation type other than G00 (rapid traverse) is active in the displayed block. The F value is missing.	
Effect	Stop NC block preparation	
Elimination	Program the path feed in mm/min at address F .	
Acknowl.	Clear the error with "Reset".	
10 861	Channel %1 Block %3 Axis velocity for positioning axis %2 Zero is programmed	
Cause	%1 = Channel number; %2 = Axis name; %3 = Block number, label The parameter "Positioning velocity" is defined with zero by default, and a positioning axis was programmed with feedrate zero.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop of NC block preparation 	
Elimination	<ul style="list-style-type: none"> • Define an appropriate feedrate for the parameter "Positioning velocity" • Program the feedrate 	
Acknowl.	Use "Start" to clear the error.	
10 881	Channel %1 block %2 Overflow of the local block buffer with chamfers or radii	
Cause	%1 = channel number; %2 = block number, label The number of empty blocks programmed between 2 blocks containing contour elements and to be connected via a chamfer or a radius (CHF, RND) is so large that the internal buffer memory is too small.	
Effect	<ul style="list-style-type: none"> • Start inhibit • Stop of NC block preprocessing 	
Elimination	Correct the NC program.	
Acknowl.	Use "Start" to clear the error.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 890	Channel %1 block %2 Overflow of local block buffer when calculating splines	
Cause	%1 = channel number; %2 = block number, label The maximum permissible number of empty blocks (blocks without motion) has been exceeded.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknowl.	Clear the error with "Reset".	
10 891	Channel %1 block %2 Multiplicity of node is greater than its order	
Cause	%1 = channel number; %2 = block number, label Node distance PL (node = point on spline at which 2 polynomials meet) on a B spline has been programmed with zero too often in succession (i.e. the "multiplicity" of a node is too great). On a quadratic B spline, the node distance may not be specified as zero more than twice in succession and, on a cubic B spline, no more than three times.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 	
Elimination	Program node distance PL = 0 only so many times in succession as corresponds to the degree of the B spline used.	
Acknowl.	Clear the error with "Reset".	
10 913	Channel %1 block %2 Negative feed profile is omitted	
Cause	%1 = channel number; %2 = block number, label The specified feed profile is negative in places. Negative path feeds are not permissible. The feed profile will be ignored. The specified feed block end value will be applied over the entire block.	
Effect	Warning	
Elimination	Check and alter NC program	
Acknowl.	Clear error with the CANCEL key.	
10 940	Channel %1 block %2 curve table %3: cannot be deleted/overwritten	
Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table The curve table can be deleted only if it is not active in a coupling.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 	
Elimination	All couplings which are using the curve table to be deleted must be deactivated.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 941	Channel %1 block %2 curve table %3: NC program memory full	
Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table Definition of the curve table has used up all the available NC memory.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 	
Elimination	Delete any curve tables which you no longer require or reconfigure the assigned memory space for curve tables. You must then define the curve table again. See following parameter: No. of curve tables No. of curve segments No. of curve table polynomials	
Acknowl.	Clear the error with "Reset".	
10 942	Channel %1 block %2 curve table %3: Illegal instruction on defining	
Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table Various illegal instruction sequences used to define the curve table have given rise to this error. For example, it is not permissible to terminate definition of a table with M30 before instruction CTABEND has been programmed.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 	
Elimination	Correct the NC program and restart.	
Acknowl.	Clear the error with "Reset".	
10 943	Channel %1 block %2 curve table %3: Direction change of lead value in the block not allowed	
Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table The preconditions for converting a programmed contour into a curve table have not been fulfilled in this block.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 	
Elimination	Correct the NC program and restart.	
Acknowl.	Clear the error with "Reset".	
10 945	Channel %1 block %2 curve table %3: Illegal coupling of axes	
Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table No axis coupling may be programmed for the master and slave axes programmed in CTABDEF.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
10 946	Channel %1 block %2 curve table %3: No contour defined	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table No movement for the slave axis has been programmed between CTABDEF and CTABEND. A curve table without a contour is not legal.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
10 947	Channel %1 block %2 curve table %3: Discontinuous contour	
	Cause	%1 = channel number; %2 = block number, label %3 = no. of curve table The contour in a curve table must be continuous. Discontinuities may arise, for example, as a result of plane switchovers (G17 → G18).
	Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
10 948	Channel %1 block %2 curve table %3: Position jump at edge of period	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table A periodic curve table has been defined in which the position of the slave axis at the table edge differs from the position at the table start.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
10 949	Channel %1 block %2 curve table %3: No lead axis movement	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table A movement of the slave axis has been programmed without a corresponding lead axis motion.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
10 951	Channel %1 block %2 curve table %3: Follow-up value is zero	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table A table of type 2 has been declared, but a table of type 1 has been specified.
	Effect	Warning
	Elimination	Correct NC program if necessary (curve table definition).
	Acknowl.	Clear error with the CANCEL key.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy					
Channel error						
12 050	Channel %1 block %2 address %3 does not exist					
Cause	%1 = channel number; %2 = block number, label; %3 = address The name of the address (e.g. X, U, X1) is not defined.					
Effect	<ul style="list-style-type: none"> • Start disable • Stop NC block preparation 					
Elimination	Correct the NC program					
Acknowl.	Clear the error with “Reset”.					
12 060	Channel %1 block %2 Same G group programmed repeatedly					
Cause	<p>%1 = channel number; %2 = block number, label</p> <p>The G functions which can be used in the NC program are subdivided into groups. Only one G function from each group can be programmed. The functions within a group are mutually exclusive.</p> <p>The error only refers to non-syntax-governing G functions. If more than one G function from one of these groups is called in the same NC block, the last command of the group is valid (the previous ones are ignored).</p> <p>G functions:</p> <table style="width: 100%; border: none;"> <tr> <td style="width: 60%;">Syntax-governing G functions:</td> <td style="text-align: right;">1st to 4th G group</td> </tr> <tr> <td>Non-syntax-governing G functions:</td> <td style="text-align: right;">5th to nth G group</td> </tr> </table>		Syntax-governing G functions:	1st to 4th G group	Non-syntax-governing G functions:	5th to nth G group
Syntax-governing G functions:	1st to 4th G group					
Non-syntax-governing G functions:	5th to nth G group					
Effect	NC block preparation stop					
Elimination	Correct the NC program					
Acknowl.	Clear the error with “Reset”.					
12 070	Channel %1 block %2 Too many syntax-defining G functions					
Cause	<p>%1 = channel number; %2 = block number, label</p> <p>Syntax-defining G functions determine the structure of the NC program block and the addresses it contains. Only one syntax-governing G function can be programmed in a block. Syntax-defining functions are G functions of the 1st to 4th G group.</p>					
Effect	NC block preparation stop					
Elimination	Correct the NC program					
Acknowl.	Clear the error with “Reset”.					
12 080	Channel %1 block %2 Syntax error in text %3					
Cause	<p>%1 = channel number; %2 = block number, label; %3 = source text area</p> <p>The grammar rules of the block have been violated at the displayed point in the text. There are too many potential causes to specify the error more accurately.</p> <p>Example</p> <p>N10 IF GOTOF ... ; The condition for the jump is missing!</p>					
Effect	NC block preparation stop					
Elimination	Correct the NC program					
Acknowl.	Clear the error with “Reset”.					

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
12 110	Channel %1 block %2 Syntax cannot be interpreted	
	Cause	%1 = channel number; %2 = block number, label The addresses programmed in the block are not permitted with the valid syntax-governing G function. e.g. G1 I10 X20 Y30 F1000 No interpolation parameters can be programmed in the linear block.
	Effect	NC block preparation stop
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
12 120	Channel %1 block %2 G function not programmed alone	
	Cause	%1 = channel number; %2 = block number, label The G function programmed in this block must be programmed on its own. No general addresses or synchronized actions can occur in the same block. These functions are: G25, G26 Working area limitation G110, G111, G112 Pole programming with polar coordinates e.g. G4 F1000 M100 An M function is not allowed in the G4 block.
	Effect	NC block preparation stop
	Elimination	Program the G function on its own in the block.
	Acknowl.	Clear the error with "Reset".
12 400	Channel %1 block %2 array %3 Index does not exist	
	Cause	%1 = channel number; %2 = block number, label; %3 = field index A system variable was programmed without an index
	Effect	NC block preparation stop
	Elimination	Program an index for the system variable
	Acknowl.	Clear the error with "Reset".
12 550	Channel %1 block %2 identifier %3 not defined or option does not exist	
	Cause	%1 = channel number; %2 = block number, label; %3 = source symbol The displayed identifier is not defined.
	Effect	NC block preparation stop
	Elimination	Correct the names used (notation error)
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
12 570	Channel %1 block %2 Too many motion synchronous actions in %3	
	Cause	%1 = channel number; %2 = block number, label; %3 = source symbol A maximum of 16 actions may be programmed in a synchronous motion block.
	Effect	NC block preparation stop
	Elimination	Reduce the number of programmed actions.
	Acknowl.	Clear the error with "Reset".
12 571	Channel %1 block %2 %3 Not allowed in motion synchronous action	
	Cause	%1 = channel number; %2 = block number, label; %3 = source symbol It is not permissible to program the specified predefined subroutine %3 in a block with a motion synchronous action. It may only be programmed on its own in a "normal" block.
	Effect	NC block preparation stop
	Elimination	Change the NC program
	Acknowl.	Clear the error with "Reset".
12 572	Channel %1 block %2 %3 Only allowed in motion synchronous action	
	Cause	%1 = channel number; %2 = block number, label; %3 = source symbol The specified predefined subroutine %3 may only be programmed in blocks with motion synchronous actions. It must not be programmed on its own in a "normal" block.
	Effect	NC block preparation stop
	Elimination	Change the NC program
	Acknowl.	Clear the error with "Reset".
12 580	Channel %1 block %2 Invalid assignment to %3 for motion synchronous action	
	Cause	%1 = channel number; %2 = block number, label; %3 = source symbol The displayed variable must not be written in a motion synchronous action. Only selected variables may be written in these actions. e.g.: DO \$AA_IW[X]=10 is not allowed
	Effect	NC block preparation stop
	Elimination	Change the NC program Only certain variables may be used in a motion synchronous action. E.g.: \$AA_IM
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
12 581	Channel %1 block %2 illegal read access to %3 in motion synchronous action	
Cause	%1 = channel number; %2 = block number, label; %3 = source symbol The displayed variable must not be used in a motion synchronous action. Example: The displayed variable may not be programmed to the left of the comparison in a motion synchronous action. Only selected variables may be used for this purpose, e.g.: WHEN \$AA_OVR == 100 DO ...	
Effect	NC block preparation stop	
Elimination	Change the NC program	
Acknowl.	Clear the error with "Reset".	
12 582	Channel %1 block %2 array index %3 invalid	
Cause	%1 = channel number; %2 = block number, label; %3 = source symbol \$A or \$V variables are evaluated in real time, i.e. in the interpolation cycle, in motion synchronous actions. All other variables (e.g. user-defined variables) are calculated as before during block preparation. Variables to be calculated during block preparation must not be indexed with real-time variables. Example: WHEN \$A_IN[1] == R[\$A_INA[1]] DO ... The R parameter must not be indexed with a real-time variable. Program editing: WHEN \$A_IN[1] == \$AC_MARKER[\$A_INA[1]] DO ...	
Effect	NC block preparation stop	
Elimination	Change the NC program: Use real-time variables	
Acknowl.	Clear the error with "Reset".	
12 583	Channel %1 block %2 variable %3 no system variable	
Cause	%1 = channel number; %2 = block number, label; %3 = source symbol In motion synchronous actions, only special system variables are allowed on the left side of the compare operation for the assigned variable. These can be accessed in real-time synchronism. The programmed variable is not a system variable.	
Effect	NC block preparation stop	
Elimination	Change the NC program. Local variables or machine data may not be used as parameters for SYNFACT.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
12 587	Channel %1 block %2 Motion synchronous action: Operation / function %3 not valid	
Cause	%1 = channel number; %2 = block number, label; %3 = operator/function The specified function / operator may not legally be used to link real-time variables in motion synchronous actions. The following operators / functions are legal: == >= <= > < <> + - * / AND OR XOR NOT B_AND B_OR B_XOR B_NOT SIN COS TAN SQRT POT TRUNC ABS	
Effect	NC block preparation stop	
Elimination	Change the NC program	
Acknowl.	Clear the error with "Reset".	
12 588	Channel %1 block %2 Motion synchronous action: Address %3 illegal	
Cause	%1 = channel number; %2 = block number, label; %3 = address <ul style="list-style-type: none"> The specified address may not be programmed in motion synchronous actions. Example: ID = 1 WHENEVER \$A_IN[1]==1 DO T2 <ul style="list-style-type: none"> The tool offset cannot be modified from motion synchronous actions. 	
Effect	NC block preparation stop	
Elimination	Change the NC program	
Acknowl.	Clear the error with "Reset".	
12 589	Channel %1 block %2 Motion synchronous action: Variable %3 not allowed with modal ID	
Cause	%1 = channel number; %2 = block number, label; %3 = variable name The ID in motion synchronous actions may not be formed by a system variable. Examples: ID=\$AC_MARKER[1] WHEN \$a_in[1] == 1 DO \$AC_MARKER[1] = \$AC_MARKER[1]+1 This can be corrected in the following way: R10 = \$AC_MARKER[1] ID=R10 WHEN \$a_in[1] == 1 DO \$AC_MARKER[1] = \$AC_MARKER[1]+1 The ID of a synchronous action is always fixed, it cannot be changed in the interpolation cycle.	
Effect	NC block preparation stop	
Elimination	Change the NC program, replace the system variable by an arithmetic variable.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy								
Channel error									
12 660	<p>Channel %1 block %2 Motion synchronous action: Variable %3 reserved for motion synchronous actions and subroutines as action</p> <table border="1"> <tr> <td data-bbox="344 472 480 640">Cause</td> <td data-bbox="480 472 1367 640"> %1 = channel number; %2 = block number, label; %3 = variable name The displayed variable may only be used as an action in motion synchronous actions or subroutines. For example, "\$R1" may only be programmed in motion synchronous actions. R parameters are programmed with R1 in a normal NC program. </td> </tr> <tr> <td data-bbox="344 640 480 685">Effect</td> <td data-bbox="480 640 1367 685">NC block preparation stop</td> </tr> <tr> <td data-bbox="344 685 480 730">Elimination</td> <td data-bbox="480 685 1367 730">Change the NC program</td> </tr> <tr> <td data-bbox="344 730 480 770">Acknowl.</td> <td data-bbox="480 730 1367 770">Clear the error with "Reset".</td> </tr> </table>	Cause	%1 = channel number; %2 = block number, label; %3 = variable name The displayed variable may only be used as an action in motion synchronous actions or subroutines. For example, "\$R1" may only be programmed in motion synchronous actions. R parameters are programmed with R1 in a normal NC program.	Effect	NC block preparation stop	Elimination	Change the NC program	Acknowl.	Clear the error with "Reset".
Cause	%1 = channel number; %2 = block number, label; %3 = variable name The displayed variable may only be used as an action in motion synchronous actions or subroutines. For example, "\$R1" may only be programmed in motion synchronous actions. R parameters are programmed with R1 in a normal NC program.								
Effect	NC block preparation stop								
Elimination	Change the NC program								
Acknowl.	Clear the error with "Reset".								
12 661	<p>Channel %1 block %2 Subprograms as action in motion synchronous action %3: No further subroutine call possible</p> <table border="1"> <tr> <td data-bbox="344 837 480 949">Cause</td> <td data-bbox="480 837 1367 949"> %1 = channel number; %2 = block number, label; %3 = name of subprogram as action It is not possible to call another subroutine as an action in a subroutine. </td> </tr> <tr> <td data-bbox="344 949 480 994">Effect</td> <td data-bbox="480 949 1367 994">NC block preparation stop</td> </tr> <tr> <td data-bbox="344 994 480 1039">Elimination</td> <td data-bbox="480 994 1367 1039">Change the NC program</td> </tr> <tr> <td data-bbox="344 1039 480 1084">Acknowl.</td> <td data-bbox="480 1039 1367 1084">Clear the error with "Reset".</td> </tr> </table>	Cause	%1 = channel number; %2 = block number, label; %3 = name of subprogram as action It is not possible to call another subroutine as an action in a subroutine.	Effect	NC block preparation stop	Elimination	Change the NC program	Acknowl.	Clear the error with "Reset".
Cause	%1 = channel number; %2 = block number, label; %3 = name of subprogram as action It is not possible to call another subroutine as an action in a subroutine.								
Effect	NC block preparation stop								
Elimination	Change the NC program								
Acknowl.	Clear the error with "Reset".								
14 000	<p>Channel %1 block %2 Illegal end of file</p> <table border="1"> <tr> <td data-bbox="344 1128 480 1263">Cause</td> <td data-bbox="480 1128 1367 1263"> %1 = channel number; %2 = block number, label An M02, an M17 or an M30 is expected at the end of the file. The block preparation (data management) does not return a following block although no end of file was programmed at the end of the previous block. </td> </tr> <tr> <td data-bbox="344 1263 480 1352">Effect</td> <td data-bbox="480 1263 1367 1352"> <ul style="list-style-type: none"> • Start disable • NC block preparation stop </td> </tr> <tr> <td data-bbox="344 1352 480 1397">Elimination</td> <td data-bbox="480 1352 1367 1397">Correct the NC program</td> </tr> <tr> <td data-bbox="344 1397 480 1431">Acknowl.</td> <td data-bbox="480 1397 1367 1431">Clear the error with "Reset".</td> </tr> </table>	Cause	%1 = channel number; %2 = block number, label An M02 , an M17 or an M30 is expected at the end of the file. The block preparation (data management) does not return a following block although no end of file was programmed at the end of the previous block.	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	Elimination	Correct the NC program	Acknowl.	Clear the error with "Reset".
Cause	%1 = channel number; %2 = block number, label An M02 , an M17 or an M30 is expected at the end of the file. The block preparation (data management) does not return a following block although no end of file was programmed at the end of the previous block.								
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 								
Elimination	Correct the NC program								
Acknowl.	Clear the error with "Reset".								
14 011	<p>Channel %1 block %2 NC program does not exist or not released for machining</p> <table border="1"> <tr> <td data-bbox="344 1476 480 1588">Cause</td> <td data-bbox="480 1476 1367 1588"> %1 = channel number; %2 = block number, label The NC program which was called is not available in the NC memory or was not enabled. </td> </tr> <tr> <td data-bbox="344 1588 480 1677">Effect</td> <td data-bbox="480 1588 1367 1677"> <ul style="list-style-type: none"> • Start disable • NC block preparation stop </td> </tr> <tr> <td data-bbox="344 1677 480 1744">Elimination</td> <td data-bbox="480 1677 1367 1744"> <ul style="list-style-type: none"> • Check whether NC program is stored in NC memory. • Check program enable </td> </tr> <tr> <td data-bbox="344 1744 480 1789">Acknowl.</td> <td data-bbox="480 1744 1367 1789">Clear the error with "Reset".</td> </tr> </table>	Cause	%1 = channel number; %2 = block number, label The NC program which was called is not available in the NC memory or was not enabled.	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	Elimination	<ul style="list-style-type: none"> • Check whether NC program is stored in NC memory. • Check program enable 	Acknowl.	Clear the error with "Reset".
Cause	%1 = channel number; %2 = block number, label The NC program which was called is not available in the NC memory or was not enabled.								
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 								
Elimination	<ul style="list-style-type: none"> • Check whether NC program is stored in NC memory. • Check program enable 								
Acknowl.	Clear the error with "Reset".								

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
14 014	Channel %1 Selected program %3 or access permission not available	
Cause	%1 = channel number; %3 = program name The selected NC program is not in the NC memory or has a higher protection level than currently active.	
Effect	Warning	
Elimination	Enter or read in NC program	
Acknowl.	Clear error with the CANCEL key.	
14 033	Channel %1 block %2 involute: No end point programmed	
Cause	%1 = channel number; %2 = block number, label No end point was programmed for the involute.	
Effect	NC block preparation stop	
Elimination	Check the NC program and modify it if necessary.	
Acknowl.	Use "Start" to clear the error.	
14 034	Channel %1 block %2 involute: Angle of rotation too large	
Cause	%1 = channel number; %2 = block number, label The base circle of the involute is reached using the programmed angle of rotation (AR).	
Effect	NC block preparation stop	
Elimination	Check the NC program and modify it if necessary.	
Acknowl.	Use "Start" to clear the error.	
14 035	Channel %1 block %2 involute: Starting point invalid	
Cause	%1 = channel number; %2 = block number, label The starting point of the involute must be located outside the base circle. The programmed center point or radius must be adapted accordingly.	
Effect	NC block preparation stop	
Elimination	Check the NC program and modify it if necessary.	
Acknowl.	Use "Start" to clear the error.	
14 036	Channel %1 block %2 involute: End point invalid	
Cause	%1 = channel number; %2 = block number, label The end point of the involute must be located outside the base circle. The programmed center point or radius must be adapted accordingly.	
Effect	NC block preparation stop	
Elimination	Check the NC program and modify it if necessary.	
Acknowl.	Use "Start" to clear the error.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
14 037	Channel %1 block %2 involute: Invalid radius	
	Cause	%1 = channel number; %2 = block number, label The programmed radius of the base circle must be greater than zero.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it if necessary.
	Acknowl.	Use "Start" to clear the error.
14 038	Channel %1 block %2 Involute cannot be determined: End point error	
	Cause	%1 = channel number; %2 = block number, label The programmed end point is not on the involute defined by starting point, radius and center point of the base circle. The effective end radius deviates from the programmed value more than 0.01 mm.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it if necessary.
	Acknowl.	Use "Start" to clear the error.
14 039	Channel %1 block %2 involute: End point programmed several times	
	Cause	%1 = channel number; %2 = block number, label Either the end point or the angle of rotation AR can be programmed. End point and angle of rotation must not be programmed in one and the same block.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it if necessary.
	Acknowl.	Use "Start" to clear the error.
14 040	Channel %1 block %2 Error in end point of circle	
	Cause	%1 = channel number; %2 = block number, label The circle starting point or circle center or circle end point are incorrectly programmed.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Check the circle geometry
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
14 080	Channel %1 block %2 Jump destination not found	
Cause	%1 = channel number; %2 = block number, label In jump commands, the jump destination must lie within the NC program in the specified direction.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Correct NC program (jump direction, jump destination)	
Acknowl.	Clear the error with "Reset".	
14 092	Channel %1 block %2 axis %3 has wrong axis type	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name Certain keywords require a defined type of axis (see Chapter 10). Example: WAITP, G74	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Correct the NC program	
Acknowl.	Clear the error with "Reset".	
14 095	Channel %1 block %2 Circle radius has been programmed too small	
Cause	%1 = channel number; %2 = block number, label Circle radius has been programmed too small or as zero.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Check the circle geometry	
Acknowl.	Clear the error with "Reset".	
14 750	Channel %1 block %2 Too many auxiliary functions programmed	
Cause	%1 = channel number; %2 = block number, label More than 5 M functions and/or 3 H functions were programmed in an NC block.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Reduce the number of auxiliary functions to the value permitted per NC block.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
14 751	Channel %1 block %2 Maximum number of motion synchronous actions exceeded (identifier %3)	
	Cause	%1 = channel number; %2 = block number, label; %3 = code The permissible number of active motion synchronous actions has been exceeded (max. 320 elements).
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Reduce the number of synchronous actions to the value permitted.
	Acknowl.	Clear the error with "Reset".
14 757	Channel %1 block %2 Motion synchronous action and wrong type	
	Cause	%1 = channel number; %2 = block number, label Programmed combination between action and type of motion synchronous action is illegal.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
14 760	Channel %1 block %2 Auxiliary function of a group programmed repeatedly	
	Cause	%1 = channel number; %2 = block number, label Only one auxiliary function is permitted within a group.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Program only one auxiliary function per auxiliary function group.
	Acknowl.	Clear the error with "Reset".
14 770	Channel %1 block %2 Auxiliary function programmed incorrectly	
	Cause	%1 = channel number; %2 = block number, label The programmed auxiliary function has an invalid value. e.g.: programmed value is negative
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
14 790	Channel %1 block %2 axis %3 currently controlled by CPU	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name An axis which is already being traversed by the CPU was programmed in the NC block.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	<ul style="list-style-type: none"> • Do not use this axis in the NC program while it is being traversed by the CPU. • Change the NC program (insert WAITP). 	
Acknowl.	Clear the error with "Reset".	
15 460	Channel %1 block %2 Syntax conflict with modal G function	
Cause	%1 = channel number; %2 = block number, label; The addresses programmed in the block are not compatible with the modal G function. e.g. G01 and I, J or K	
Effect	NC block preparation stop	
Elimination	Correct the NC program	
Acknowl.	Clear the error with "Reset".	
16 410	Channel %1 block %3 axis %2 is not a geometry axis	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name Geometry axes must be programmed in the block. e.g. G2 X... Y... X and Y must be geometry axes.	
Effect	NC block preparation stop	
Elimination	Correct the NC program	
Acknowl.	Clear the error with "Reset".	
16 420	Channel %1 block %2 axis %3 repeatedly programmed	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name It is illegal to program an axis more than once in a block.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Delete axis addresses programmed more than once.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 771	Channel %1 following axis %2 Overlaid motion not enabled	
	Cause	%1 = channel number; %2 = axis number No overlaid motion can be performed for the specified axis; the interface signal "Enable following axis override" (user DB "AXy", DBX8.4+m) is not set.
	Effect	Warning
	Elimination	Set interface signal.
	Acknowl.	Correct the error.
16 776	Channel %1 block %2 curve table %3 does not exist for axis %4	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of curve table; %4 = axis name An attempt has been made to link axis %4 to the curve table with number %3, although no curve table with this number exists.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Change the NC program such that the required curve table exists at the point in time at which the axis coupling is to be activated.
	Acknowl.	Clear the error with "Reset".
16 777	Channel %1 block %2 Master value coupling: Lead axis %4 slave axis %3 not available	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name; %4 = axis name A coupling has been activated in which the slave axis is not currently available. Possible cause: The axis has been traversed by the CPU and is not yet released.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Release leading axis from the CPU.
	Acknowl.	Clear the error with "Reset".
16 778	Channel %1 block %2 Master value coupling: Ring coupling for slave axis %3 and lead axis %4 not allowed	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name; %4 = axis name A coupling has been activated which, when other couplings are taken into account, produces a ring coupling. Ring couplings cannot be unambiguously calculated.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Correct coupling accordingly
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 779	Channel %1 block %2 Master value coupling: Too many couplings for axis %3, see active leading axis %4	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name; %4 = axis name Too many leading axes have been defined for the specified axis. A leading axis to which the specified axis is already coupled has been named in the last parameter.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Correct the NC program
	Acknowl.	Clear the error with "Reset".
16 780	Channel %1 block %2 Following axis missing	
	Cause	%1 = channel number; %2 = block number, label No following axis has been programmed for the electronic gear (EG).
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.
16 781	Channel %1 block %2 Master axis missing	
	Cause	%1 = channel number; %2 = block number, label No master axis has been programmed for the EG.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.
16 782	Channel %1 block %2 Following axis %3 not available	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name An electronic gear (EG) has been activated, but the following axis is not available. Possible causes are: <ul style="list-style-type: none"> • The axis is active in a different channel. • The axis is a CPU axis.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly; do not use the axis as a CPU axis.
	Acknowl.	Use "Start" to clear the error.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 783	Channel %1 block %2 Master axis %3 not available	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name An electronic gear (EG) has been activated, but the master axis is not available. Possible causes are: <ul style="list-style-type: none"> • The axis is active in a different channel. • The axis is a CPU axis.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly; do not use the axis as a CPU axis.
	Acknowl.	Use "Start" to clear the error.
16 785	Channel %1 block %2 Identical axes %3	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name An EG has been activated at which the following axis is identical to the master axis.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.
16 788	Channel %1 block %2 Ring coupling	
	Cause	%1 = channel number; %2 = block number, label An electronic gear has been activated with which a ring coupling results, taking into account further couplings. This cannot be calculated unambiguously.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.
16 789	Channel %1 block %2 Multiple coupling	
	Cause	%1 = channel number; %2 = block number, label A coupling has been activated with which the axes are already occupied by another coupling. Parallel couplings cannot be processed.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.
16 790	Channel %1 block %2 Parameter is zero or missing	
	Cause	%1 = channel number; %2 = block number, label An EG has been programmed with which a relevant parameter was specified with zero or was not written at all (e.g. denominator of transmission ratio).
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 791	Channel %1 block %2 Parameter is not relevant	
	Cause	%1 = channel number; %2 = block number, label A parameter not relevant for the EG was programmed.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.
16 792	Channel %1 block %2 Too many couplings for axis %3	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name For the specified axis, more master axes have been programmed than permitted.
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
Acknowl.	Use "Start" to clear the error.	
16 794	Channel %1 block %2 Coupling of axis/spindle %3 prohibits referencing	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name The specified axis is a (gantry) slave axis and cannot therefore approach the reference point independently.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	<ul style="list-style-type: none"> • Correct the NC program • Deactivate coupling(s) of this axis before reference point approach • or do not reference the axis A gantry slave (synchronized axis) cannot be referenced independently.
	Acknowl.	Clear the error with "Reset".
16 795	Channel %1 block %2 String cannot be interpreted	
	Cause	%1 = channel number; %2 = block number, label An EG containing a string which cannot be interpreted was programmed (e.g. block change behavior).
	Effect	NC block preparation stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Use "Start" to clear the error.
16 796	Channel %1 block %2 Coupling not defined	
	Cause	%1 = channel number; %2 = block number, label An EG whose parameter is not programmed is to be activated.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block processing stop
	Elimination	Check the NC program and modify it accordingly.
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 797	Channel %1 block %2 Coupling is active	
Cause	%1 = channel number; %2 = block number, label An operation is to be carried out during which no EG must be active.	
Effect	NC block preparation stop	
Elimination	Check the NC program and modify it accordingly.	
Acknowl.	Use "Start" to clear the error.	
16 800	Channel %1 block %2 Traversing statement DC/CDC not permitted for axis %3	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name The traversing statement DC cannot be used for linear axes. The traversing statement CDC can only be used for modulo rotary axes.	
Effect	NC block preparation stop	
Elimination	Check the NC program and modify it accordingly.	
Acknowl.	Use "Start" to clear the error.	
16 810	Channel %1 block %2 Traversing statement ACP not permitted for axis %3	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name The traversing statement ACP can only be used for modulo rotary axes.	
Effect	NC block preparation stop	
Elimination	Check the NC program and modify it accordingly.	
Acknowl.	Use "Start" to clear the error.	
16 820	Channel %1 block %2 Traversing statement ACN not permitted for axis %3	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name The traversing statement ACP can only be used for modulo rotary axes.	
Effect	<ul style="list-style-type: none"> • NC block preparation stop • Start disable 	
Elimination	Check the NC program and modify it accordingly.	
Acknowl.	Clear the error with "Reset".	
16 830	Channel %1 block %2 Invalid position for axis/spindle %3 programmed	
Cause	%1 = channel number; %2 = block number, label; %3 = axis name A position outside the range 0...359.999 was programmed for a modulo axis.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Program a position within the range 0 to 359.999.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 903	Channel %1 Action %2 not permitted in current state	
	Cause	%1 = channel number; %2 =action numer/name The action in question can currently not be processed. This error can occur, e.g. when reading in machine data.
	Effect	Warning
	Elimination	Wait until the other process is completed.
	Acknowl.	Use the CANCEL key to clear the error.
16 904	Channel %1 Action %2 not permitted in current state	
	Cause	%1 = channel number; %2 =action number/name The processing in question (program, jogging, reference point, ...) cannot be started or continued in the current state.
	Effect	Warning
	Elimination	Check program status anmd channel status.
Acknowl.	Use the CANCEL key to clear the error.	
16 906	Channel %1 Action %2 aborted due to an error	
	Cause	%1 = channel number; %2 =action number/name The action has been aborted due to an error.
	Effect	Warning
	Elimination	Correct and acknowledge the error.
Acknowl.	Use the CANCEL key to clear the error.	
16 907	Channel %1 action %2 only possible in Stop mode	
	Cause	%1 = channel number; %2 = action number/name The action can only be executed in the stopped state.
	Effect	Warning
	Elimination	Check program and channel state
Acknowl.	Clear error with the CANCEL key.	
16 908	Channel %1 Action %2 only possible in the Reset state or at block end	
	Cause	%1 = channel number; %2 =action number/name The action may only be carried out in the Reset state or at the end of the block.
	Effect	Warning
	Elimination	Check program status and channel status.
Acknowl.	Use the CANCEL key to clear the error.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 909	Channel %1 Action %2 not permitted in current mode	
	Cause	%1 = channel number; %2 =action number/name Another mode must be activated for the function to be activated.
	Effect	Warning
	Elimination	Check operation and mode.
	Acknowl.	Use the CANCEL key to clear the error.
16 912	Channel %1 Action %2 only possible in the Reset state	
	Cause	%1 = channel number; %2 =action number/name The action can only be carried out in the Reset state. Example: Program selection is only possible in the Reset state.
	Effect	Warning
	Elimination	Reset or wait until the end of the program is reached.
	Acknowl.	Use the CANCEL key to clear the error.
16 926	Channel %1 Channel coordination: Action %2 not permitted in block %3; marker %4 already set	
	Cause	%1 = channel number; %2 =Aktionsnummer/-name %3 = block number, label; %4 = marker number The action has already been denied; the marker to be set is already set.
	Effect	<ul style="list-style-type: none"> • NC block preparation stop • Start disable
	Elimination	Correct the NC program accordingly.
	Acknowl.	Use the CANCEL key to clear the error.
16 930	Channel %1: Previous block and current block %2 must be separated by an executable block	
	Cause	%1 = channel number; %2 = block number, label An executable block must be programmed between the statements WAITMC, SETM, CLEARM. An executable block contains, e.g. a traversing motion, an auxiliary function, dwell time, ...
	Effect	NC block preparation stop
	Elimination	Correct the NC program accordingly.
	Acknowl.	Use "Start" to clear the error.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
16 941	Channel %1 Action %2 denied, since program event not yet processed	
Cause	%1 = channel number; %2 = action number/name The function "Event-controlled program call" was activated. The appropriate program EVENT.SPF, however, could not be started; the action (usually 'program start') must therefore be denied. Reasons why it was not possible to start the program EVENT.SPF: 1. Program does not exist. 2. Program may only start in the referenced state. 3. FM READY missing, e.g. due to an error	
Effect	Warning	
Elimination	Eliminate the a/m causes	
Acknowl.	Use the CANCEL key to clear the error.	
16 949	Relation between the mark of channel %1 and channel %2 is invalid	
Cause	%1 = channel number; %2 = channel number This channel defines a WAIT label with other channels which, in turn, are not related to this WAIT label. The WAIT label of this channel does not have an explicit pendant in the other channel, i.e. the channels do not wait for each other.	
Effect	Warning	
Elimination	Eliminate the a/m causes.	
Acknowl.	Press CANCEL or START to cancel the error.	
17 100	Channel %1 block %2 Digital input no. %3 is not active	
Cause	%1 = channel number; %2 = block number, label; %3 = no. of input An attempt was made to read a digital input of the FM 357-2 using system variable \$A_IN [n] with an index [n] greater than the number of digital inputs configured.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Read an index [n] of system variable \$A_IN [n] that is between 0 and max. value of digital inputs.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
17 110	Channel %1 block %2 Digital output no. %3 is not active	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of output An attempt was made to read or set a digital output of the FM 357-2 using system variable \$A_OUT [n] with an index [n] greater than the number of digital outputs configured.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Program an index [n] of system variable \$A_OUT [n] that is between 0 and the max. value of digital outputs configured.
	Acknowl.	Clear the error with "Reset".
17 120	Channel %1 block %2 Analog input no. %3 not activated	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of input It was tried to read the system variable \$A_INA[n] of an analog input n. The analog input, however, is not yet activated.
	Effect	NC block preparation stop
	Elimination	<ul style="list-style-type: none"> • Check the NC program and modify it accordingly. • Activate the analog input in the parameterization.
	Acknowl.	Use "Start" to clear the error.
17 130	Channel %1 block %2 Analog output no. %3 not activated	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of output It was tried to read or write the system variable \$A_OUTA[n] of an analog output n. The analog output, however, is not activated.
	Effect	NC block preparation stop
	Elimination	<ul style="list-style-type: none"> • Check the NC program and modify it accordingly. • Activate the analog input in the parameterization.
	Acknowl.	Use "Start" to clear the error.
17 140	Channel %1 block %2 Output no. %3 is assigned to function via machine data	
	Cause	%1 = channel number; %2 = block number, label; %3 = no. of output The programmed digital output is already being used by an NC function (e.g. software cam).
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Use an unassigned output.
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
17 150	Channel %1 block %2 Maximum %3 FM outputs per block exceeded	
Cause	%1 = channel number; %2 = block number, label; %3 = number of outputs You cannot program more than the specified number of outputs in a block. The number of digital outputs is defined in the parameters:	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Program fewer digital outputs in the block.	
Acknowl.	Clear the error with "Reset".	
17 190	Channel %1 block %2 Illegal T number	
Cause	%1 = channel number; %2 = block number, label The displayed block accesses a T number (tool number) which has not been created.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Check the tool number in the NC program.	
Acknowl.	Clear the error with "Reset".	
18 000	Channel %1 block %2 Error in protection zone %3 error no. %4	
Cause	%1 = channel number; %2 = block number, label; %3 = protection zone number; %4 = error specification The definition of the protection zone contains an error. The error number localizes the cause of the error. Error numbers: 1: Incomplete or implausible contour definition 2: Contour surrounds more than one area 3: Tool-related protection zone is not convex. 4: If both limits are active in the 3rd dimension of the protection zone and both limits have the same value. 5: The protection zone number is invalid (negative number, zero or greater than the maximum number of protection zones). 6: Protection zone description includes more than 4 contour elements 7: Tool-related protection zone is defined as internal protection zone 8: Invalid parameter used 9: Protection zone to be activated is not defined 10: Invalid modal G code used for protection zone definition 11: Invalid contour definition or frame activated 12: Unspecified error	
Effect	Start disable	
Elimination	Change definition of protection zone.	
Acknowl.	Clear error with the CANCEL key.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
18 002	Channel %1 block %2 protection zone %3 cannot be activated. Error no. %4	
Cause	<p>%1 = channel number; %2 = block number, label; %3 = protection zone number; %4 = error specification</p> <p>An error occurred on activation of the protection zone. The error number localizes the cause of the error.</p> <p>Error numbers:</p> <ul style="list-style-type: none"> 1: Incomplete or implausible contour definition 2: Contour surrounds more than one area 3: Tool-related protection zone is not convex. 4: If both limits are active in the 3rd dimension of the protection zone and both limits have the same value. 5: The protection zone number is invalid (negative number, zero or greater than the maximum number of protection zones). 6: Protection zone description includes more than 4 contour elements 7: Tool-related protection zone is defined as internal protection zone 8: Invalid parameter used 9: Protection zone to be activated is not defined 10: Error on internal setup of the protection zones 11: Unspecified error 12: The number of protection zones which can be active simultaneously was exceeded. 13, 14: Contour element cannot be created for protection zones 15, 16: No more memory for protection zones 17: No more memory for contour elements 	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	<ul style="list-style-type: none"> • Change definition or activation of protection zones. • Reduce number of active protection zones. 	
Acknowl.	Clear the error with "Reset".	
18 004	Channel %1 block %2 Unequal orientation between workpiece-related %3 and tool-related %4 protection zones	
Cause	<p>%1 = channel number; %2 = block number, label; %3 = number of the workpiece-related protection zone; %4 = number of the tool-related protection zone</p> <p>The orientation of the workpiece-related protection zone and the orientation of the tool-related protection zone are different.</p>	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Change the protection zone definition or do not activate protection zones with different orientation simultaneously.	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
18 005	Channel %1 block %2 Serious error in definition of protection zone %3	
Cause	%1 = channel number; %2 = block number, label; %3 = protection zone number You must end the protection zone definition with EXECUTE. This also applies to protection zones which are initiated implicitly e.g. with M30.	
Effect	Start disable	
Elimination	Change definition of protection zone.	
Acknowl.	Clear the error with "Reset".	
18 100	Channel %1 block %2 Invalid value passed to FXS[]	
Cause	%1 = channel number; %2 = block number, label At the present time, only the values: 0: "Deselect travel to fixed stop" 1: "Select travel to fixed stop" are valid.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Change NC program or parameter settings	
Acknowl.	Clear the error with "Reset".	
18 101	Channel %1 block %2 Invalid value passed to FXST[]	
Cause	%1 = channel number; %2 = block number, label At the present time, the only valid range is 0.0 to 100.0.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Change NC program or parameter settings	
Acknowl.	Clear the error with "Reset".	
18 102	Channel %1 block %2 Invalid value passed to FXSW[]	
Cause	%1 = channel number; %2 = block number, label At the present time, only positive values including zero are valid.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Change NC program or parameter settings	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Channel error		
18 200	Channel %1 block %2 Curve table: Block search stop not allowed with definition CTABDEF	
Cause	%1 = channel number; %2 = block number, label Program instructions which cause a block search stop may not be included in the definition of a curve table. System variable \$P_CTABDEF can be interrogated to establish whether table definition is currently active.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Factor the block using "IF NOT(\$P_CTABDEF) ... ENDIF" or remove the instruction which causes a block search stop. Start the NC program again afterwards.	
Acknowl.	Clear the error with "Reset".	
18 201	Channel %1 block %2 Curve table: Table %3 does not exist	
Cause	%1 = channel number; %2 = block number, label; %3 = No. of curve table An attempt has been made to use a curve table whose table number is not recognized in the system.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Change the table number in the program instruction or define the curve table with the desired table number.	
Acknowl.	Clear the error with "Reset".	
18 202	Channel %1 block %2 Curve table: Instruction CTABEND illegal without CTABDEF	
Cause	%1 = channel number; %2 = block number, label In the program, instruction CTABEND (which is used to terminate curve table definitions) has been used without instruction CTABDEF (which is used to start curve table definitions) being programmed beforehand or the instructions CTABDEF and CTABEND have not been programmed at the same program level.	
Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop 	
Elimination	Remove instruction CTABEND from the program or insert instruction CTABDEF at the appropriate place in the program. Start the program again. The instructions STABDEF and STABEND must be programmed at the same program level (main program or subprogram).	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 000	Channel %1 axis %2 Reference cam not reached	
Cause	%1 = channel number; %2 = axis name After you start reference point approach, the rising edge of the reference point switch (RPS) must be reached within the distance defined in the MD “max. distance to RPS”.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	<ul style="list-style-type: none"> • The value in MD “max. distance to RPS” is too small. • Check the RPS signal up to the CPU interface. 	
Acknowl.	Clear the error with “Reset”.	
20 001	Channel %1 axis %2 cam signal missing	
Cause	%1 = channel number; %2 = axis name The deceleration path after the RPS signal of the axis is greater than the length of the RPS.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Check whether the deceleration path from the referencing velocity is greater than the reference point switch. If this is the case, the axis cannot stop until it has passed the RPS. Use a longer RPS or reduce the MD “referencing velocity”.	
Acknowl.	Clear the error with “Reset”.	
20 002	Channel %1 axis %2 zero reference mark not found	
Cause	%1 = channel number; %2 = axis name The zero marker of the incremental encoder is not within the distance defined in MD “max. distance to zero marker/BERO” . The monitoring system prevents a zero marker signal from being crossed and the next being used as the reference point signal! (deficient RPS alignment or excessive delay by user program).	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Check the RPS alignment and ensure that there is sufficient clearance between the end of the RPS and the subsequent zero marker signal. The distance must be greater than the distance the axis can cover within the CPU cycle time. Increase MD “max. distance to zero marker/BERO”, but do not select a value larger than the distance between 2 zero markers (monitoring system ineffective).	
Acknowl.	Clear the error with “Reset”.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 005	Channel %1 axis %2 Reference point approach aborted	
Cause	%1 = channel number; %2 = axis name Referencing could not be completed for the specified axis (e.g. cancellation of the traversing command).	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Check the cause of cancellation:	
Acknowl.	Clear the error with "Reset".	
20 006	Channel %1 axis %2 Reference point creep velocity not reached	
Cause	%1 = channel number; %2 = axis name During reference point approach (approach zero marker), the end of the RPS was reached, but the creep velocity was not inside the tolerance window. This can happen if the axis is already positioned at the end of the RPS at the start of the reference point approach.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Reduce the MD "reducing velocity".	
Acknowl.	Clear the error with "Reset".	
20 070	Channel %1 axis %2 Programmed end position is beyond software limit switch %3	
Cause	%1 = channel number; %2 = axis number; %3 = "+" or "-" The axis is traversed by the CPU as a positioning axis and the target position is located behind the corresponding software limit switch. The axis is not moved.	
Effect	Axis does not move	
Elimination	Specify a target position within the permissible traversing range.	
Acknowl.	Error remedy	
20 071	Channel %1 axis %2 Programmed end position is beyond working area limit %3	
Cause	%1 = channel number; %2 = axis number; %3 = "+" or "-" The displayed axis is operated by the CPU as a positioning axis. Its target position is located behind the defined working area limitation.	
Effect	Axis does not move	
Elimination	Specify a target position within the permissible working area.	
Acknowl.	Error remedy	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 073	Channel %1 axis %2 Repositioning is not possible	
	Cause	%1 = channel number; %2 = axis number The position axis – operated by the CPU – cannot be repositioned because it has already been started via the FM and is still active. No repositioning movement takes place, the motion initiated by the FM remains unaffected.
	Effect	Warning
	Elimination	None
	Acknowl.	Clear error with the CANCEL key.
20 075	Channel %1 axis %2 Oscillation currently not possible	
	Cause	%1 = channel number; %2 = axis number The axis cannot execute an oscillation motion because it is already traversing, e.g. under CPU control.
	Effect	Warning
	Elimination	End the other traversing motion
	Acknowl.	Clear error with the CANCEL key.
20 076	Channel %1 axis %2 Change of operation mode not possible during oscillation	
	Cause	%1 = channel number; %2 = axis number The axis is performing an oscillation motion. The operating mode cannot be changed because oscillation is not permitted in the selected operating mode.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Do not change operating mode or end the oscillation motion.
	Acknowl.	Clear the error with “Reset”.
20 077	Channel %1 axis %2 Programmed position is beyond software limit switch %3	
	Cause	%1 = channel number; %2 = axis number; %3 = “+” or “-” The axis is traversing as an oscillation axis and the target position (reversal point or end position) lies beyond the corresponding software switch. The axis is not moved.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	<ul style="list-style-type: none"> • Enter a target position at a shorter distance. • Change the parameters for the software limit switch • Activate another software switch if appropriate
	Acknowl.	Clear the error with “Reset”.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 078	Channel %1 axis %2 Programmed position is beyond working area limit %3	
	Cause	%1 = channel number; %2 = axis number; %3 = “+” or “-” The axis is traversing as an oscillation axis and the target position (reversal point or end position) lies beyond the corresponding working area limit. The axis is not moved.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	<ul style="list-style-type: none"> • Enter a target position at a shorter distance. • Deactivate the working area limitation • Change the working area limitation settings
	Acknowl.	Clear the error with “Reset”.
20 079	Channel %1 axis %2 Oscillating path length %3 ≤ 0	
	Cause	%1 = channel number; %2 = axis number; %3 = length The axis is traversing as an oscillating axis and the distance to be traversed is less than or equal to zero, both reversal points are in an identical position. One reversal point has been shifted beyond the other reversal point in the opposite direction to oscillation. The axis is not moved.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Enter the correct target position (reversal point, end position)
	Acknowl.	Clear the error with “Reset”.
20 090	Axis %1 Activation of fixed stop not possible	
	Cause	%1 = axis name <ul style="list-style-type: none"> • The “Travel to fixed stop” function has been programmed with FXS[axis]=1, but this is (not yet) supported by the axis. This function is not available for gantry and simulated axes. • No motion has been programmed for the axis on selection. • A traversing motion must always be programmed in the selection block of the axis for which “Travel to fixed stop” is activated.
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop
	Elimination	<ul style="list-style-type: none"> • Check axis type • Does the approach block contain a programmed motion of the machine axis?
	Acknowl.	Clear the error with “Reset”.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 091	Axis %1 has not reached fixed stop	
Cause	%1 = axis name On the attempt to reach a fixed stop, the programmed end position has been reached or the traversing motion aborted. The error can be concealed via parameter "Error message: Axis has not reached fixed stop".	
Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop 	
Elimination	Correction of NC program and settings: <ul style="list-style-type: none"> • Has the traversing block been aborted? • If the axis is positioned on the programmed end position, then the end position must be corrected. • If the programmed end position is in the part, then the trigger criterion must be checked. • Has the contour deviation that acted as the trigger criterion been overdimensioned? Is the torque limit too high? 	
Acknowl.	Clear the error with "Reset".	
20 092	Axis %1 Fixed stop mode still active	
Cause	%1 = axis name An attempt has been made to traverse the axis positioned at the fixed stop or when the function has not been fully deselected.	
Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop 	
Elimination	Check the following points: <ul style="list-style-type: none"> • Is the movement of geometry axes causing the axis at the fixed stop to move as well? • Is the function being selected even though the axis is at the fixed stop? • Has the selection been aborted with Reset? • Has the CPU switched acknowledgement signals? 	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 093	Axis %1 Standstill monitoring at fixed stop end point has triggered	
	Cause	%1 = axis name The position of the axis since completion of selection is outside the monitoring window.
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop
	Elimination	<ul style="list-style-type: none"> • Check mechanical components, e.g. end stop broken off? • Monitoring window too small
	Acknowl.	Clear the error with "Reset".
20 094	Axis %1 Fixed stop mode has been aborted	
	Cause	%1 = axis name The function has been aborted. Possible causes are: <ul style="list-style-type: none"> • An impulse disable has made it impossible to maintain the necessary torque. • The CPU has cancelled the acknowledgement signals.
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop
	Elimination	Has the CPU cancelled the acknowledgement bits even though the FM has not requested deselection?
	Acknowl.	Clear the error with "Reset".
20 140	Channel %1 Motion synchronous action: Positioning axis cannot be traversed from synchronous action %2, error cause %3	
	Cause	%1 = channel number; %2 = axis name; %3 = error cause (error no.) An error has been detected in the positioning axis to be traversed from the synchronous action.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Change the NC program
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 141	Channel %1 Motion synchronous action: Illegal axis type	
	Cause	%1 = channel number The requested instruction is not legal for the positioning axis in its current status. The error occurs during positioning (POS, MOV), coupled motion (TRAILON, TRAILOF) and master value coupling (LEADON, LEADOF).
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Stop the axis first and deactivate the coupling. Then select the new state.
	Acknowl.	Clear the error with "Reset".
20 143	Channel %1 Axis %2 cannot be started, CPU axis	
	Cause	%1 = channel number; %2 = axis number The axis is currently a CPU axis and can thus not traverse within a synchronized action.
	Effect	Warning
	Elimination	Enable the axis from the CPU.
	Acknowl.	Use the CANCEL key to clear the error.
20 144	Channel %1 block %2 Motion synchronous action: Access to system variable not possible.	
	Cause	%1 = channel number; %2 = block number Access to system variable not possible. Example: Reading of an actual value of a non-existing axis.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Change the NC program or connect the hardware required.
	Acknowl.	Clear the error with "Reset".
20 145	Channel %1 block %2 Motion synchronous action: Arithmetic error	
	Cause	%1 = channel number; %2 = block number An overflow has occurred (e.g. division by zero) on calculation of an arithmetic expression for a motion synchronous action.
	Effect	<ul style="list-style-type: none"> • Start disable • Stop
	Elimination	Correct the relevant expression
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 146	Channel %1 block %2 Motion synchronous action: Nesting depth exceeded	
Cause	%1 = channel number; %2 = block number To calculate arithmetic expressions in motion synchronous actions, an operand stack of a fixed size must be used. This stack may overflow with very complex expressions.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Correct the relevant expression	
Acknowl.	Clear the error with "Reset".	
20 147	Channel %1 block %2 Motion synchronous action: Command not executable	
Cause	%1 = channel number; %2 = block number A command in the synchronous action block cannot be executed, e.g. it is not possible to perform a Reset for the action in this block.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Change the synchronous action	
Acknowl.	Clear the error with "Reset".	
20 148	Channel %1 block %2 Motion synchronous action: Internal error %3	
Cause	%1 = channel number; %2 = block number; %3 = error number An internal error has occurred while a synchronous action was being processed. If this occurs, please note the error no. and contact the SIEMENS AG hotline see Preface.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Change the synchronous action	
Acknowl.	Clear the error with "Reset".	
20 149	Channel %1 block %2 Motion synchronous action: Illegal index	
Cause	%1 = channel number; %2 = block number An illegal index has been used to access a variable in the motion synchronous action. Example: ... DO \$R[\$AC_MARKER[1]] = 100 The error occurs if marker 1 is set to a higher value than the maximum permissible R parameter number.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	Use a valid index	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
20 170	Channel %1 FIFO has exceeded the R parameter range	
	Cause	%1 = channel number The end of the FIFO range exceeds the end of the R parameter range.
	Effect	Start inhibit
	Elimination	<ul style="list-style-type: none"> • Relocate the start of the FIFO range in the R parameter range forward • Increase the number of R parameters • Reduce the number of FIFO elements
	Acknowl.	Turn off / turn on the FM.
21 610	Channel %1 axis %2 Encoder frequency limit exceeded	
	Cause	%1 = channel number; %2 = axis name; The maximum permissible frequency of the encoder defined in MD "encoder frequency limit" was exceeded.
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop
	Elimination	<ul style="list-style-type: none"> • Check MD "encoder frequency limit" . • Check axis velocity or encoder matching .
	Acknowl.	Clear the error with "Reset".
21 612	Channel %1 axis %2 Servo enable signal reset during traverse motion	
	Cause	%1 = channel number; %2 = axis name The "servo enable" interface signal was reset for the axis although the axis was in motion.
	Effect	<ul style="list-style-type: none"> • Stop • The FM switches to follow-up mode
	Elimination	Check the user program and/or the complete system.
	Acknowl.	Clear error with "Start" or CANCEL key.
21 614	Channel %1 axis %2 Hardware limit switch %3 reached	
	Cause	%1 = channel number; %2 = axis name; %3 = string Axis has reached hardware limit switch.
	Effect	Start disable
	Elimination	<ul style="list-style-type: none"> • Move the axis within the permissible traversing range • Check the plant geometry and the software limit positions.
	Acknowl.	Clear the error with "Reset".

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
21 700	Channel %1 block %3 axis %2 Touch probe already deflected, edge polarity not possible	
Cause	%1 = channel number; %2 = axis name; %3 = block number The selected probe is deflected and is therefore incapable of measuring a variable from the non-deflected to the deflected state.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	<ul style="list-style-type: none"> • Check the probe • Check the start position for measurement • Check the program 	
Acknowl.	Clear the error with "Reset".	
21 702	Channel %1 block %3 axis %2 Measurement aborted	
Cause	%1 = channel number; %2 = axis name; %3 = block number The measuring block has ended (the programmed position of the axis was reached), but a response has not yet been registered from the activated probe.	
Effect	Warning	
Elimination	<ul style="list-style-type: none"> • Check the probe • Check the program 	
Acknowl.	Clear error with the CANCEL key.	
21 703	Channel %1 block %3 axis %2 Touch probe not deflected, edge polarity not possible	
Cause	%1 = channel number; %2 = axis name; %3 = block number The selected probe is not deflected and is therefore incapable of measuring a variable from the deflected to the non-deflected state.	
Effect	<ul style="list-style-type: none"> • Start disable • Stop 	
Elimination	<ul style="list-style-type: none"> • Check the probe • Check the start position for measurement • Check the program 	
Acknowl.	Clear the error with "Reset".	
21 740	Output value for analog output no. %1 limited	
Cause	%1 = no of output The range of values of the analog output n is defined in the parameter "Evaluation". A value outside the range of values was specified via the system variable \$A_OUTA[n].	
Effect	Warning	
Elimination	<ul style="list-style-type: none"> • Specify a permissible value. • Change the range of values; if necessary replace the analog output module 	
Acknowl.	Use the CANCEL key to clear the error.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
25 000	Axis %1 Hardware fault of active encoder	
	Cause	%1 = axis name The encoder signals are missing or invalid.
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode
	Elimination	Check the encoder
	Acknowl.	FM restart
25 020	Axis %1 Zero mark monitoring of active encoder	
	Cause	%1 = axis name The encoder signals are invalid. The fluctuation in the number of pulses between zero markers exceeds the permissible tolerance (MD “number for zero marker monitoring”).
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode • The axes are no longer synchronized with the actual machine value (reference point).
	Elimination	The deviations can be caused by transfer errors, interference, encoder errors or faults in the encoder power supply.
	Acknowl.	Clear the error with “Reset”.
25 030	Axis %1 Actual velocity alarm	
	Cause	%1 = axis name The actual velocity of the axis has exceeded the threshold of the velocity monitoring system (MD “actual velocity”). This can be caused by: <ul style="list-style-type: none"> • Invalid parameterization of the axis velocity • Incorrect position control direction • Error on the drive
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode
	Elimination	Check the parameters or drive
	Acknowl.	Clear the error with “Reset”.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
25 040	Axis %1 Standstill monitoring	
Cause	%1 = axis name The position of the axis is monitored continuously when it is at a standstill (tolerance threshold in MD “zero speed range”). Monitoring begins after a time defined in MD “delay time” .	
Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode 	
Elimination	<ul style="list-style-type: none"> • Check the values in MD “delay time” and MD “zero speed range”. • Improve the optimization of the position control loop • Reduce the load torque • Optimize the drive 	
Acknowl.	Clear the error with “Reset”.	
25 050	Axis %1 Contour tolerance monitoring	
Cause	%1 = axis name The tolerance between the value calculated internally and the actual value exceeds the threshold stored in MD “contour tolerance”.	
Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode 	
Elimination	<ul style="list-style-type: none"> • Check the values in MD “contour tolerance” • Improve the optimization of the position control loop • Optimize the drive • Check the mechanical system 	
Acknowl.	Clear the error with “Reset”.	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
25 060	Axis %1 Desired speed setpoint	
	Cause	%1 = axis name The setpoint has exceeded the limit in MD "set velocity" for longer than the time permitted in MD "timeout". Brief violations are tolerated, the output setpoint is limited to the setting in MD "Setpoint velocity".
	Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode
	Elimination	<ul style="list-style-type: none"> • Check the values in MD "set velocity" and MD "timeout". • Improve the optimization of the position control loop • Optimize the drive • Check the mechanical system
	Acknowl.	Clear the error with "Reset".
25 070	Axis %1 Drift limit exceeded	
	Cause	%1 = axis name The maximum permissible value for the automatic drift (MD "drift limit") was exceeded. The static value (MD "Offset compensation") is not taken into account by the monitoring function.
	Effect	Warning
	Elimination	<ul style="list-style-type: none"> • Check the drive • Optimize the static drift
	Acknowl.	Clear error with the CANCEL key.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy									
Axis errors										
25 080	Axis %1 Positioning monitoring									
Cause	%1 = axis name During positioning, the “fine” target range (MD “Fine target range”) has not been reached within a specified time period (MD “Monitoring time”) . Target range coarse: MD “target range coarse” Target range fine: MD “target range fine”									
Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode 									
Elimination	<ul style="list-style-type: none"> • Check the parameter settings in MD “Fine target range”, MD “Monitoring time” and MD “Coarse target range”. • Improve the optimization of the position control loop • Optimize the drive • Check the mechanical system 									
Acknowl.	Clear the error with “Reset”.									
25 120	Axis %1 error identifier %3 active encoder %2									
Cause	%1 = axis name %3= encoder number %3 = error identifier Encoder-specific error identifier; is provided as an additional information to error 25 000. <table border="0" data-bbox="491 1205 1230 1328"> <thead> <tr> <th data-bbox="491 1205 660 1234">Error identifier</th> <th data-bbox="794 1205 911 1234">Comment</th> </tr> </thead> <tbody> <tr> <td data-bbox="491 1240 533 1270">100</td> <td data-bbox="794 1240 1075 1270">Implausible encoder value</td> </tr> <tr> <td data-bbox="491 1270 533 1299">101</td> <td data-bbox="794 1270 1193 1299">Failure of 1st encoder message fame</td> </tr> <tr> <td data-bbox="491 1299 533 1328">102</td> <td data-bbox="794 1299 1230 1328">Failure of cyclic encoder message frame</td> </tr> </tbody> </table>		Error identifier	Comment	100	Implausible encoder value	101	Failure of 1st encoder message fame	102	Failure of cyclic encoder message frame
Error identifier	Comment									
100	Implausible encoder value									
101	Failure of 1st encoder message fame									
102	Failure of cyclic encoder message frame									
Effect	<ul style="list-style-type: none"> • Start disable • Stop 									
Elimination	<ul style="list-style-type: none"> • Check plug connections • Check grounding and shielding conditions • Reduce the clock frequency 									
Acknowl.	Clear the error with ”Reset”.									

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
25 201	Axis %1 drive fault/error in rotation monitoring (stepper motor)	
Cause	%1 = axis name Group error for PROFIBUS-DP drives; evaluate further pending drive errors (if any). The rotation monitoring for the stepper motor has responded.	
Effect	<ul style="list-style-type: none"> • No ready message (no channel ready) • Start disable • Stop • The FM switches to follow-up mode. 	
Elimination	<ul style="list-style-type: none"> • Check the plug connections. • Check the drive. 	
Acknowl.	Clear the error with "Reset".	
25 202	Axis %1 Wait for drive	
Cause	%1 = axis name After PowerOn, it was not possible to set up the communication to the PROFIBUS-DP drive. After an internal timeout, the error is replaced with error 25 201.	
Effect	Warning	
Elimination	<ul style="list-style-type: none"> • Check the plug connections • Check the drive 	
Acknowl.	Correct the error.	
26 000	Axis %1 Clamping monitoring	
Cause	%1 = axis name The clamped axis has been pushed out of its set position. The permissible deviation is defined in MD "clamping tolerance".	
Effect	<ul style="list-style-type: none"> • No ready signal (no Channel ready) • Start disable • Stop • The FM switches to follow-up mode 	
Elimination	<ul style="list-style-type: none"> • Check the values in MD "clamping tolerance". • Improve the clamping • Reduce the load torque 	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
26 052	Channel %1 in block %2 Drop in velocity in continuous-path mode	
	Cause	%1 = channel number; %2 = block number, label Auxiliary function output too late or not acknowledged. The next block for path interpolation is missing.
	Effect	Warning
	Elimination	<ul style="list-style-type: none"> • Acknowledge the auxiliary function output immediately • Correct MD "Maximum cycle time in user program" • Program G09 in the relevant block to achieve a defined interpolation stop
	Acknowl.	Clear error with the CANCEL key.
26 070	Channel %1 axis %2 Max. number of independent CPU axes exceeded	
	Cause	%1 = channel number; %2 = axis name The number of axes defined in the parameter "Independent CPU axis" was exceeded.
	Effect	Warning
	Elimination	Increase the number of independent CPU axes.
	Acknowl.	Switch of the FM and back on again.
26 072	Channel %1 Axis %2 cannot be traversed as an independent CPU axis	
	Cause	%1 = channel number; %2 = axis name The axis can only assume the state "independent axis" in the states neutral axis, CPU axis or positioning axis within a synchronized action.
	Effect	Warning
	Elimination	Use RELEASE or WAITP to make the axis a neutral axis
	Acknowl.	Switch of the FM and back on again.
26 074	Channel %1 Axis %2 is an independent CPU axis	
	Cause	%1 = channel number; %2 = axis name The axis traverses as an independent CPU axis and cannot be traversed by the FM in the current state.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	<ul style="list-style-type: none"> • Use axial reset (user DB "AXy", DBX65.1+m) to end the motion. • Modify the NC program accordingly.
	Acknowl.	Use the CANCEL key to clear the error.

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Axis errors		
26 101	Axis %1 Drive %2 no communication	
Cause	%1 = axis name; %2 = axis number The drive does not communicate.	
Effect	<ul style="list-style-type: none"> • No ready message (no channel ready) • Start disable • Stop • The FM switches to the follow-up mode. • The axes are no longer synchronized with the machine actual value (reference point). 	
Elimination	<ul style="list-style-type: none"> • Check the bus configuration. • Check all connections (check connectors, check whether the option module is inactive etc.) 	
Acknowl.	Clear the error with "Reset".	
26 102	Axis %1 drive %2 sign-of-life failure	
Cause	%1 = axis name; %2 = axis number The sign of life was not updated by the drive.	
Effect	<ul style="list-style-type: none"> • No ready message (no channel ready) • Start disable • Stop • The FM switches to the follow-up mode. • The axes are no longer referenced with the machine actual value (reference point). 	
Elimination	<ul style="list-style-type: none"> • Check the clock settings. • Extend the cycle time if necessary. • Restart the drive. • Check the drive software. 	
Acknowl.	Clear the error with "Reset".	

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy				
Functional errors					
380 001	PROFIBUS DP: Power-up error, cause %1 parameter %2 %3 %4				
	Cause	%1 = error cause; %2 = parameter 1; %3 = parameter 2; %4 = parameter 3 An error occurred during power-up of the PROFIBUS DP master (DPM).			
	Effect	No ready signal			
	Elimination	<ul style="list-style-type: none"> • Check connection • If the error cannot be remedied by switching off/on and checking the connection, please make a note of the error no. (code for error cause and parameters 1...3) and contact the SIEMENS AG Hotline see Preface. • Replace the FM. 			
	Acknowl.	Switch FM off/on			
380 003	PROFIBUS DP: Operating error, cause %1 parameter %2 %3 %4				
	Cause	%1 = error cause; %2 = parameter 1; %3 = parameter 2; %4 = parameter 3 An operating error has occurred on the PROFIBUS DP during cyclic operation.			
	Error cause		Parameter 1	Parameter 2	Parameter 3
	01 = error		error class	logical address	
	02 = DPM cycle time-out				
	03 = DPM cycle status				
	04 = DPM cycle error				
	For the error class, see error 380 060 The following causes are possible:				
	<ul style="list-style-type: none"> • For error cause 01: <ul style="list-style-type: none"> – Fault in data communication via PROFIBUS DP – Fault on drive (SIMODRIVE 611-U) • For error causes 02, 03, 04: <ul style="list-style-type: none"> Internal error on FM 				
	Effect	No ready signal			
Elimination	For error cause 01: <ul style="list-style-type: none"> • Please check compliance with the electrical and interference specifications for PROFIBUS DP, check the cabling. • Check the terminating resistors of the PROFIBUS DP connectors (position ON at cable ends, otherwise position OFF required). • Check the drive (SIMODRIVE 611-U). For error causes 02, 03, 04: <ul style="list-style-type: none"> • If the error cannot be remedied by switching off/on and checking the connection, please make a note of the error no. (code for error cause and parameters 1...3) and contact the SIEMENS AG Hotline Tel. +49 911/895-7000. • Replace the FM. 				
Acknowl.	Switch FM off/on				

Table 12-3 Error list, continued

Error no.	Error message, error analysis and remedy	
Functional errors		
380 020	PROFIBUS-DP: SDB1000 error %1 for SDB source %2	
Cause	%1 = error cause; %2 = SDB1000 source Error cause: 01 = SDB1000 source does not exist 02 = SDB1000 source too large 03 = SDB1000 source cannot be activated SDB1000 source: 00 = Default SDB (no axis module selected) 01 = SDB1 02 = SDB2 ... 100 = SDB1000 contained in the battery-backed memory (SRAM) 101 = User SDB1000 contained in the file system 102 = SDB1000 reloaded into the SRAM	
Effect	<ul style="list-style-type: none"> • PROFIBUS DP is inactive • No readiness message (no FM READY) • Start disable 	
Elimination	<ul style="list-style-type: none"> • Select the appropriate (internal) SDB • Check the external SDB; reload if necessary 	
Acknowl.	Switch off the FM and back on again.	
380 060	PROFIBUS DP: Error %1 at logical address %2 of non-assigned station	
Cause	%1 = error class; %2 = logical address A drive (SIMODRIVE 611-U) is connected but is not assigned to an axis. Error class: 01 = Station return 02 = Station failure It is possible to operate the FM.	
Effect	Warning	
Elimination	<ul style="list-style-type: none"> • Assign the drive to an axis • Disconnect the drive from PROFIBUS DP 	
Acknowl.	Clear error with the CANCEL key.	
380 500	PROFIBUS DP: Fault drive %1, code %2, value %3, time %4	
Cause	%1 = Axis %2 = fault code of the drive (P824) %3 = Störwert des Antriebs (P826); %4 = fault time of the drive (P825) Contents of the assigned drive	
Effect	Warning	
Elimination	For fault codes/fault values, see drive documentation	
Acknowl.	Correct the error.	

Error list of all possible errors

Errors marked with an * affect functions that are not available in the FM 357-2.

000000	No (further) error exists
001000	System error %1
001001	System error %1
001002	System error %1
001003	Alarm pointer for this selfclearing alarm %1 is zero
001004	Alarm reaction to NCK alarm incorrectly configured
001005	Operating system error %1 parameter %2 %3 %4
001010	Channel %1 system error %2 %3
001011	Channel %1 %3 %4 system error %2
001012	Channel %1 system error %2 %3 %4
001013	Channel %1 system error %2
001014	Channel %1 system error %2
001015	Channel %1 axis %2 system error %3
001016	Channel %1 axis %2 system error %3
001017	Channel %1 axis %2 system error %3
001018	Floating point arithmetic error in channel %1 task %2 station %3 FPU status: %4
001019	Floating point arithmetic error in address %3 in channel %1 task %2 FPU status: %4
001030*	System-Error in Link-Modul Error-Code %1 Error-Type %2
001031*	Error in Link-Modul without specification %1 Ncu: %2 %3 %4
001100	No license or incorrect license
001160	Assertion failed in %1 : line %2
002000	Sign of life monitoring: CPU not alive
002001	CPU has not started up
002100	Battery warning threshold reached
002101	Battery alarm
002102	NCK battery alarm
002110*	NCK temperature alarm
002120*	NCK fan alarm
002130	The power supply of encoder 5 V or 24 V has failed
002140	The current service switch position forces the SRAM to be cleared at the next Power On (general reset active).
002190*	Hardware plug-in module for communication with the digitizer missing
002192*	No NCU-Link Module available, Machine data %1 reset
002193*	Safety Integrated™ not available for link axis.
002195*	Channel %1 axis %2 no punching nor nibbling available via nculink
002196*	Link axis active and \$MN_MM_SERVO_FIFO_SIZE != %1
002900	Reboot is carried out with delay
003000	EMERGENCY STOP
003001	Internal EMERGENCY STOP
004000	Channel %1 machine data %2 [%3] has gap in axis assignment
004001*	Channel %1 axis %2 defined for more than one channel via machine data %3
004002	Channel %1 machine data %2 [%3] assigns an axis not defined in channel
004003*	Axis %1 Assignment of master channel in machine data %2 incorrect or missing
004004	Channel %1 machine data %2 axis %3 multiple definition as geometry axis
004005	Number of Axis in Channel %1 exceeded. Limit %1
004007*	Axis %1 assignment of master nck in machine data %2 incorrect or missing
004010	Invalid identifier used in machine data %1 [%2]
004011	Channel %1 machine data %2 [%3] Invalid identifier used
004012	Invalid identifier used in machine data %1 [%2]
004013	Invalid NCU-Link configuration by machine data %1 = %2, to NCU_1 = %3

- 004014* Axis %1 multiple defined in %2
- 004016* Axis %1 already used by NCU %2
- 004017* Axiscontainer %1, Slot %2 already used by NCU %3
- 004018* axcontainer %1, location %2 is not used from a Channel
- 004019* axcontainer %1 switch not allowed in the current state from NCU %2: %3 ;
- 004020 Identifier %1 used several times in machine data %2
- 004021 Channel %1 identifier %2 used several times in machine data %3
- 004022* axcontainer %3 switch not allowed: ext. offset setting active channel %1 axis %2
- 004023* axcontainer %1 switch not allowed, switch from container %2 is active
- 004024* axconfiguration MD %1[%2] wrong because missing axcontainer machine data
- 004026* Machine data %1[%2], link-axes NC%3_AX%4 is not used from a Channel
- 004027* Attention: Md %1 is changed also for all other axes from the axcontainer %2
- 004028* Attention the axial MDs from the axes of the axcontainers was adapted
- 004029* Attention: by the next poweron the axial MDs of the axcontainer %1 will be adapted
- 004030 Channel %1 Missing identifier in machine data %2 [%3]
- 004032 Channel %1 Incorrect identifier for transverse axis in %2
- 004033* CAUTION: Could not yet build up NCU link communication
- 004034* Local link axis %1 with different interpolation clock = %2 /%3 not permitted
- 004035* Interpolation clock of NCU%1 = %2 does not match with NCU%3 = %4
- 004036* Faulty NCU link configuration through MD %1
- 004040 Channel %1 identifier %2 not consistent with machine data %3
- 004045 Channel %1 Conflict between machine data %2 and machine data %3
- 004050 NC code identifier %1 cannot be reconfigured to %2
- 004060 Default MD loaded
- 004062 Backup data loaded
- 004065* Buffered memory restored from backup medium (potential loss of data !)
- 004066* Battery-backed memory of the FFS was restored from backup copy (data loss possible!)
- 004070 Normalizing machine data has been altered
- 004073 Compile cycle functions define machine data number %1 more than once
- 004075 Machine data %1 (and maybe others) not altered permission level %2 needed
- 004076 %1 machine data could not be altered with permission level %2
- 004077 New value %1 of MD %2 not set. Requires %3 bytes too much %4 memory.
- 004080* Incorrect configuration of indexing axis in machine data %1
- 004090 too many erros during startup
- 004100* System cycle time corrected for digital drive
- 004101* Position control cycle for digital drive reduced to %1 ms
- 004102 Different standard clocks of the drives
- 004110 IPO cycle factor increased to %1
- 004111 IPO cycle increased to %1 ms
- 004112 Servo cycle changed to %1 ms
- 004113 Sysclock cycle changed to %1 ms
- 004114 Error in DP cycle of SDB1000
- 004115 Time ratio of communication tasak to IPO changed to %1
- 004150 Channel %1 Invalid M function subprogram call configured
- 004152* Illegal configuration of the function 'Block display with absolute values
- 004160 Channel %1 Invalid M function number configured for spindle switchover
- 004170* Illegal M function number configured for channel configuration
- 004180* Invalid M function number configured for ASUP activation
- 004181* Channel %1 Illegal assignment of an auxiliary M function number
- 004182* Channel %1 Illegal auxiliary M function number reset in %2 %3,MD
- 004183* Channel %1 Auxiliary M function number %2 used severaly times (%3 and %4)
- 004184 Channel %1 Illegal predefined auxiliary function in %2%3, MD reset

004185	Channel %1 Illegal configuration of an auxiliary function %2 %3 %4
004200	Channel %1 Geometry axis %2 must not be declared a rotary axis
004210*	Channel %1 Spindle %2 declaration as rotary axis missing
004215*	Channel %1 Spindle %2 declaration as modulo axis missing
004220*	Channel %1 Spindle %2 declared repeatedly
004225	Channel %1 axis %2 declaration as rotary axis missing
004230*	Channel %1 Data alteration from external not possible in current channel state
004240	Runtime overflow for IPO cycle or position controller cycle IP %1
004250	FastPlcCom functionality no longer available
004252	PLCIO read error: %1
004254	PLCIO write error: %1
004260	Wrong maschin data %1
004270	Machine data %1 assigns nonactivated NCK input/output byte %2
004275	Machine data %1 and %2 both assign the same NCK output byte no. %3
004280	Assignment of NCK-input/output byte via MD %1 [%2] does not match hardware configuration
004282	Hardware of external NCK outputs assigned repeatedly
004285	Error on terminal block %1, error code %2
004290	Sign-of-life monitoring: Local P bus not alive
004291	Failure of module in local P bus slot %1, error codes %2 %3 %4
004300	Declaration in machine data %1 is not allowed for geometry axis/spindle %2.
004310	Declaration in machine data %1 index 2% is not allowed.
004320	Axis %1 function %2 %3 and %4 not permitted
004340	Channel %1 Invalid transformation type in transformation no. %2
004341	Channel %1 No data block available for transformation no. %2
004342	Channel %1 Invalid machine data for general 5-axis transformation error no. %2
004343	Channel %1 Attempt to change machine data of an active transformation.
004345	Channel %1 Parameter error in chained transformation no. %2
004346	Channel %1 Incorrect geometry axis assignment in machine data %2 [%3]
004347	Channel %1 Incorrect channel axis assignment in machine data %2 [%3]
004350	Channel %1 axis identifier %2 machine data %3 not consistent with machine data %4
004400	Machine data alteration causes reorganization of buffered memory (loss of data !)
004502	Channel %1 %2(%3) invalid, %4 was adapted
005000	Cannot execute communication task
006000	Memory reorganized using standard machine data
006010	Channel %1 data block %2 not or not completely created, error number %3
006020	Machine data have been altered – now memory is reorganized
006030	Limit of user memory has been adapted
006035	Instead of %1 kB, the system has only %2 kB of free user memory of type %3
006100*	Error creating %1, error number %2 %
006401*	Channel %1 Tool not changed: No empty location for tool %2 Duplo no. %3 on magazine %4.
006402*	Channel %1 Tool not changed. Magazine no. %2 not available
006403*	Channel %1 Tool not changed. Magazine location %2 on magazine %3 not available
006404*	Channel %1 Tool not changed. Tool %2 not available or missing
006405*	Channel %1 Command %2 has invalid PLC acknowledge parameter %3 – identifier = %4
006406*	Channel %1 PLC acknowledge for command %2 is missing
006407*	Channel %1 Tool %2 cannot be placed in the magazine %3 in location %4. Invalid definition of magazine!

006410*	TO-unit %1 tool %2 duplo no. %3 has reached monitor warning limit with D= %4
006411*	Channel %1 tool %2 duplo no. %3 has reached monitor warning limit with D= %4
006412*	TO-unit %1 tool %2 duplo no. %3 has reached monitor limit with D= %4
006413*	Channel %1 tool %2 duplo no. %3 has reached monitor limit with D= %4
006415	TO unit %1 tool %2 with cutting edge no. %3 has reached tool prewarning limit
006421*	Channel %1 Tool not moved. Empty location for tool %2 Duplo no. %3 on magazine %4 not available
006422*	Channel %1 Tool not moved. Magazine no. %2 not available
006423*	Channel %1 Tool not moved. Magazine location %2 on magazine %3 not available
006424*	Channel %1 Tool not moved. Tool %2 not available or missing
006425*	Channel %1 Tool %2 cannot be placed in the magazine %3 in location %4. Invalid definition of magazine.
006430*	Workpiece counter: Overflow in table of monitored cutting edges.
006431*	Function not allowed. Tool management is not active.
006432*	Function not allowed. No tool in spindle.
006433*	Channel %1 block %2 %3 not available with tool management
006434*	Channel %1 block %2 Language command SETMTH not permitted, since toolholder function not active
006441*	Writing \$P_USEKT not permitted
006442*	Channel %1 Function cannot be executed. No tool installed on the desired magazine/location %2
006450*	Channel %1 No tool change possible. Illegal machine location no. %2 in intermediate buffer magazine
006451*	Channel %1 No tool change possible. No intermediate buffer magazine defined
006452*	Channel %1 No tool change possible. Toolholder no./spindle no.= %2 not defined
006453*	Channel %1 No tool change possible. No assignment between tool holder/spindle no.= %2 and intermediate buffer location %3
006454*	Channel %1 No tool change possible. No distance relation available.
006500	NC memory is full
006510	Too many files in the NC memory
006520	Too many protocol files in the NC memory
006530	Too many files in directory
006540	Too many directories in the NC memory
006550	Too many subdirectories
006560	Data format not allowed
006570	NC memory full
006580	NC memory limit reached"
006600	NC card memory is full
006610	Too many files open on NC card
006620	NC card has incorrect format.
006630	NC card hardware is defective.
006640	NC card is not plugged in.
006650	Write protection of NC card active.
006660	'Flash File System' option not set.
006670	Reading from NC card
006671	Writing to NC card
006690*	It is not possible to copy files from the NC card into the passive file system
006691*	It is not possible to copy files from the passive file system to the NC card
006692*	File %1 was lost
006693*	File %1 was lost
006698	Unknown NC card (%1/%2). Cannot write.

006700	Channel %1. The value of machine data %2%3 is too small.
007000*	Too many compile cycle alarms defined
007010*	Range of MMC alarm numbers for compile cycles exceeded
007020*	Compile cycle alarm number has not been defined
008036	Option 'Activation of FAST-IPO LINK' not set"
008038	Option 'Activation of more than %1 lead link axes' not set
007100*	VDI area of %1 input and %2 output bytes defined by compile cycles. Maximum %3 bytes available.
007200*	Problem with externally linked CC %1 %2"
007201*	Assertion error in %1 line %2"
008000	Channel %1 Option "Interrupt routines" not set
008010	Option "Activation of more than %1 axes" not set
008020*	Option "Activation of more than %1 channels" not set
008021*	Option "Activation of more than %1 mode groups" not set
008022	Option "Activation of more than %1 KB SRAM" not set
008030*	Channel %1 block %2 Option "Interpolation of more than 4 axes" not set
008032	Option "Activation of more than %1 link axes" not set
008034	Option "Activation of axis containers" not set
008036*	Option 'Activation of FAST-IPO LINK' not set
008038*	Option 'Activation of more than %1 lead link axes' not set
008040	Machine data %1 reset, corresponding option is not set
008041	Axis %1: Machine data %2 deleted, corresponding option not sufficient.
008044	Option for IPO cycle time %1 ms not set
008080*	%1 options have been set and no license key was entered for licensing.
008081*	%1 options have been set which are not licensed by the license key.
008082*	The license key was entered three times; before you enter it once more, carry out PowerON.
008098	Illegal combination of options (%1)
008100	Channel %1 block %2 : Function not possible
010203	Channel %1 Start without reference point (action=%2)
010207*	Channel %1 Error when selecting or deselecting the digitize function
010208	Channel %1 Continue program with Start
010209	Channel %1 Internal Stop after block search
010222*	Channel %1 Inter-channel communication not possible
010223	Channel %1: Command %2 already active
010225	Channel %1: Command %2 refused
010299	Channel %1 Function is not enabled
010600*	Channel %1 block %2 Auxiliary function during thread cutting active
010601*	Channel %1 block %2 Zero velocity at block end point during thread cutting
010604	Channel %1 block %2 Increase in thread lead too high
010605	Channel %1 block %2 Decrease in thread lead too high
010607*	Channel %1 block %2 Thread with frame not executable
010610	Channel %1 axis %2 not stopped
010620	Channel %1 block %3 axis %2 at software limit switch %4
010621	Channel %1 axis %2 is resting on software limit switch %3
010630	Channel %1 block %2 axis %3 at working area limit %4
010631	Channel %1 axis %2 is resting on working area limitation %3
010650	Channel %1 axis %2 Incorrect gantry machine data, error code %3
010651	Channel %1 Gantry grouping undefined. %2
010652	Channel %1 axis %2 Gantry warning threshold exceeded
010653	Channel %1 axis %2 Gantry trip limit exceeded
010654	Channel %1 Waiting for synchronization start of gantry group %2
010655	Channel %1 Synchronization of gantry group %2

010656	Channel %1 axis %2 Unused gantry alarm
010700	Channel %1 block %2 Protection zone %3 violated in Automatic or MDI mode
010701	Channel %1 block %2 Channel-specific protection zone %3 violated in Automatic or MDI mode
010702	Channel %1 Protection zone %2 violated in Jogging or Incremental relative mode
010703	Channel %1 Channel-specific protection zone %2 violated in manual mode
010704	Channel %1 block %2 Protection zone monitoring is not guaranteed.
010706	Channel %1 Protection zone %2 reached with axis %3 in Jogging or Incremental relative mode
010707	Channel %1 Channel-specific protection zone %2 reached with axis %3 in manual mode
010710*	Channel %1 block %2 Conflict with centerless grinding
010720	Channel %1 block %3 axis %2 software limit switch %4
010721	Channel %1 block %3 axis %2 software limit switch %4
010730	Channel %1 block %3 axis %2 working area limitation %4
010731	Channel %1 block %3 axis %2 working area limitation %4
010740*	Channel %1 block %2 Too many empty blocks in WAB programming
010741*	Channel %1 block %2 Direction reversal with WAB infeed motion
010742*	Channel %1 block %2 WAB distance invalid or not programmed
010743*	Channel %1 block %2 WAB programmed several times
010744*	Channel %1 block %2 No valid WAB direction defined
010745*	Channel %1 block %2 WAB end positioning not clear
010746*	Channel %1 block %2 Block search stop for WAB
010747*	Channel %1 block %2 Retraction direction not defined for WAB
010748	Channel %1 block %2 Illegal retraction plane with SAR
010750*	Channel %1 block %2 Tool radius compensation activated without tool number
010751*	Channel %1 block %2 Danger of collision due to tool radius compensation
010752*	Channel %1 block %2 Overflow of local block buffer with tool radius compensation
010753*	Channel %1 block %2 Activate tool radius compensation in linear block only
010754*	Channel %1 block %2 Deactivate tool radius compensation in linear block only
010755*	Channel %1 block %2 Do not activate tool radius compensation via KONT at the current starting point
010756*	Channel %1 block %2 Do not deactivate tool radius compensation via KONT at the programmed end point
010757*	Channel %1 block %2 Do not change the compensation plane while tool radius compensation is active
010758*	Channel %1 block %2 Curvature radius with variable compensation value too small
010759*	Channel %1 block %2 Path is parallel to tool orientation
010760*	Channel %1 block %2 Helical axis is not parallel to tool orientation
010761*	Channel %1 block %2 Tool radius compensation for ellipse with more than one revolution not possible
010762*	Channel %1 block %2 Too many empty blocks between two traversing blocks with active tool radius compensation
010763*	Channel %1 block %2 Path component of the block in the compensation plane becomes zero.
010764*	Channel %1 block %2 Discontinuous path with active tool radius compensation
010765*	Channel %1 block %2 3D tool radius compensation not possible
010766*	Channel %1 Illegal change of surface orientation between block %2 and block %3
010767*	Channel %1 block %2 Processing with tilt angle unequal 0 not possible

010768*	Channel %1 block %2 Illegal tool orientation with 3D tool radius compensation
010769*	Channel %1 block %2 Illegal surface normal vector with 3D tool radius compensation
010770*	Channel %1 block %2 Change of corner type due to change of orientation with active tool radius compensation
010771*	Channel %1 block %2 Overflow of local block buffer due to orientation smoothing
010772*	Channel %1 block %2 Illegal orientation change or deactivation of 3D face milling
010773*	Channel %1 Illegal tool orientation in block %2 at inside corner with block %3
010774*	Channel %1 Illegal tool dimensions with face cutting in block %2
010775*	Channel %1 Illegal tool change with face cutting in block %2
010776*	Channel %1 block %2 axis %3 must be geo axis if cutter compensation is active.
010777*	Channel %1 block %2 Tool radius compensation: Too many blocks with suppression of compensation
010778*	Channel %1 block %2 Preparation stop with active tool radius compensation
010779	Channel %1 block %2 Tool with tool point direction on 3D circumferential milling
010780	Channel %1 block %2 Illegal tool change with tool radius compensation
010781*	Channel %1 block %2 Involute with illegal orientation in tool radius compensation
010782*	Channel %1 block %2 Illegal curve type in tool radius compensation
010783*	Channel %1 block %2 Tool radius compensation type requires orientierung transformation
010784*	Channel %1 block %2 Illegal tool in tool radius compensation with bordering faces
010790	Channel %1 block %2 Plane change in linear programming with angles
010791	Channel %1 block %2 Invalid angle in linear programming
010792	Channel %1 block %2 Illegal interpolation type in linear programming with angles
010793	Channel %1 block %2 Second block missing in linear interpolation with angles
010794	Channel %1 block %2 in 2nd block missing in linear interpolation with angles
010795	Channel %1 block %2 End point conflict in angle programming
010800	Channel %1 block %3 axis %2 is not a geometry axis
010805	Channel %1 block %2 Repositioning after switch of geoaxes or transformation
010810*	Channel %1 block %2 Master spindle not defined
010820	Channel %1 Rotary axis/spindle %2 not defined
010860	Channel %1 block %2 Feedrate not programmed
010861	Channel %1 block %2 Velocity of positioning axis %3 is zero
010862*	Channel %1 block %2 Master spindle is axis of path.
010870*	Channel %1 block %2 Facing axis not defined
010880*	Channel %1 block %2 Too many empty blocks between two traversing blocks when inserting chamfer or radius
010881	Channel %1 block %2 Overflow of the local block buffer with chamfers or radii
010882*	Channel %1 block %2 Do not activate chamfer or radius (non-modal) without traversing movement in block
010883*	Channel %1 block %2 Chamfer or rounding must be reduced
010890	Channel %1 block %2 Overflow of local block buffer when calculating splines
010891	Channel %1 block %2 Multiplicity of node is greater than its order.
010900*	Channel %1 block %2 No S value programmed for constant cutting speed
010910	Channel %1 block %2 Excessive velocity of one path axis
010911*	Channel %1 block %2 Transformation prohibits traversal of the pole.
010912*	Channel %1 block %2 Preparation and main run might not be synchronized

010913	Channel %1 block %2 Negative feed profile is omitted
010914*	Movement not possible while transformation active – in channel %1 for block %2
010930*	Channel %1 block %2 Interpolation type not allowed in stock removal contour
010931*	Channel %1 block %2 Error in programmed stock removal contour
010932*	Channel %1 block %2 Preparation of contour has been restarted
010933*	Channel %1 block %2 Contour program contains too few contour blocks
010934*	Channel %1 block %2 Array for contour segmentation is set too small
010940	Channel %1 block %2 curve table %3: cannot be deleted/overwritten
010941	Channel %1 block %2 curve table %3: NC memory full
010942	Channel %1 block %2 curve table %3: Illegal instruction in definition
010943	Channel %1 block %2 curve table %3: Direction change of lead value in the block not allowed
010944*	Channel %1 block %2 curve table %3: Illegal transformation
010945	Channel %1 block %2 curve table %3: Illegal coupling of axes
010946	Channel %1 block %2 curve table %3: No contour defined
010947	Channel %1 block %2 curve table %3: Discontinuous contour
010948	Channel %1 block %2 curve table %3: Position jump at edge of period
010949	Channel %1 block %2 curve table %3: No lead axis movement
010950	Channel %1 Calculation of arc length function is not accurate enough
010951	Channel %1 block %2 curve table %3: Follow-up value is zero
010955	Channel %1 block %2 curve table %3: Missing master axis motion
010956	Channel %1 block %2 curve table %3: NC memory limit DRAM reached
010960*	Channel %1 block %2 COMPCURV/COMPCAD and tool path correction cannot be used simultaneously
010961*	Channel %1 block %2 With the radius compensation active, max. cubical polynomials are permitted
010962*	Channel %1 block %2 Function %3 not possible with path correction
012000	Channel %1 block %2 address %3 programmed repeatedly
012010	Channel %1 block %2 address %3 address type programmed too often
012020	Channel %1 block %2 Combination of address modification not allowed
012030	Channel %1 block %2 Invalid arguments or data types in %3
012040*	Channel %1 block %2 Expression %3 is not of data type "AXIS"
012050	Channel %1 block %2 address %3 does not exist
012060	Channel %1 block %2 Same G group programmed repeatedly
012070	Channel %1 block %2 Too many syntax-defining G functions
012080	Channel %1 block %2 Syntax error in text %3
012090	Channel %1 block %2 Unexpected argument %3
012100	Channel %1 block %2 Number of passes %3 not permissible
012110	Channel %1 block %2 Syntax cannot be interpreted
012120	Channel %1 block %2 G function not programmed alone
012130*	Channel %1 block %2 Tool orientation not permissible
012140	Channel %1 block %2 Expression %3 not contained in this release
012150	Channel %1 block %2 Operation %3 not compatible with data type
012160	Channel %1 block %2 Value range exceeded
012170	Channel %1 block %2 Identifier %3 defined repeatedly
012180	Channel %1 block %2 Illegal chaining of operators %3
012190*	Channel %1 block %2 Variable of type ARRAY has too many dimensions
012200	Channel %1 block %2 Symbol %3 cannot be created
012210*	Channel %1 block %2 String %3 too long
012220*	Channel %1 block %2 Binary constant %3 in string too long
012230*	Channel %1 block %2 Hexadecimal constant %3 in string too long
012240*	Channel %1 block %2 Tool orientation %3 defined repeatedly
012250*	Channel %1 block %2 Do not nest macro %3

012260	Channel %1 block %2 Too many initialization values given for %3
012261	Channel %1 block %2 Initialization of %3 not allowed
012270*	Channel %1 block %2 Macro identifier %3 already defined
012280*	Channel %1 block %2 Maximum macro length %3 exceeded
012290	Channel %1 block %2 Arithmetic variable %3 not defined
012300	Channel %1 block %2 Call-by-reference argument missing on subroutine call %3
012310	Channel %1 block %2 Axis argument missing on procedure call %3
012320	Channel %1 block %2 Argument %3 must be call-by-reference
012330	Channel %1 block %2 Type of argument %3 incorrect
012340	Channel %1 block %2 Number of arguments exceeded in %3
012350	Channel %1 block %2 Argument %3 not accepted because AXIS argument is missing
012360	Channel %1 block %2 Dimension of argument %3 incorrect
012370	Channel %1 block %2 Range of values exceeded for %3
012380	Channel %1 block %2 Maximum memory capacity exceeded
012390	Channel %1 block %2 Type of initial value for %3 cannot be converted
012400	Channel %1 block %2 array %3 Index does not exist
012410	Channel %1 block %2 Incorrect index type for %3
012420	Channel %1 block %2 Identifier %3 too long
012430	Channel %1 block %2 Invalid index
012440	Channel %1 block %2 Maximum number of formal arguments exceeded
012450	Channel %1 block %2 label defined repeatedly
012460	Channel %1 block %2 Maximum number of symbols exceeded with %3
012470	Channel %1 block %2 Unknown G function %3 used
012480	Channel %1 block %2 Subroutine %3 already defined
012490	Channel %1 block %2 Access permission level %3 is not valid
012500	Channel %1 block %2 Do not use %3 in this module
012510	Channel %1 block %2 Too many machine data %3
012520	Channel %1 block %2 Too many tool data %3
012530	Channel %1 block %2 Invalid index for %3
012540	Channel %1 block %2 is too long or too complex
012550	Channel %1 block %2 identifier %3 not defined or option does not exist
012552*	Channel %1 block %2 No tool/magazine OEM parameters defined. Option not set.
012560	Channel %1 block %2 Programmed value %3 exceeds allowed limits
012570	Channel %1 block %2 Too many motion synchronous actions in %3
012571	Channel %1 block %2 %3 Not allowed in motion synchronous action
012572	Channel %1 block %2 %3 Only allowed in motion synchronous action
012580	Channel %1 block %2 Invalid assignment to %3 for motion synchronous action
012581	Channel %1 block %2 Illegal read access to %3 in motion synchronous action
012582	Channel %1 block %2 array index %3 invalid
012583	Channel %1 block %2 Variable %3 no system variable
012584	Channel %1 block %2 variable %3 cannot be read synchronously with motion
012585	Channel %1 block %2 variable %3 cannot be changed synchronously with motion
012586	Channel %1 block %2 Motion synchronous action: Type conflict in variable %3
012587	Channel %1 block %2 Motion synchronous action: Operator / function %3 invalid
012588	Channel %1 block %2 Motion synchronous action: Address %3 illegal
012589	Channel %1 block %2 Motion synchronous action: Variable %3 not allowed with modal ID
012590*	Channel %1 block %2 Global user data cannot be created
012600*	Channel %1 block %2 Invalid checksum of line
012610	Channel %1 block %2 Accessing single char with call-by-reference argument not allowed %3

012620	Channel %1 block %2 Accessing this variable as single char not allowed
012630*	Channel %1 block %2 Skip / label not allowed
012640*	Channel %1 block %2 Invalid nesting of control structures
012641*	Channel %1 block %2 Nesting level of control structures exceeds limit
012650*	Channel %1 block %2 axis %3 name different in channel %4
012660	Channel %1 block %2 Motion synchronous action: Variable %3 reserved for motion synchronous actions and subroutines as action
012661	Channel %1 block %2 Subprograms as action in motion synchronous action %3: No further subroutine call possible
012700	Channel %1 block %2 Contour programming not allowed because modal subprogram active
012701	Channel %1 block %2 Illegal interpolation type active for contour definition
012710	Channel %1 block %2 Illegal language element in external language mode
012720	Channel %1 block %2 Program number for macro call (G65/G66) missing
012722*	Channel %1 block %2 Block with multiple ISO_2/3 macro or cycle calls
012724*	Channel %1 block %2 No radius programmed for selecting/deselecting cylindrical interpolation"
012726*	Channel %1 block %2 Illegal plane selection with parallel axes
012728*	Channel %1 block %2 Spacing for dual revolver not set
012730*	Channel %1 block %2 No valid transformation machine data parameterized
012740	Channel %1 block %2 Calling macro %3 not possible
014000	Channel %1 block %2 Illegal end of file
014001	Channel %1 block %2 Illegal end of block
014009	Channel %1 block %2 axis %work area limitation %3
014010	Channel %1 block %2 Invalid default argument in subroutine call
014011	Channel %1 block %2 NC program %3 does not exist or not released for machining
014012	Channel %1 block %2 Lowest subroutine level exceeded
014013	Channel %1 block %2 Number of subroutine passes invalid
014014	Channel %1 Selected NC program %3 or access permission not available
014015	Channel %1: No access permission for file
014016	Channel %1 block %2 Error on subprogram call by M/T function
014017	Channel %1 block %2 Syntax error when calling a subroutine using an M function
014020	Channel %1 block %2 Incorrect value or wrong number of arguments on function or procedure call
014021	Channel %1 block %2 Incorrect value or wrong number of arguments on function or procedure call
014025	Channel %1 block %2 Motion synchronous action: Illegal modal ID
014026*	Channel %1 block %2 Motion synchronous action: Invalid polynomial in the FCTDEF command
014030	Channel %1 block %2 oscillation with feed axis: combine OSCILL and POSP
014033	Channel %1 block %2 involute: No end point programmed
014034	Channel %1 block %2 involute: Angle of rotation too large
014035	Channel %1 block %2 involute: Illegal starting point
014036	Channel %1 block %2 involute: Illegal starting point
014037	Channel %1 block %2 involute: Illegal radius
014038	Channel %1 block %2 Involute cannot be determined: End point error
014039	Channel %1 block %2 involute: End point programmed several times
014040	Channel %1 block %2 Error in end point of circle
014045*	Channel %1 block %2 Error in tangent circle programming
014048	Channel %1 block %2 Incorrect number of revolutions programmed for circle
014050	Channel %1 block %2 Nesting depth for arithmetic operations exceeded
014051	Channel %1 block %2 Arithmetic error in part program

014060*	Channel %1 block %2 Invalid skip level with differential block skip
014070	Channel %1 block %2 Memory for variables not sufficient for subroutine call
014080	Channel %1 block %2 Jump destination not found
014082	Channel %1 block %2 Label %3 not found"
014085*	Channel %1 block %2 Illegal statement"
014088	Channel %1 block %2 axis %3 suspect position
014090	Channel %1 block %2 Invalid D number
014091	Channel %1 block %2 Invalid function, index: %3
014092	Channel %1 block %2 axis %3 has wrong axis type
014093*	Channel %1 block %2 Path interval ≤ 0 with polynomial interpolation
014094*	Channel %1 block %2 Polynomial degree greater than 3 programmed for polynomial interpolation
014095	Channel %1 block %2 Circle radius has been programmed too small
014096	Channel %1 block %2 Type conversion not possible
014097*	Channel %1 block %2 String cannot be converted to AXIS type
014098	Channel %1 block %2 Conversion error: Not a number
014099	Channel %1 block %2 Result in string concatenation too long
014100	Channel %1 block %2 Orientation transformation not available
014101*	Channel %1 block %2 Orientation transformation not active
014102*	Channel %1 block %2 polynomial degree greater than 5 programmed for angle of orientation vector interpolation
014110*	Channel %1 block %2 Do not mix use of Euler angles and orientation vector components
014111*	Channel %1 block %2 Do not mix use of Euler angles, orientation vector and transformation axes
014112*	Channel %1 block %2 Programmed orientation path not possible
014113*	Channel %1 block %2 Programmed lead angle too large
014114*	Channel %1 block %2 Programmed tilt angle too large
014115*	Channel %1 block %2 Illegal definition of part surface
014116*	Channel %1 block %2 Absolute orientation programmed while ORIPATH is active
014117*	Channel %1 block %2 No angle or direction of taper programmed
014118*	Channel %1 block %2 No end orientation programmed
014119*	Channel %1 block %2 No intermediate orientation programmed
014120*	Channel %1 block %2 Determination of plane not possible for programmed
014122*	Channel %1 block %2 Angle and direction of taper programmed
014123*	Channel %1 block %2 Aperture angle of taper too small
014124*	Channel %1 block %2 Start tangent for orientation is zero
014125*	Channel %1 block %2 Programmed rotation not possible orientation
014129	Channel %1 block %2 Do not mix use of orientation axes and vector components
014130	Channel %1 block %2 Too many initialization values given
014131	Channel %1 block %2 Do not mix use of orientation axes and lead/tilt angle
014132	Channel %1 block %2 Incorrect configuration of orientation axes
014133	Channel %1 block %2 G code not allowed for orientation definition
014134	Channel %1 block %2 G code not allowed for orientation interpolation
014140	Channel %1 block %2 Programming of position without transformation not allowed
014144	Channel %1 block %2 PTP motion not allowed
014146	Channel %1 block %2 CP or PTP motion not allowed without transformation
014148*	Channel %1 Illegal reference system for manual traversing with Cartesian coordinates

014150*	Channel %1 block %2 Illegal tool carrier number programmed or declared (MD)
014151*	Channel %1 block %2 Illegal tool carrier rotation
014152*	Channel %1 block %2 Tool carrier: Invalid orientation
014153*	Channel %1 block %2 Unknown tool carrier type: %3
014155*	Channel %1 block %2 invalid definition of base frame for tool carrier offset
014156*	Channel %1 Error selecting toolholder at RESET"
014157*	Channel %1 block %2 illegal interpolation type with MOV T
014159	Channel %1 block %2 more then 2 to angles programmed with ROTS or AROTS
014160*	Channel %1 block %2 tool length compensation activation without geo axis
014165*	Channel %1 block %2 active T-number does not match activated tool
014170*	Channel %1 block %2 illegal interpolation type with tool lengt compensation
014180	Channel %1 block %2 H-code undefined
014185*	Channel %1 block %2 D-code undefined
014190*	Channel %1 block %2 H-code and G49 simultaneously
014195*	Channel %1 block %2 D-code and G49 simultaneously
014197*	Channel %1 block %2 D-code and H-code simultaneously
014198*	Channel %1 block %2 Illegal change of tool direction with tool offset
014199*	Channel %1 block %2 Illegal plane change for tool with diameter component
014200	Channel %1 block %2 Polar radius negative
014210	Channel %1 block %2 Polar angle too large
014250	Channel %1 block %2 Pole radius negative
014260	Channel %1 block %2 Pole angle too large
014270	Channel %1 block %2 Pole programmed incorrectly
014280	Channel %1 block %2 Polar coordinates programmed incorrectly
014290	Channel %1 block %2 Degree of polynomial programmed greater than 5 for polynomial interpolation
014300*	Channel %1 block %2 Overlaid handwheel motion activated incorrectly
014310*	Handwheel %1 configuration not correct or inactive
014400*	Channel %1 block %2 Tool radius compensation active at transformation switchover
014401*	Channel %1 block %2 Transformation not available
014402*	Channel %1 block %2 Spline active at transformation change
014403	Channel %1 block %2 Preparation and main run might not be synchronized
014404*	Channel %1 block %2 Invalid argument in selection of transformation
014410	Channel %1 block %2 Spline active at change of geoaxis
014411*	Channel %1 block %2 Tool radius compensation active at change of geoaxis
014412*	Channel %1 block %2 Transformation active at change of geoaxis
014413*	Channel %1 block %2 Fine tool correction: Changeover geometry / channel axis not allowed
014414*	Channel %1 block %2 GEOAX function: Incorrect call
014415	Channel %1 block %2 Tangential control: Geometry/channel axis switchover not permitted
014420*	Channel %1 block %2 Indexing axis %3 frame not allowed
014500	Channel %1 block %2 Illegal DEF or PROC statement within part program
014510	Channel %1 block %2 PROC statement missing on subroutine call
014520	Channel %1 block %2 Illegal PROC statement in data definition section
014530	Channel %1 block %2 EXTERN and PROC statement do not correspond
014600*	Channel %1 block %2 Buffer %3 for sequential reload cannot be established
014601*	Channel %1 block %2 Reload buffer cannot be erased
014602*	Channel %1 block %2 Timeout for EXTCALL
014610*	Channel %1 block %2 Compensation block not possible
014650	Channel %1 block %2 SETINT uses invalid input to trigger ASUB
014660	Channel %1 block %2 SETINT instruction uses invalid priority level

014700	Channel %1 block %2 Timeout after command to interpreter
014701	Channel %1 block %2 Number of available NC blocks reduced by %3
014710	Channel %1 block %2 Error during function %3 of INIT block generation
014720*	Channel %1 block %2 Axes missing for centerless grinding
014730*	Channel %1 block %2 Conflict at activation of centerless transformation
014740*	Channel %1 block %2 Tool data missing for centerless grinding vorhanden
014745*	Channel %1 block %2 Centerless grinding not active
014750	Channel %1 block %2 Too many auxiliary functions programmed
014751	Channel %1 block %2 Maximum number of motion synchronous actions exceeded (identifier %3)
014752	Channel %1 block %2 Conflict with DELDTG STOPREOF
014753	Channel %1 block %2 Motion synchronous action uses illegal interpolation type
014754	Channel %1 block %2 Motion synchronous action uses wrong feed type
014755	Channel %1 block %2 Motion synchronous action needs traverse motion
014756	Channel %1 block %2 Motion synchronous action uses wrong value
014757	Channel %1 block %2 Motion synchronous action uses wrong type
014758*	Channel %1 block %2 Programmed synchronous value is not available
014759	Channel %1 block %2 Motion synchronous action uses wrong axis type
014760	Channel %1 block %2 Auxiliary function of a group programmed repeatedly
014761*	Channel %1 block %2 Motion synchronous action: DELDTG not allowed with active radius compensation
014762	Channel %1 block %2 Too many PLC variables programmed
014763*	Channel %1 block %2 too many link-variables programmed
014764*	NCU-Link cannot communicate all Link-Variables immediately
014765*	NCU-Link cannot communicate all Link-Variables
014766*	NCU-Link is strongly loaded with Messages that threatens Memorylack
014767*	machinedata match over NCU-Link not completely
014770	Channel %1 block %2 Auxiliary function programmed incorrectly
014780	Channel %1 block %2 Unreleased option used
014790	Channel %1 block %2 axis %3 currently controlled by CPU
014800	Channel %1 block %2 Programmed path speed less or equal to zero
014810	Channel %1 block %2 Negative axis speed for positioning axis %3
014811	Channel %1 block %2 Acceleration value of axis/spindle %3 out of range
014812	Channel %1 block %2 Axis %3 SOFTA not available
014815	Channel %1 block %2 Negative thread lead change programmed
014820*	Channel %1 block %2 Negative maximum spindle speed programmed for constant cutting rate
014821*	Channel %1 block %2 Error in selection or disabling of GWPS
014822*	Channel %1 block %2 Incorrect programming of GWPS
014823*	Channel %1 block %2 Error on selection or disabling of tool monitoring
014824*	Channel %1 block %2 GWPS conflict
014840*	Channel %1 block %2 Value for constant cutting speed out of range
014900	Channel %1 block %2 Use either center point or end point programming
014910	Channel %1 block %2 Invalid angle of aperture for programmed circle
014920	Channel %1 block %2 Intermediate point of circle incorrect
015000*	Channel %1 block %2 Channel-sync instruction using illegal mark
015010*	Channel %1 block %2 Program coordination instruction with invalid channel number
015020*	Channel %1 block %2 Instruction CHANDATA cannot be executed. Channel %3 is not active
015021*	Channel %1 block %2 Instruction CHANDATA uses invalid channel number.
015025	CHANDATA(%2): Channel not active. Data will be ignored
015030	Channel %1 block %2 Different system of measurement settings
015100	Channel %1 block %2 REORG abort caused by overflow of log file

015110	Channel %1 block %2 REORG not possible
015150*	Channel %1 block %2 Reload from external aborted
015160	Channel %1 block %2 Wrong configuration of block buffer
015165	Channel %1 block %2 Error on compiling or interpreting PLC ASUB %3
015166	Channel %1 user system ASUP _N_ASUP_SPF does not exist
015170	Channel %1 block %2 Program %3 could not be compiled
015171*	Channel %1 block %2 Compilat %3 older than relevant subroutine
015175	Channel %1 block %2 Program %3. interfaces could not be generated
015180*	Channel %1 block %2 Program %3 cannot be executed as INI file
015185*	Channel %1 block %2 Error in INI file
015190	Channel %1 block %2 Not enough free memory for subroutine call
015300	Channel %1 block %2 Invalid number-of-passed blocks during block search
015310*	Channel %1 block %2 File requested during block search is not loaded
015320*	Channel %1 block %2 Invalid block search command
015330*	Channel %1 block %2 Invalid block number as target of block search
015340*	Channel %1 block %2 Invalid label as target of block search
015350*	Channel %1 block %2 Target of block search not found
015360*	Channel %1 Invalid target of block search (syntax error)
015370*	Channel %1 Target of block search not found
015380	Channel %1 block %2 illegal incremental programming in axis %3
015390*	Channel %1 block %2 %3 not executed in block search
015395*	Channel %1 Master/slave cannot be executed with block search
015400*	Channel %1 block %2 Selected initial ini file does not exist
015410*	Channel %1 block %2 Initialization file contains invalid M function
015420	Channel %1 block %2 Instruction not accepted in current mode
015450	Channel %1 block %2 Compiled program cannot be stored
015460	Channel %1 block %2 Syntax conflict with modal G function
015500*	Channel %1 block %2 Illegal angle of shear
015700*	Channel %1 block %2 Illegal cycles alarm number
015800*	Channel %1 block %2 Wrong starting condition for CONTPRON/CONTDCON
015810*	Channel %1 block %2 Wrong array dimension for CONTPRON/CONTDCON
015900	Channel %1 block %2 Touch probe not available
015910	Channel %1 block %2 Touch probe not available
015950	Channel %1 block %2 No traverse motion programmed
015960	Channel %1 block %2 No traverse motion programmed
016000*	Channel %1 block %2 Invalid value for lifting direction
016005*	Channel %1 block %2 Invalid value for lifting distance
016010*	Channel %1 block %2 Processing stop after rapid lift from contour
016015	Channel %1 block %2 Incorrect axis identifier %3
016016	Channel %1 block %2 No retraction position programmed for axis %3
016020	Channel %1 Repositioning in block %2 is not possible
016100*	Channel %1 block %2 spindle %3 not available in channel
016105*	Channel %1 block %2 spindle %3 cannot be assigned
016110*	Channel %1 block %2 spindle %3 for dwell time not in speed control mode
016120*	Channel %1 block %2 Invalid index to online tool compensation
016130*	Channel %1 block %2 Instruction not allowed with active FTOCON
016140*	Channel %1 block %2 FTOCON not allowed
016150*	Channel %1 block %2 Invalid spindle no. with PUTFTOCF
016200	Channel %1 block %2 Spline and polynomial interpolation not available
016300*	Channel %1 block %2 Invalid zero crossing of denominator polynomial within parameter range
016400	Channel %1 block %2 Positioning axis %3 cannot participate in spline interpolation
016410	Channel %1 block %2 axis %3 is not a geometry axis

016420	Channel %1 block %2 axis %3 repeatedly programmed
016421*	Channel %1 block %2 angle %3 repeatedly programmed in the block
016422*	Channel %1 block %2 Angle %3 programmed in the block repeatedly
016423*	Channel %1 block %2 Angle %3 programmed in the block repeatedly
016424*	Channel %1 block %2 Coordinate %3 programmed in the block repeatedly
016430	Channel %1 block %2 Geometry axis %3 cannot traverse as positioning axis in rotated coordinate system
016440	Channel %1 block %2 Rotation programmed for non-existent geometry axis.
016500*	Channel %1 block %2 Chamfer or radius negative
016510*	Channel %1 block %2 Facing axis is not defined
016700*	Channel %1 block %2 axis %3 Invalid feed type
016710*	Channel %1 block %2 axis %3 Master spindle not programmed
016715*	Channel %1 block %2 axis %3 Master spindle not at standstill
016720*	Channel %1 block %2 axis %3 Thread lead is zero
016730	Channel %1 block %2 axis %3 Wrong parameter for thread cutting
016740	Channel %1 block %2 Geometry axis must be programmed
016750*	Channel %1 block %2 axis %3 SPCON not programmed
016751*	Channel %1 block %2 spindle/axis %3 SPCOF cannot be executed.
016755	Channel %1 block %2 No wait needed
016760*	Channel %1 block %2 axis %3 S value missing
016761	Channel %1 block %2 axis/spindle %3 not programmable in channel
016762*	Channel %1 block %2 spindle %3 Function of thread or drill is active
016763*	Channel %1 block %2 axis %3 Programmed speed is illegal (zero or negative)
016770	Channel %1 block %2 axis %3 Encoder missing
016771	Channel %1 following axis %2 Overlaid motion not released
016776	Channel %1 block %2 curve table %3 does not exist for axis %4
016777	Channel %1 block %2 Master value coupling: Lead axis %4 slave axis %3 not available
016778	Channel %1 block %2 Master value coupling: Ring coupling for slave axis %3 and lead axis %4 not allowed
016779	Channel %1 block %2 Master value coupling: Too many couplings for axis %3, see active leading axis %4
016780	Channel %1 block %2 Slave axis/spindle missing
016781	Channel %1 block %2 Master axis/spindle missing
016782	Channel %1 block %2 Slave axis/spindle %3 currently not available
016783	Channel %1 block %2 Master axis/spindle %3 currently not available
016785	Channel %1 block %2 Master and slave axis/spindle %3 are identical
016787	Channel %1 block %2 Coupling parameter is not to be changed
016788	Channel %1 block %2 Resulting coupling definition is cyclic
016789	Channel %1 block %2 Axis/spindle used in another coupling definition
016790	Channel %1 block %2 Coupling parameter is zero or missing
016791	Channel %1 block %2 Coupling parameter neglected
016792	Channel %1 block %2 Too many couplings for axis/spindle %3
016793*	Channel %1 block %2 Coupling of axis %3 prohibits switchover of transformation
016794	Channel %1 block %2 Coupling of axis/spindle %3 prohibits referencing
016795	Channel %1 block %2 String cannot be interpreted
016796	Channel %1 block %2 Coupling not defined
016797	Channel %1 block %2 Coupling is active
016798*	Channel %1 block %2 axis %3 is depend axis and prohibits rotation of axcontainer
016799*	Channel %1 block %2 axis %3 is leadaxis and prohibits rotation of axcontainer
016800	Channel %1 block %2 Traverse instruction DC/CDC for axis %3 not allowed
016810	Channel %1 block %2 Traverse instruction ACP for axis %3 not allowed
016820	Channel %1 block %2 Traverse instruction ACN for axis %3 not allowed

016830	Channel %1 block %2 Invalid position for axis/spindle %3 programmed
016903	Channel %1 action %2 not allowed in current state
016904	Channel %1 action %2 not allowed in current state
016905	Channel %1 action %2 not allowed
016906	Channel %1 action %2 is aborted because of an active alarm
016907	Channel %1 action %2 only possible in Stop mode
016908	Channel %1 action %2 only possible in Reset or at the block end
016909	Channel %1 action %2 is not allowed in current mode
016911	Channel %1 Mode change is not allowed
016912	Channel %1 action %2 only possible in Reset
016913	Mode group %1 channel %2 Mode change: Action %3 not allowed
016914	Mode group %1 channel %2 Mode change: Action %3 not allowed
016915	Channel %1 action %2 not allowed in current block
016916	Channel %1 Repositioning: Action %2 not allowed in the current state
016918*	Channel %1 action %2 needs reset in all channels
016919	Channel %1 action %2 is not allowed because of an alarm
016920	Channel %1 action %2 is already enabled
016921*	Channel %1 mode group %2 machine data: Channel/mode group assignment not allowed or double
016922	Channel %1 subprograms: action %2 maximum stack level exceeded
016923	Channel %1 action %2 not allowed in current state
016924*	Channel %1 Caution: Program test will change the tool data. Make a tool/magazine data backup!
016925	Channel %1 action %2 not allowed in the current state, action %3 active
016926*	Channel %1 Channel coordination: action %2 not allowed in block %3, marker %4 already set
016927	Channel %1 action %2 at active interrupt treatment not allowed
016928	Channel %1 Interrupt handling: action %2 not possible
016930	Channel %1: Predecessor and current block %2 must be separated by an executable block.
016931	Channel %1 subprograms: action %2 maximum stack level exceeded
016932	Channel %1 Conflict on activation of user data type %2
016933	Channel %1 Interrupt handling: action %2 not allowed in current state
016934	Channel %1 Interrupt handling: action %2 not possible because of Stop
016935*	Channel %1 Action %2 not possible due to block search
016936*	Channel %1 Action %2 not possible due to active dry run feed
016937*	Channel %1 Action %2 not possible due to program test
016938	Channel %1 action %2 terminated because gear change active
016939*	Channel %1 action %2<ALNX> rejected because gear change active
016940*	Channel %1 action %2<ALNX> wait for gear change
016941	Channel %1 Action %2 denied, as program event not yet processed
016942*	Channel %1 Start program command action %2 not possible
016943*	Channel %1 Action %2 not possible due to an ASUB
016944*	Channel %1 Action %2 not possible due to active block search blocks
016945	Channel %1 Action %2 is delayed up to block end
016946*	Channel %1 Start via START no permitted
016947*	Channel %1 Start via PLC is not permitted
016948*	Channel %1 dependent channel %2 still active
016949	Relation between the mark of channel %1 and channel %2 is invalid
017000	Channel %1 block %2 Maximum number of symbols exceeded
017001*	Channel %1 block %2 No memory left for tool or magazine data
017010	Channel %1 block %2 No memory left for symbol
017020	Channel %1 block %2 1st array index out of range

017030	Channel %1 block %2 2nd array index out of range
017040	Channel %1 block %2 Illegal axis index
017050	Channel %1 block %2 Illegal value
017055	Channel %1 block %2 GUD variable does not exist
017060	Channel %1 block %2 Requested data area too large
017070	Channel %1 block %2 Data is write-protected
017080	Channel %1 block %2 %3 Value violates lower limit
017090	Channel %1 block %2 %3 Value violates upper limit
017095	Channel %1 block %2 Invalid value
017100	Channel %1 block %2 Digital input no. %3 is not active
017110	Channel %1 block %2 Digital output no. %3 is not active
017120	Channel %1 block %2 Analog input no. %3 is not active
017130	Channel %1 block %2 Analog output no. %3 is not active
017140	Channel %1 block %2 Output no. %3 is assigned to function via machine data
017150	Channel %1 block %2 Maximum %3 FM outputs per block exceeded
017160	Channel %1 block %2 Tool is not selected
017170	Channel %1 block %2 Too many symbols defined
017180	Channel %1 block %2 Illegal D number
017181	Channel %1 block %2 T no.= %3, D no.= %4 does not exist
017182	Channel %1 block %2 Illegal total offset number
017188	Channel %1 D number %2 for tool T no. %3 and %4 defined
017189	Channel %1 D number %2 of tools at magazine/location %3 and %4 defined
017190	Channel %1 block %2 Illegal T number
017191	Channel %1 block %2 T= %3 does not exist, program %4
017192*	TO-unit %1 non valid tool naming '%2', duplo no. %3. No more replacement tools in group '%4' possible
017193*	Channel %1 block %2 The active tool is no longer at toolholder number/spindle no. %3 , program %4
017194	Channel %1 block %2 Suitable tool not found
017200	Channel %1 block %2 Cannot delete tool data
017202*	Channel %1 block %2: cannot delete magazine data
017210	Channel %1 block %2 Access to variable not possible
017202*	Channel %1 block %2: cannot delete magazine data
017212*	Channel %1 tool management: insert manual tool %3, duplo no. %2 in spindle/tool holder %4
017214*	Channel %1 tool management: remove manual tool %3 from spindle/tool holder %2
017216*	Channel %1 tool management: remove manual tool from spindle/tool holder %4 and insert manual tool %3, duplo no. %2
017220	Channel %1 block %2 Tool not available
017230*	Channel %1 block %2 Duplo number already allocated
017240*	Channel %1 block %2 Invalid definition of tool
017250*	Channel %1 block %2 Illegal definition of magazine
017260*	Channel %1 block %2 Illegal definition of magazine location
017262*	Channel %1 block %2 invalid operation on a tool adapter
017270	Channel %1 block %2 call-by-reference: Illegal variable
017500*	Channel %1 block %2 axis %3 is not an indexing axis
017501*	Channel %1 block %2 Indexing axis %3 with Hirth tooth system is active
017502*	Channel %1 block %2 Indexing axis %3 with Hirth tooth system Stop delayed
017503*	Channel %1 block %2 Indexing axis %3 with Hirth tooth system and axis not referenced
017510*	Channel %1 block %2 Invalid index for indexing axis %3
017600*	Channel %1 block %2 Preset on transformed axis %3 not possible
017605*	Channel %1 block %2 axis %3 trafo activ: prohibits rotation of axcontainer

017610*	Channel %1 block %2 Positioning axis %3 cannot participate in transformation
017620	Channel %1 block %2 Fixpoint cannot be approached for transformed axis %3
017630	Channel %1 block %2 Referencing not possible for transformed axis %3
017640*	Channel %1 block %2 Spindle cannot be used as transformed axis %3
017650	Channel %1 block %2 Machine axis %3 not programmable
017800*	Channel %1 block %2 Illegal fixed-stop end point programmed
017900	Channel %1 block %2 axis %3 is not a machine axis
018000	Channel %1 block %2 Error in protection zone %3 error no. %4
018001	Channel %1 block %2 Wrong definition of channel-specific protection area %3. Error code %4
018002	Channel %1 block %2 protection zone %3 cannot be activated. error code %4
018003	Channel %1 block %2 Channel-specific protection area %3 cannot be activated. Error code %4
018004	Channel %1 block %2 unequal orientation between workpiece-related %3 and tool-related %4 protection zones
018005	Channel %1 block %2 Serious error in definition of protection zone %3
018006	Channel %1 block %2 Serious error in definition of channel-specific protection zone %3
018100	Channel %1 block %2 Invalid argument passed to FXS[]
018101	Channel %1 block %2 Invalid argument passed to FXST[]
018102	Channel %1 block %2 Invalid argument passed to FXSW[]
018200	Channel %1 block %2 Curve table: Block search stop not allowed with definition CTABDEF
018201	Channel %1 block %2 Curve table: Table %3 does not exist
018202	Channel %1 block %2 Curve table: Instruction CTABEND illegal without CTABDEF
018300	Channel %1 block %2 Frame: Fine shift not possible
018310	Channel %1 block %2 Frame: Illegal rotation
018311	Channel %1 block %2 Frame: Illegal instruction
018312	Channel %1 block %2 Frame: Fine shift not configured
018313	Channel %1 block %2 Frame: Geometry axis changeover not allowed
018314	Channel %1 block %2 Frame: Type conflict
018400	Channel %1 block %2 Language switchover not possible:%3
020000	Channel %1 axis %2 Reference cam not reached
020001	Channel %1 axis %2 Cam signal missing
020002	Channel %1 axis %2 Zero reference mark not found
020003	Channel %1 axis %2 Encoder error
020004	Channel %1 axis %2 Reference mark missing
020005	Channel %1 axis %2 Reference point approach aborted
020006	Channel %1 axis %2 Reference point creep velocity not reached
020007*	Channel %1 axis %2 Reference point approach needs 2 encoders
020008*	Channel %1 axis %2 Reference point approach needs second referenced encoder
020050*	Channel %1 axis %2 Handwheel mode active
020051*	Channel %1 axis %2 Handwheel mode not possible
020052	Channel %1 axis %2 already active
020053*	Channel %1 axis %2 DRF, FTOCON, external setting of offset not possible
020054*	Channel %1 axis %2 Wrong index for indexing axis in JOG mode
020055*	Channel %1 Master spindle not available in JOG mode
020056*	Channel %1 axis %2 No revolutional feedrate possible. Axis/spindle %3 stationary
020057*	Channel %1 block %3 Revolutional feedrate for axis/spindle %2 is ≤ zero.
020058*	Channel %1 axis %2 revolutional feedrate: illegal feedsources
020060	Channel %1 axis %2 cannot move as geometry axis
020061	Channel %1 axis %2 cannot move as orientation axis

020062	Channel %1 axis %2 already active
020063	Channel %1 axis %2 cannot move orientation axes without transformation
020065*	Channel %1 Master spindle not defined for geometry axes in JOG mode
020070	Channel %1 axis %2 Programmed end position is beyond software limit switch %3
020071	Channel %1 axis %2 Programmed end position is beyond working area limit %3
020072*	Channel %1 axis %2 is not an indexing axis
020073	Channel %1 axis %2 cannot be repositioned
020074*	Channel %1 axis %2 Wrong index position
020075	Channel %1 axis %2 Oscillation currently not possible
020076	Channel %1 axis %2 Change of operation mode not possible during oscillation
020077	Channel %1 axis %2 Programmed position is beyond software limit switch %3
020078	Channel %1 axis %2 Programmed position is beyond working area limit %3
020079	Channel %1 axis %2 Oscillating path %3 ≤ 0
020080*	Channel %1 axis %2 Handwheel not assigned for overlaid handwheel motion
020085*	Channel %1 Contour handwheel: Traverse direction or overtravel not allowed from beginning of block
020090	Axis %1 Activation of fixed stop not possible
020091	Axis %1 has not reached fixed stop
020092	Axis %1 Fixed stop mode still active
020093	Axis %1 Standstill monitoring at fixed-stop end point has triggered
020094	Axis %1 Fixed-stop mode has been aborted
020095*	Achse %1 Illegal holding torque, measured torque %2
020096*	Achse %1 Brake test canceled, additional info %2
020100*	Channel %1: Invalid configuration for digitizing
020101*	Timeout during initialization of communication with digitizer
020102*	Channel %1: Illegal or no transformation active for digitizing
020103*	Channel %1: Digitizing module does not support 3+2-axis digitization
020105*	Channel %1: Axis stopped by digitizer. Error code: %2
020106*	Emergency stop set by digitizer
020108*	Invalid data packet received from digitizer. Error codes: %1, %2
020109*	Error in communication with the digitizer: Status code of com-circuit: %1
020120*	Axis %1: Too many relations defined for cross error compensation
020121*	Axis %1: Configuration error in cross error compensation table %2
020122*	Invalid axis assignment for cross error compensation table %1
020123*	Axis %1: Assignment of different output axes in cross error compensation tables to be multiplied
020124*	Axis %1: Sum of compensation values has been limited
020125*	Axis %1: Variation of compensation value is too rapid
020130*	Channel %1 Contour tunnel monitoring
020140	Channel %1 Motion synchronous action: Positioning axis cannot be traversed from synchronous action %2, error cause %3
020141	Channel %1 Motion synchronous action: Illegal axis type
020142*	Channel %1 command axis : rotation of axescontainer already allowed
020143	Channel %1 Axis %2 cannot be started, CPU axis
020144	Channel %1 block %2 Motion synchronous action: Access to system variable not possible.
020145	Channel %1 block %2 Motion synchronous action: Arithmetic error
020146	Channel %1 block %2 Motion synchronous action: Nesting depth exceeded
020147	Channel %1 block %2 Motion synchronous action: Command not executable
020148	Channel %1 block %2 Motion synchronous action: Internal error %3
020149	Channel %1 block %2 Motion synchronous action: Illegal index
020150*	Channel %1 tool management: PLC terminates interrupted command

020160*	Channel %1 tool management: PLC can terminate only incorrectly aborted commands
020170	Channel %1 FIFO has exceeded the R parameter range
020200*	Channel %1 Invalid spindle no. %2 with fine compensation
020201*	Channel %1 Spindle %2 No tool assigned
020203*	Channel %1 No tool selected
020204*	Channel %1 Instruction PUTFTOC not allowed during FTOCOF
020210*	Channel %1 block %3 spindle %2 Wrong values for centerless grinding
020211*	Channel %1 block %3 spindle %2 Support point beyond limits
021500	Channel %1 block %2 axis %3 controlling not permitted: %4
021600*	Monitoring for ESR active
021610	Channel %1 axis %2 Encoder frequency limit exceeded
021611*	Channel %1 NC-controlled, extended stop and retract triggered
021612	Channel %1 axis %2 servo enable reset during traverse motion
021613*	Axis %1 switches active encoder
021614	Channel %1 axis %2 Hardware limit switch %3 reached
021615	Channel %1 axis %2 taken from traverse mode to follow-up mode
021616	Channel %1 block %2 Overlaid motion active at transformation switchover
021617	Channel %1 block %2 Transformation prohibits traversal of the pole.
021618	Channel %1, starting at block %2 Transformation active: Overlaid motion too great
021619	Channel %1 block %2 Transformation active: Motion not possible
021650	Channel %1 axis %2 Overlaid motion not allowed
021660*	Channel %1 block %2 axis %3 Conflict between SYNACT:\$AA_OFF and CORROF
021665*	Channel %1 \$AA_TOFF reset
021670*	Channel %1 block %2 Illegal change of tool direction since \$AA_TOFF active
021700	Channel %1 block %3 axis %2 Touch probe already deflected, edge polarity not possible
021701	Channel %1 block %3 axis %2 Measurement not possible
021702	Channel %1 block %3 axis %2 Measurement aborted
021703	Channel %1 block %3 axis %2 Touch probe not deflected, edge polarity not possible
021740*	Output value at analog output no. %1 has been limited
021750	Error during timer-controlled cam output
021760	Channel %1 block %2 Too many auxiliary functions programmed
021800	Channel %1 Required number of workpieces = %2 reached
022000*	Channel %1 block %3 spindle %2 Change of gear stage not possible
022010*	Channel %1 block %3 spindle %2 Actual gear stage differs from requested gear stage.
022011*	Channel %1 block %3 spindle %2 Cannot change to programmed gear stage
022012*	Channel %1 block %2 master spindle %3 is in simulation mode
022013*	Channel %1 block %2 master spindle %3 is in simulation mode
022014*	Channel %1 block %2. The dynamic properties of master spindle %3 and following spindle %4 significantly differ.
022020*	Channel %1 block %3 spindle %2 Change of gear stage position not reached
022040*	Channel %1 block %3 spindle %2 is not referenced with zero marker.
022045*	Block %2 spindle/axis %3 not available in channel %1 because it is active in channel %4
022050*	Channel %1 block %3 spindle %2 Transition from speed control mode to position control mode not possible
022051*	Channel %1 block %3 spindle %2 Reference mark not found
022052*	Channel %1 block %3 spindle %2 Zero speed on block change not reached
022053*	Channel %1 block %3 spindle %2 Reference mode not supported
022054*	Channel %1 block %3 spindle %2 Improper punch signal

022055*	Channel %1 block %3 spindle %2 Configured positioning speed is too high.
022060	Channel %1 Position control expected for axis spindle %2
022062	Channel %1 axis %2 Reference point approach: Search speed for zero mark (MD) is not reached.
022064	Channel %1 axis %2 Reference point approach: Search speed for zero mark (MD) is too high
022065*	Channel %1 Tool management: Tool move not possible since there is no tool %2 with Duplo no. %3 in magazine %4
022066*	Channel %1 Tool management: Tool not changed since there is no tool %2 with Duplo no. %3 in magazine %4
022067*	Channel %1 Tool management: Tool not changed because no tool available in tool group %2
022068*	Channel %1 block %2 Tool management: No tool available in tool group %3
022069	Channel %1 block %2 Tool management: No tool available in tool group %3, program %4
022070	TO unit %1 Please load tool T=%2 into magazine. Repeat data backup.
022071	TO unit %1 tool %2 Duplo no. %3 is active but not in active wear grouping
022100*	Channel %1 block %3 spindle %2 Chuck speed exceeded
022150*	Channel %1 block %3 spindle %2 Maximum speed for position control exceeded
022200*	Channel %1 spindle %2 Axis stopped during tapping
022250*	Channel %1 spindle %2 Axis stopped during thread cutting
022260*	Channel %1 spindle %2 Thread might be damaged
022270*	Channel %1 block %2 Maximum velocity of thread axis reached at position %3
022275	Channel %1 block %2 Zero velocity of thread axis reached at position %3
022280	Channel %1 programmed ramp-up distance in block %2: too short %3, required %4
022320*	Channel %1 block %2 PUTFTOCF data block could not be transferred
022321*	Channel %1 axis %2 PRESET not allowed during traverse motion
022322*	Channel %1 axis %2 PRESET: Invalid value
025000	Axis %1 Hardware fault of active encoder
025001	Axis %1 Hardware fault of passive encoder
025010	Axis %1 Pollution of active encoder
025011	Axis %1 Pollution of passive encoder
025020	Axis %1 zero mark monitoring of active encoder
025021	Axis %1 zero mark monitoring of passive encoder
025022	Axis %1 Encoder %2 Warnng %3
025030	Axis %1 Actual velocity alarm
025031	Axis %1 Actual velocity warning
025040	Axis %1 Standstill monitoring
025042*	Axis %1 standstill monitoring during torque or force reduction
025050	Axis %1 Contour tolerance monitoring
025060	Axis %1 Desired speed limit
025070	Axis %1 Drift limit exceeded
025080	Axis %1 Positioning monitoring
025100*	Axis %1 Switchover of encoder not possible
025105*	Axis %1 Encoder positions tolerance exceeded
025110	Axis %1 Selected encoder not available
025120	Axis %1 error detection %3 active encoder %2
025190	Channel %1 axis %2 position control in controlling
025200	Axis %1 Requested set of parameters invalid
025201	Axis %1 Drive fault/Error rotation monitoring (stepper motor)
025202	Axis %1 Waiting for drive
026000	Axis %1 Clamping monitoring
026001*	Axis %1 Invalid output rating friction compensation

026002	Axis %1 linear encoder %2 Invalid grid Encoder marks
026003*	Axis %1 Invalid output rating pitch
026004	Axis %1 linear encoder %2 Invalid grid point distance
026005	Axis %1 Invalid output rating configured
026006	Axis %1 encoder %2 Encoder type/output type %3 not possible
026007*	Axis %1 Feedforward control step size for quadrant error compensation
026008*	Axis %1 Feedforward control for quadrant error compensation
026009*	Axis %1 Feedforward control for quadrant error compensation
026010*	Axis %1 Feedforward control characteristic for quadrant error compensation
026011*	Axis %1 Feedforward control for quadrant error compensation
026012*	Axis %1 Feedforward control not active for quadrant error compensation
026014	Axis %1 Machine data %2 invalid value
026015	Axis %1 Machine data %2 [%3] invalid value
026016	Axis %1 Machine data %2 invalid value
026017	Axis %1 Machine data %2 [%3] invalid value
026018	Axis %1 Control output branch drive %2 multiply used
026019	Axis %1 encoder %2 Measurement not possible with this controller module
026020	Axis %1 encoder %2 Hardware fault %3 during encoder %2 initialization
026022	Axis %1 encoder %2 Measurement not supported, encoder simulation
026024	Axis %1 machine data %2 Value adapted
026025	Axis %1 machine data %2 [%3] Value adapted
026030	Axis %1 encoder %2 Absolute position lost
026031	Axis %1 Configuration error master/slave
026032	Axis %1 Master/slave not configured
026050	Axis %1 Parameter block change from %2 to %3 not possible.
026051	Channel %1 Unanticipated stop in block %2 crossed in continuous-path mode
026052	Channel %1 in block %2: Drop in velocity in continuous-path mode
026070	Channel %1 axis %2 Max. number of independent CPU axes exceeded
026072	Channel %1 Axis %2 cannot be traversed as an independent CPU axis
026074	Channel %1 Axis %2 is independent CPU axis
026080	Channel %1 Retraction position of axis %2 not programmed or invalid
026081	Channel %1 Axial trigger for axis %2 was triggered, but axis is not PLC controlled
026100	Axis %1 drive %2 Sign-of-life failure
026100	Axis %1, Drive %2 Sign of life of drive %2 missing
026101	Axis %1, Drive %2 communication failure
026102	Axis %1, Drive %2 Sign of life of drive %2 missing
026105	Drive of Axis %1 not found
026106	Encoder %2 of Axis %1 not found
026110*	Extended stopping/retract on digital drive activated
060000*	Channel %1 block %2
061000*	Channel %1 block %2
062000*	Channel %1 block %2
063000*	Channel %1 block %2
065000*	Channel %1 block %2
066000*	Channel %1 block %2
067000*	Channel %1 block %2
068000*	Channel %1 block %2
070000*	Compile cycle alarm
075000*	OEM alarm
075200	Channel %1 Incorrect MD configuration, %2 invalid
075210	Channel %1 Number of axes / axis assignment inconsistent
075250	Channel %1 Tool parameter invalid

075255	Channel %1 Working area error
075260	Channel %1 block %2 Tool parameter invalid
075265	Channel %1 block %2 Working area error
075270	Channel %1 Tool parameter invalid
075275	Channel %1 block %2 Working area error
028000*	NCU-Link-Connection to all other NCUs is terminated
028001*	NCU-Link-Connection to the NCUs %1 is terminated
028002*	Error durring machine data refresh, NCU-Cluster global machine data was changed from NCU %1
028004*	NCU-Link: NCU %1 is not connected
028005*	NCU-Link: NCU %1 is not synconized
028007*	NCU-Link: project data error between local NCU and NCU %1
028008*	NCU-Link: timer data error between local NCU and NCU %1
028009*	NCU-Link: bus parameter error between local NCU and NCU %1
028010*	NCU-Link: the NCU %1 did not receive a link message
028011*	NCU-Link exceeds the interpolation cycle time with %1 us
028012*	NCU-Link: syncorisation signal fail to appear %1 times
028020*	NCU-Link: Number of Linkaxis exceeded. %1
028030*	Severe Alarm on NCU % 1, axes in followup
028031*	Severe Alarm on NCU % 1 not cleared, axes in followup
028032*	Emergency stop on NCU % 1 set, axes in followup
028033*	Emergency stop on NCU % 1, axes in followup
029033	Channel %1 axchange from axes %2 not possible, plc-ax movement not finished
300000*	Hardware not found: DCM drive bus ASIC
300001*	Axis %1 drive no. %2 not allowed
300002*	Axis %1 drive no. %2 assigned twice
300003*	Axis %1 drive %2 Module type found differs from configured type %3
300004*	Axis %1 drive %2 Drive type found (FDD/MSD) differs from configured type %3
300005*	At least one module %1 too many on drive bus
300006*	At least one configured module (module/drive %1) has not been found on drive bus
300007*	Axis %1 drive %2 not configured or inactive
300008*	Axis %1 drive %2 encoder %3 is not available
300009*	Axis %1 drive %2 encoder %3 Configured encoder type differs from type found (%4)
300010*	Axis %1 drive %2 active without NC axis assignment
300011*	Axis %1 drive %2 Hardware version of spindle not supported
300012*	Axis %1 drive %2 Hardware version of control module not supported
300020*	Drive %1 removed for diagnostics
300100*	Drive power failure
300101*	Drive power missing
300200*	Drive bus hardware fault
300201*	Axis %1 drive %2 Timeout during bus access, error code %3
300202*	Axis %1 drive %2 CRC error, error code %3
300300*	Axis %1 drive %2 Boot error, error code %3
300400*	Axis %1 drive %2 System error, error codes %3, %4
300401*	Software for drive type %1, block %2 missing or defective
300402*	System error in drive interface, error codes %1, %2
300403*	Axis %1 drive %2 Version number unmatched of drive software and machine data
300404*	Axis %1 drive %2 Machine data file contains unmatched drive no.
300405*	Axis %1 drive %2 Unknown drive alarm, code %3
300410*	Axis %1 drive %2 Data file could not be stored (%3, %4)
300411*	Axis %1 drive %2 Data file could not be read (%3, %4)
300412	Data file could not be stored (%1, %2)
300413	Data file could not be read (%1, %2)

300423	Trace results could not be read (%1)
300500*	Axis %1 drive %2 system error, error codes %3, %4
300501*	Axis %1 drive %2 error in measuring circuit of absolute current
300502*	Axis %1 drive %2 error in phase current R of motor current transducer
300503*	Axis %1 drive %2 error in phase current S of motor current transducer
300504*	Axis %1 drive %2 fault of motor transducer
300505*	Axis %1 drive %2 optical encoder of motor measuring system failed, Code %3
300506*	Axis %1 drive %2 sign of life of NC missing
300507*	Axis %1 drive %2 synchronization error of rotor position
300508*	Axis %1 drive %2 zero mark monitoring of motor measuring system
300509*	Axis %1 drive %2 converter cut-off frequency exceeded
300510*	Axis %1 drive %2 error in mid-frequency measurement
300511*	Axis %1 drive %2 recording of measured values active during boot
300515*	Axis %1 drive %2 heat sink temperature exceeded
300604*	Axis %1, drive %2 motor encoder not adjusted
300605*	Axis %1, drive %2 motor switching not permitted
300606*	Axis %1 drive %2 flux control output limited
300607*	Axis %1 drive %2 current control output limited
300608*	Axis %1 drive %2 speed control output limited
300609*	Axis %1 drive %2 encoder cut-off frequency exceeded
300610*	Axis %1 drive %2 rotorposition identification failed
300611*	Axis %1 drive %2 illegal motion during rotorposition identification
300612*	Axis %1 drive %2 illegal current during rotorposition identification
300613*	Axis %1 drive %2 maximum permissible motor temperature exceeded
300614*	Axis %1 drive %2 motor temperature exceeded
300701*	Axis %1 drive %2 needs setup
300702*	Axis %1 drive %2 base cycle time invalid
300703*	Axis %1 drive %2 cycle time of current control invalid
300704*	Axis %1 drive %2 cycle time of speed control invalid
300705*	Axis %1 drive %2 cycle time of position control invalid
300706*	Axis %1 drive %2 cycle time for monitoring invalid
300707*	Axis %1 drive %2 basic cycle times of axes differ
300708*	Axis %1 drive %2 current control cycle times of axes differ
300709*	Axis %1 drive %2 speed control cycle times of axes differ
300710*	Axis %1 drive %2 position control cycle times of axes differ
300711*	Axis %1 drive %2 monitoring cycle times of axes differ
300712*	Axis %1 drive %2 configuration of controller structure not possible
300713*	Axis %1 drive %2 lead time for position control invalid
300714*	Axis %1 drive %2 ID-code of power section invalid
300715*	Axis %1 drive %2 maximum current of power section invalid
300716*	Axis %1 drive %2 constant of torque invalid
300717*	Axis %1 drive %2 motor moment of inertia invalid
300718*	Axis %1 drive %2 calculation dead time of current control invalid
300719*	Axis %1 drive %2 motor not configured for delta connection
300720*	Axis %1 drive %2 maximum motor speed invalid
300721*	Axis %1 drive %2 zero-load current exceeds rated current of motor
300722*	Axis %1 drive %2 zero-load current exceeds rated current of power section
300723*	Axis %1 drive %2 STS configuration of axes differ
300724*	Axis %1 drive %2 number of pole pairs invalid
300725*	Axis %1 drive %2 encoder resolution invalid
300726*	Axis %1 drive %2 voltage constant invalid
300727*	Axis %1 drive %2 reactance invalid
300728*	Axis %1 drive %2 adaption factor torque/current too high

300729*	Axis %1 drive %2 zero-speed current of motor invalid
300730*	Axis %1 drive %2 rotor resistance invalid
300731*	Axis %1 drive %2 rated power invalid
300732*	Axis %1 drive %2 rated speed invalid
300733*	Axis %1 drive %2 zero-load voltage invalid
300734*	Axis %1 drive %2 zero-load current invalid
300735*	Axis %1 drive %2 field weakening speed invalid
300736*	Axis %1 drive %2 Lh characteristic invalid
300737*	Axis %1 drive %2 configuration of two EnDat encoders not possible
300738*	Axis %1 drive %2 encoder module number not possible
300739*	Axis %1 drive %2 encoder already used as motor measuring system
300740*	Axis %1 drive %2 encoder multiple used
300741*	Axis %1 drive %2 asynchronous mode: gain of feedforward control out of range
300742*	Axis %1 drive %2 voltage/frequency mode: converter frequency invalid value
300743*	Axis %1 drive %2 function not supported on this 611D module
300750*	Axis %1 drive %2 configuration of speed control adaption invalid
300751*	Axis %1 drive %2 gain of speed control too high
300752*	Axis %1 drive %2 blocking frequency of setpoint current filter invalid
300753*	Axis %1 drive %2 rotor position identification current less than minimum value
300754*	Axis %1 drive %2 signal number invalid
300755*	Axis %1 drive %2 voltage/frequency mode: motor is already turning
300756*	Axis %1 drive %2 speed hysteresis of setpoint current smoothing invalid
300757*	Axis %1 drive %2 adaption factor of torque limit invalid
300758*	Axis %1 drive %2 generator mode: response voltage exceeds switch-off threshold
300759*	Axis %1 drive %2 generator mode: response voltage exceeds monitoring threshold
300760*	Axis %1 drive %2 generator mode: emergency-retraction-speed exceeds maximum motor speed
300761*	Axis %1 drive %2 generator mode: minimum axis speed exceeds maximum motor speed
300762*	Axis %1 drive %2 emergency retraction mode / generator mode already active
300763*	Axis %1 drive %2 emergency retraction mode / generator mode invalid
300764*	Axis %1 drive %2 emergency retraction mode / generator mode not possible
300765*	Axis %1 drive %2 measurement of d.c. link voltage not possible
300766*	Axis %1 drive %2 blocking frequency greater Shannon frequency
300767*	Axis %1 drive %2 natural frequency > Shannon frequency
300768*	Axis %1 drive %2 bandwidth numerator > 2. blocking frequency
300769*	Axis %1 drive %2 bandwidth denominator > 2. natural frequency
300770v	Axis %1 drive %2 format error
300771*	Axis %1 drive %2 asynchronous mode: converter frequency invalid value
300772*	Axis %1 drive %2 asynchronous mode: gain of speed control too high
300773*	Axis %1 drive %2 asynchronous mode: configuration of feedforward control structure not possible
300774*	Axis %1 drive %2 asynchronous mode: changeover speed invalid value
300775*	Axis %1 drive %2 fixed link voltage of axes differ
300776*	Axis %1 drive %2 encoder signal monitoring must be active
300777*	Axis %1 drive %2 rotorposition identification current invalid value
300778*	Axis %1 drive %2 rotorposition identification converter frequency invalid value
300779*	Axis %1 drive %2 motor moment of inertia invalid
300780*	Axis %1 drive %2 zero-load current exceeds rated current of motor
300781*	Axis %1 drive %2 zero-load current exceeds rated current of power section
300782*	Axis %1 drive %2 reactance invalid
300783*	Axis %1 drive %2 rotor resistance invalid
300784*	Axis %1 drive %2 zero-load voltage invalid

300785*	Axis %1 drive %2 zero-load current invalid
300786*	Axis %1 drive %2 field weakening speed invalid
300787*	Axis %1 drive %2 asynchronous mode: gain of feedforward control out of range
300788*	Axis %1 drive %2 configuration of current control adaption invalid
300789*	Axis %1 drive %2 function not possible with this 611D closed-loop control module
300799*	Axis %1 drive %2 data backup and reboot required
300850*	Axis %1 drive %2 configuration of speed control adaption invalid
300854*	Axis %1 drive %2 signal number invalid
300855*	Axis %1 drive %2 voltage/frequency mode: motor is already turning
300858*	Axis %1 drive %2 generator mode: response voltage exceeds switch-off threshold
300859*	Axis %1 drive %2 generator mode: response voltage exceeds monitoring threshold
300860*	Axis %1 drive %2 generator mode: emergency-retraction-speed exceeds maximum motor speed
300861*	Axis %1 drive %2 generator mode: minimum axis speed exceeds maximum motor speed
300862*	Axis %1 drive %2 emergency retraction mode / generator mode already active
300863*	Axis %1 drive %2 emergency retraction mode / generator mode invalid
300864*	Axis %1 drive %2 emergency retraction mode / generator mode not possible
300865*	Axis %1 drive %2 measurement of d.c. link voltage not possible
300875*	Axis %1 drive %2 fixed link voltage of axes differ
300888*	Axis %1 drive %2 configuration of current control adaption invalid
301702*	Axis %1 drive %2 wrong inversion of actual value
301703*	Axis %1 drive %2 measurement system and category of motor are not compatible
301704*	Axis %1 drive %2 division of linear scale and pole pair pitch not compatible
301705*	Axis %1 drive %2 machine data for scale with position coded reference marks are faulty
301709*	Axis %1 drive %2 submodule with integrated linearization is invalid
301710*	Axis %1 drive %2 invalid resolution of SSI motor measuring system
301711*	Axis %1 drive %2 message frame length SSI motor measuring system invalid
301712*	Axis %1 drive %2 multiturn SSI motor measuring system invalid
301713*	Axis %1 drive %2 resolution SSI direct measuring system invalid
301714*	Axis %1 drive %2 message frame length SSI direct measuring system invalid
301715*	Axis %1 drive %2 multiturn SSI direct measuring system invalid
301716*	Axis %1 drive %2 SSI direct measuring system without incremental tracks not possible
301717*	Axis %1 drive %2 SSI transmission time exceeded
301718*	Axis %1 drive %2 illegal motor/power section combination
301719*	Axis %1 drive %2 incomplete power section data
380001	Profibus DP: Power-up error, cause %1 parameter %2 %3 %4
380003	Profibus DP: Operating error, cause %1 parameter %2 %3 %4
380020	Profibus DP: SDB1000 error %1 for SDB source %2.
380021	Profibus DP: Default SDB1000 was loaded.
380022	Profibus DP: Configuration of DP master has been changed
380040	Profibus DP: Configuration error %1, parameter %2.
380050	Profibus DP: Multiple assignment of inputs at address %1.
380051	Profibus DP: Multiple assignment of outputs at address %1.
380060	Profibus DP: Error %1 at logical address %2 of non-assigned station
380070	Profibus-DP: No input slot for base address %1 (length %2) available.
380071	Profibus-DP: No output slot for base address %1 (length %2) available.
380072	Profibus DP: no outbound slot for baseaddress %1 (size %2) allowed
380075	Profibus-DP: Failure of DP I/Os slave %1
380500	Profibus-DP: Failure of drive %1, code %2, value %3, timer %4.

Action list

In the “Error evaluation” window of the parameterizing tool action numbers are displayed in the error text (e.g. action 7 is not permitted). The following table lists and explains the action numbers.

Text marked with an * affects functions that are not available in the FM 357-2.

Table 12-4 Action list

No./name	Explanation	Not allowed if ...	Remedy
1. INIT	Execute INIT phase (tasks are initialized after Power On)		
2. RESET	Execute Reset (Reset signal or after Power On)		
3. RESET_INITBLOCK	Activating Reset Init blocks (signal: after Reset)		
4. PROG_END	Execute Reset, end of program detected (NC block with M30)		
5. MODESWITCHTOA-PROG MODE	Mode change to program mode MDI or Automatic	<ul style="list-style-type: none"> The channel is active (program running, block search*, load machine data) Another operating mode has already been started. *A channel has quit the mode group because of an interrupt. *Overstore or digitizing is selected. 	<ul style="list-style-type: none"> Abort program with Reset or stop program (not during block search*, load machine data) Abort program with Reset *Abort program with Reset key or wait until interrupt is finished. *Deselect overstore, digitizing
6. *MODESWITCHTO-SAVE-MODE	*Automatic switch-over from an internal mode to the external mode setting (with TEACH_IN, an attempt is made after each stop to switch from the internal mode “AUTOMATIC, MDA” to TEACH_IN)		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
7. MODESWITCHTO-HAND-MODE	Mode change to a setup mode (Signal: Jog, TEACH IN, Reference point approach)	<ul style="list-style-type: none"> Nesting depth exceeded: Various events (such as an interrupt) can cause the current machining operation to be interrupted. ASUB programs are activated according to the event. These ASUB programs can be interrupted like the user program. The memory capacity restricts the nesting depth of ASUB programs. The channel is active (program running, block search*, load machine data) *A channel has quit the mode group because of an interrupt. *Overstore or digitizing is selected. 	<ul style="list-style-type: none"> Abort program with Reset Abort program with Reset or stop program (not during block search*, load machine data) Abort program with Reset or wait until interrupt is finished. *Deselect overstore, digitizing
8. *OVERSTOREON	Select overstore (PI command).		
9. *OVERSTOREOFF	Deselect overstore (PI command).		
10. INTERRUPT	Execute a user interrupt "ASUB" (signal, digital/analog interface*, ASUB interface).	<ul style="list-style-type: none"> The channel is active due to block search* or load machine data The channel is stopped, the ASUB has to be started, and the current block cannot be reorganized. *Digitizing is selected Reference point approach has not yet been performed On a deceleration Reorg error 	<ul style="list-style-type: none"> Wait until block search* or load machine data has finished or abort program with Reset Activate block change until NC block can be reorganized. *Deselect digitizing Perform reference point approach Abort program
11. INTERRUPTFASTLIFT-OFF	Execute a user interrupt "ASUB" with lift-fast (signal, ASUB interface, digital/analog interface*)	see 10.	
12. INTERRUPTBL-SYNC	Execute a user interrupt "ASUB" at end of block (signal, ASUB interface, digital/analog interface*)	see 10.	

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
13. *FASTLIFTOFF	Execute lifftast		
14. *TM_MOVETOOL	Move tool (with tool management only) (PI command)		
15. DELDISTOGO_SYNC	Delete distance to go or synchronize axis. (Signal: delete distance to go or follow-up mode) Follow-up mode: e.g. when activating axis control	<ul style="list-style-type: none"> • Nesting depth exceeded • If deceleration Reorg error 	<ul style="list-style-type: none"> • Abort program • Abort program
16. *PROGRESETREPEAT	Cancel subprogram repetition (VDI signal: Clear subprogram number of passes)	<ul style="list-style-type: none"> • Nesting depth exceeded • If deceleration Reorg error 	<ul style="list-style-type: none"> • Abort program • Abort program
17. *PROGCANCELSUB	Cancel subprogram processing. (VDI signal: cancel program level)	<ul style="list-style-type: none"> • Nesting depth exceeded • If deceleration Reorg error 	<ul style="list-style-type: none"> • Abort program • Abort program
18. SINGLEBLOCKSTOP	Activate single-block (Signal: Activate single block)		
19. SINGLEBLOCKOFF	Deactivate single-block (Signal: Activate single block)		
20. *SINGLEBLOCK_IPO	Activate main run single-block. (OPI variable and VDI signal: Activate single block)		
21. *SINGLEBLOCK_DECODIER	Activate decode single block. (OPI variable and VDI signal: Activate single block)	<ul style="list-style-type: none"> • Nesting depth exceeded • If deceleration Reorg error 	<ul style="list-style-type: none"> • Wait until preceding ASUB has finished or abort program • Abort program
22. *SINGLEBLOCK_MAINBLOCK	Activate main program single-block. (OPI variable and VDI signal: Activate single block)		
23. *SINGLEBLOCK_PATH	Activate traverse single block. (OPI variable and VDI signal: Activate single block)		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
24. STARTPROG	Start program processing. (Signal: Start)	<ul style="list-style-type: none"> • Program status active • An alarm reaction is active which inhibits a start or forces a deceleration. • Reference point approach not yet performed 	<ul style="list-style-type: none"> • Execute alarm cancelation condition • Reference point approach
25. *CHANNELSTART-PROG	Start program processing. (Channel communication, NC block: Start)	<ul style="list-style-type: none"> • Program status active • An alarm reaction is active which inhibits a start or forces a deceleration. • Reference point approach not yet performed • An incorrect mode is selected. (only Automatic) 	<ul style="list-style-type: none"> • Secure start with WAITE • Execute alarm cancelation condition • Reference point approach • Select program mode
26. RESUMEPROG	Resume program processing. (Signal: Start)	<ul style="list-style-type: none"> • Program status active • An alarm reaction is active which inhibits a start or forces a deceleration. • Reference point approach has not yet been performed 	<ul style="list-style-type: none"> • Execute alarm cancelation condition • Reference point approach
27. RESUMEJOGREFDIGIT	Resume selected processing mode (jog, reference point or digitizing*). (Signal: Start)	<ul style="list-style-type: none"> • JOG motion active • An alarm reaction is active which inhibits a start or forces a deceleration. 	<ul style="list-style-type: none"> • Execute alarm cancelation condition
28. *STARTDIGITIZE	Start processing in digitizing submode. (VDI signal: NC Start)	<ul style="list-style-type: none"> • JOG motion is active • An alarm reaction is active which inhibits a start or forces a deceleration. • Reference point approach has not yet been performed 	<ul style="list-style-type: none"> • Execute alarm cancelation condition • Reference point approach
29. STOPALL	Stop all axes. (Signal: Stop All or Reset)		
30. STOPPROG	Execute a program stop. (NC block: M0)		
31. STOPJOGREF	Stop motion in "Jog". (Signal: Stop)		
32. *STOPDIGITIZE	Stop digitizing. (VDI signal: NC Stop)		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
33. STARTSIG	Start selected processing. (Signal: Start)	<ul style="list-style-type: none"> • *Process switch active (mode change, activate/deactivate digitizing, activate/deactivate overstore) • An error reaction is active which inhibits a start or forces a deceleration. • A process is running (NC program, block search*, load machine data) 	<ul style="list-style-type: none"> • Execute alarm cancelation condition
34. STOPSIG	Stop active processing. (Signal: Stop)		
35. *INITIALINISTART	Start machine data processing (INI file already in NCK), (PI command)		
36. *INITIALINIEXT-START	Start machine data processing (INI file already available externally, e.g. on MMC), (PI command)		
37. *BAGSTOP_SLBTYPA	Stop because of mode group single-block. (VDI signal, individual type A, after Stop in another channel of the same mode group)		
38. *BAGSTOPATEND_SLBTYPB	Stop because of mode group single-block. (VDI signal, individual type B, after stop at end of block in another channel of the same mode group)		
39. *OVERSTORE_BUFFER_END_REACHED	Stop because end of overstore buffer "_N_OSTOREXX_SYF" has been reached.		
40. PREP_STOP	Start preprocessing (NC block, STOPRE)		
41. PROG_STOP	Stop processing at block boundary. (NC block, M00/M01)		
42. STOPPROGA-BLOCKEND	Stop processing at block boundary. (Error, signal: Stop at block boundary)		
43. STOPPROGATASUP- END	Stop at end of ASUB if started from "stopped" state.		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
44. PROGSELECT	Select program. (PI command)		
45. *PROGSELECTEXT	Select program which is still stored externally. (PI command)		
46. *CHANNEL_PROGSELECT	Select program from other channel. (Channel communication, NC block: INIT)		
47. *ASUPDEFINITION	Save definition of an activatable ASUB. (PI command)		
48. NEWCONF	Set all machine data with attribute (NEW_CONF) to active. (PI command)		
49. CLEARCANCELA-LARM	Clear all alarms with clear condition CANCELCLEAR (PI command, acknowledge alarm key)		
50. *BLOCKSEARCHUN_CONTINUE	Continue block search. (NC block: STOPRE)		
51. *BLOCKSEARCHRUN_START	Start block search. (PI command)		
52. *BLOCKSEARCHRUN_RESUME	Resume block search. (PI command)		
53. *DIGITIZEON	Activate digitizing. (PI command)		
54. *DIGITIZEOFF	Deactivate digitizing. (PI command)		
55. *FUNCTGENON	Activate function generator. (PI command)		
56. *FUNCTGENOFF	Deactivate function generator. (PI command)		
57. WAITM	Wait for program marker. (Channel communication, NC block: WAITM)		
58. WAITE	Wait for end of program. (Channel communication, NC block: WAITE)		
59. INIT_SYNC	Select program from another channel with synchronization. (Channel communication, NC block: INIT+SYNC)		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
60. *MMCCMD	Wait for acknowledgement from MMC. (NC block, MMC_CMD)		
61. PROGMODESLASHON	Activate skip block. (Signal: Skip block)	Nesting depth exceeded	Wait until preceding ASUB has finished or abort program
62. PROGMODESLASHOFF	Deactivate skip block. (Signal: Skip block)	Nesting depth exceeded	Wait until preceding ASUB has finished or abort program
63. *PROGMODEDRY-RUNON	Activate test run. (VDI signal: Rapid traverse override)	<ul style="list-style-type: none"> Nesting depth exceeded If deceleration Reorg error 	<ul style="list-style-type: none"> Wait until preceding ASUB has finished or abort program Abort program
64. *PROGMODEDRY-RUNOFF	Deactivate test run (VDI signal: Rapid traverse override)	<ul style="list-style-type: none"> Nesting depth exceeded If deceleration Reorg error 	<ul style="list-style-type: none"> Wait until preceding ASUB has finished or abort program Abort program
65. *BLOCKREADINHIBIT_ON	Activate read-in disable for main run block. (VDI signal: Read-in disable)		
66. *BLOCKREADINHIBIT_OFF	Deactivate read-in disable for main run block. (VDI signal: Read-in disable)		
67. STOPATEND_ALARM	Stop at end of block. (error)		
68. STOP_ALARM	Stop all axes. (error)		
69. *PROGESTON	Activate program test. (VDI signal: Program test)	<ul style="list-style-type: none"> Tool management is active. The NCK channel state is not Ready 	<ul style="list-style-type: none"> Save tool data Abort program or process with Reset or wait for end of program
70. *PROGTESTOFF	Deactivate program test. (VDI signal: Program test)	The NCK channel state is not Ready	Abort program or process with Reset or wait for end of program
71. *STOPATIPOBUFFER_IEMPTY_ALARM	Stop at end of block preparation. (alarm)		
72. *STOPATIPOBUFFER_EMPTY_ALARM_REORG	Stop at end of block preparation with subsequent block reorganization. (alarm)	Nesting depth exceeded	Wait until preceding ASUB has finished or abort program

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
73. CON- DITIONAL_STOPA- TEND	Conditional stop at end of block. (If a "Stop at end of block" stop reason is still active after continuation with Start, processing stops again.)		
74. CONDITIONAL_ SBL_DEC_STOPA- TEND	Conditional stop at end of block. (The interpreter does not load a block into the main run in spite of Start)		
75. INTERPRETERS- TOP_ALARM	Stop preprocessing. (error)		
76. *RETREAT_MOVE_ THREAD	Retraction movement on G33 and Stop.		
77. WAITMC	Conditional wait for program marker (NC block: WAITMC)		
78. SETM	Set marker. (NC block: SETM)		
79. CLEARM	Clear marker (NC block: CLEARM)		
80. *BLOCK_SELECT	Select an NC block. (PI command)		
81. *LOCK_FOR_EDIT	Disable editing of the NC program currently being executed. (PI command)		
82. *START_TEACHIN- PROG	Start a program in TEACH_IN submode. (VDI signal: NC Start)	see 33. and 5.	
83. *RESUME_ TEACHINPROG	Resume a program in TEACH_IN submode. (VDI signal: NC Start)	see 33. and 5.	
84. PURE_REORG	Reorganize block execution.		
85. INTERRUPT_TO- PROG_NOREPOS	Activate a user interrupt "ASUB" in a setup mode. (signal, ASUB interface, digital/analog interface*)	see 10.	

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
86. INTERRUPT_START	Activate a user interrupt "ASUB". Is executed only in READY channel state. (signal, ASUB, digital/analog interface*)	see 10.	
87. INTERRUPT_SIGNAL	Execute a user interrupt "ASUB". (signal, ASUB, digital/analog interface*) relevant for all interrupt signals. This event determines which actual interrupt is triggered. Possible options are: 10, 11, 12, 85, 86.	see 10.	
88. STOPBAG	Stop processing. (Signal: Stop)		
89. NEWCONF_PREP_STOP	Set all machine data with attribute (NEW_CONF) to active. (NC block: NEW_CONF)		
90. *BLOCKSEARCH-RUN_NEWCONF	Set all machine data with attribute (NEW_CONF) to active. (NC block: NEW_CONF during block search)		
91. CONTINUE_INTERPR	BSALARMEVENT-PAR_CONTINUE_INTERPR Start continuation of interpreter processing. (internal preprocessor stop)		
92. *SLAVEDATA	Interlock for saving data	The NC channel is not stopped.	
93. *SET_USER_DATA	Activate user data i.e., for example, tool lengths modified on the MMC are activated immediately in the running program.	<ul style="list-style-type: none"> The NC channel is not stopped. The channel is stopped and the current block cannot be reorganized. 	<ul style="list-style-type: none"> Press Stop key/Single-block/Reset/StopAtEnd key (in Auto) Activate block change until NC block can be reorganized.
94. *PLCVERSION	Write user PLC version to version file		
95. *CONVERT_SCALING_SYSTEM	BSALARMEVENT-PAR_CONVERT_SCALING_SYSTEM Convert system of measurement PI service		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
96. *SYSTEM_SHUT-DOWN	Shut down system (VDI signal); no alarm should occur.		
97. *SERUPRO_ON	Enable block search PI in mode 5. In this mode, the block search is simulated by executing the program in "Programteste mode" up to the searched target block.		
98. *ESR	Extended stop and retract		
99. *BLOCKSEAR-CHRUN_SIGNAL	Block search (general) is currently activated; no alarm should occur, since it is possible that only the PI service is acknowledged negatively.		
100. *BLOCKSEAR-CHRUN_INTEGR	Intergrated block search, i.e. a new block search is started to a stopped program.		
101. *EXT_ZERO_POINT	External zero offset is activated via PLC. To this end, the travel is stopped, Reorg carried out, the interpreter switched over, then REPOS is used for selection; then the program is continued automatically.	1st channel not in AUTO or MDA. 2. The channel is stopped and the current block cannot be reorganized,	1. Select Auto or MDA. 2. Activate block change until the NC block can be reorganized.
102. *SINGLEBLOCK_IPONOSBLOF	Single block type 3 is enabled. With single block type 3, it is stopped at all main blocks. Contrary to single block type 3, the part program block SBLOF is ignored.		
103. SINGLEAX_STO-PALL_MASTER	Stops a single-axis motion (VDI signal) Not permitted if the axis is nit an independent CPU axis.		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
104. SINGLEAX_STOP-ALARM_MASTER	Stops a single-axis motion by an alarm. Not permitted if the axis is not an independent CPU axis.		
105. SINGLEAX_RESUME_MASTER	Continues a CPU axis motion (VDI signal); not permitted if the axis has been stopped first.		
106. SINGLEAX_RESET_MASTER	Cancels a CPU axis motion (VDI signal); not permitted if the axis is not an independent CPU axis.		
107. SINGLEAX_DELETE_MASTER	Deletes the distance to go for a single-axis motion (VDI signal); not permitted if the axis is not an independent CPU axis.		
108. SINGLEAX_PLCTRL_ON_MASTER	When turned on: The axis becomes an independent CPU axis (VDI signal); not permitted if the axis is not an independent CPU axis.		
109. SINGLEAX_PLCTRL_OFF_MASTER	When switched off: The axis becomes an independent CPU axis (VDI signal); only permitted if the axis is a neutral axis or programmed using synchronized actions.		
110. *SINGLEAX_JOG_WHEEL	available soon		
111. *SINGLEAX_JOG_PLUS_MASTER	available soon		
112. *SINGLEAX_JOG_MINUS_MASTER	available soon		
113. *SINGLEAX_JOG_PLUS_INC_MASTER	available soon		
114. *SINGLEAX_JOG_MINUS_INC_MASTER	available soon		

Table 12-4 Action list, continued

No./name	Explanation	Not allowed if ...	Remedy
115. *REPOSMODE-CHANGE	The event is triggered by the positive PLC edge of the signal "REPOS mode edge" triggered.	1. The channel is active (program running, block search, load machine data).	1. Press RESET to cancel the program or use STOP to stop the program (not with block search, load machine data)
116. *TOOLCHANGECMDON	Enables the tool management commands (channel VDI signal)	... the NCK channel state is not in the Ready condition	Press RESET to cancel the program or the process or wait for the end of the program
117. *TOOLCHANGECMDOFF	disables the tool management commands (channel VDI signal)	... the NCK channel state is not in the Ready condition.	Press RESET to cancel the program or the process or wait for the end of the program
118. *SIVLIMCHANGE	switches the desired safety limitations (SGE)	always permitted	
119. *STOPRUN	Stop Run, i.e. the NCK automatically stops at a block defined via OPI	Control system not in the AUTOMATIC mode	



13

Handlings Transformation

Chapter overview

Section	Title	Page
13.1	“Handling transformation” parameterization	13-5
13.2	Programming the standard function blocks for “handling transformation” with HPU or HT 6	13-10
13.3	User alarms and user messages	13-22
13.4	Changing the global data communication (SDB 210)	13-34
13.3	User alarms and user messages	13-22
13.5	Functions	13-35
13.6	NC Programming	13-69
13.7	Troubleshooting	13-78

Functionality

The “handling transformation” function is designed for use on **handling machines** and **robots**. It is a type of modular system where customers can configure transformations for their own machines provided the kinematic operations are implemented in the “Handling” transformation package.

Task

The task of the transformation is: to transform movements from the cartesian coordinate system in which they are programmed to the machine axis positions; for example: programmed movements of the tool tip.

Application

The “handling transformation” described here is designed to implement the largest possible amount of kinematic operations through machine data parameterization alone. Kinematic operations where between 2 and a maximum of 4 axes are involved in the transformation can presently be implemented. This is equivalent to up to four spatial degrees of freedom. Up to 3 degrees of freedom are possible for translation and 1 degree of freedom for orientation.

Handheld programming unit (HPU) / Handheld terminal HT 6

You can connect the HPU or HT 6 to the sub D socket connector (X8) on the FM 357-2. When the HPU or HT 6 is connected, no power units can be connected to the FM 357-2 via PROFIBUS DP. The axes must be controlled via the drive interface (X2).

You need the following components to connect the HPU or HT 6 to the FM 357-2 (see also Figure 13-1):

- Handheld programming unit (HPU), MPI type with system software
- Handheld Terminal HT 6 with system software
- HPU interconnecting cable
- HT 6 interconnecting cable
- HPU distributor box
- HT 6 distributor box
- MPI interconnecting cable (connection between CPU/distributor box and connection between CPU/FM 357-2)

For the order numbers of the individual components, see

Catalog NC Z, Order No.: E86060-K4490-A001-A□

or

Catalog NC 60, Order No.: E86060-K4460-A101-A□

or

Catalog ST 70, Order No.: E86060-K4670-A101-A□

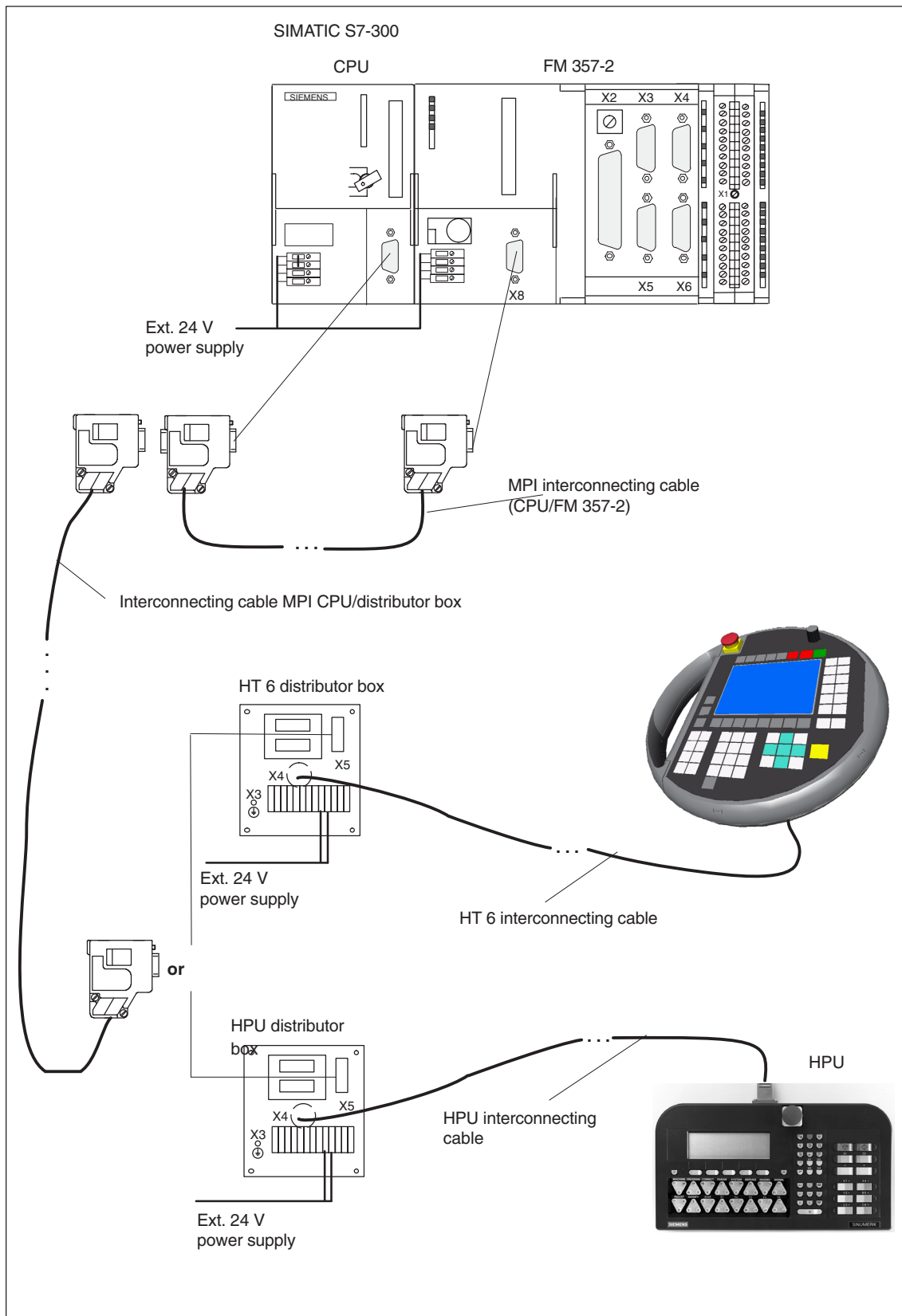


Fig. 13-1 Overview FM 357-2 with HPU or HT 6 (example)

You will need the following documentation in addition to this manual for the “handling transformation” function:

- *Operator Components* manual, Order No.: 6FC5 297-6AA50-0BP0
- *Handheld Programming Device Operator’s Guide*, Order No.: 6FC5 298-5AD20-0BP0
- *Handheld Terminal HT 6 Operator’s Guide* Order No.: 6FC5 298-4AD60-0BP0

Important information for the user

When using a PHG/HT 6, only one FM 357-2H can be used per CPU.

Only the first channel is available for the FM 357-2H. The “handling transformation” can therefore not be activated in the second channel or a further channel.

The axes (machine, plant) must be parameterized before the handling transformation.

The “handling transformation” function cannot be operated in conjunction with the “Travel to fixed stop” function.

Only tool lengths are available for entering tool parameters.

It is not possible to cross a pole when the transformation is active. Axis overloads can occur at singular positions (see Section 13.6.2).

Depending on the type of kinematic operation, individual axis overloads can occur at certain positions if the axis is traversed with the transformation active. The feedrate is not adapted automatically. The user must reduce the feedrate as appropriate at critical points.

13.1 Parameterization of “Handling transformation”

General

You will find information for parameterization in Chapter 5.

This chapter only describes the sequence of operations used to enter the machine data (parameters) in the machine data wizard for the “Handling transformation” function.

You can call up the machine data wizard as follows:

1. Open an online or offline project (see Section 5.5).
2. Select the “machine data” area in the project window.
3. Double-click the block to open the machine data wizard.
4. Click the “Handling transformation” tab

Machine data list

The tables below describe the machine data (parameters) which you can enter for the “Handling transformation” function using the parameterization tool.

Table 13-1 General machine data for “Handling transformation”

Parameters	Value range/meaning	Unit	See Section
Reset response for “active kinematic transformation”	Bit7 = 0: No transformation is active after Reset or end of program. (default value) Bit7 = 1: The handling transformation remains active after a Reset or end of program.	–	13.6.3
Assignment between geometry axis and channel axis for transformation [geoaxis no.] Geoaxis no.: 0...2	Assignment table of geometry axes for handling transformation Same as channel and axis configuration, see Section 9.1, except applies only when handling transformation is active. Note If you use geometry axes outside the handling transformation (MD20050, assignment ‘geometry axis to channel axis’ = 0), you must enter different identifiers in MD20060 (geometry axis name in the channel) and MD20080 (channel axis name in the channel) using the list parameterization. Example: MD20060 \$MC_AXCONF_GEOAX_NAME_TAB[0]=”X” MD20060 \$MC_AXCONF_GEOAX_NAME_TAB[1]=”Y” MD20060 \$MC_AXCONF_GEOAX_NAME_TAB[2]=”Z” MD20080 \$MC_AXCONF_CHANAX_NAME_TAB[0]=”XC” MD20080 \$MC_AXCONF_CHANAX_NAME_TAB[1]=”YC” MD20080 \$MC_AXCONF_CHANAX_NAME_TAB[2]=”ZC” You will get to the list parameterization from the machine data wizard using the menu View > Table View .	–	

Table 13-2 Machine data (parameters) for “Handling transformation”

Parameters	Value range/meaning	Unit	See Section
Kinematic class	1: Standard transformation (default value) 2: Special transformation	–	13.5.3
Axis type for transformation [axis no.] Axis no.: 0...3	1: Linear axis 3: Rotary axis [1, 1, 1, 3] = default value This machine data identifies the type of axis used in the transformation.	–	13.5.2
Special kinematic type	3: 2-axis SCARA with permanent link to tool 4: 3-axis SCARA with degrees of freedom X, Y, A 5: 2-axis articulated arm with connection between axis 1 and axis 2 7: 4-axis SCARA with connection between axis 1 and axis 2 This machine data identifies the type of special kinematics.	–	13.5.3
Basic axis identifier	1: SS (gantry) (default value) 2: CC (SCARA) 3: NR (articulated arm) 4: SC (SCARA) 5: RR (articulated arm) 6: CS (SCARA) 7: NN (articulated arm) This machine data identifies the type of basic axis arrangement. The first 3 axes are normally referred to as basic axes.	–	13.5.2
Number of transformed axes	1...4 (3 = default value) This machine data identifies the number of axes involved in the transformation.	–	13.5.2
Axis 4 parallel/antiparallel to last basic axis	0: Axis 4 is not parallel (default value) 1: Axis 4 is parallel/antiparallel This machine data identifies whether the 4th axis is parallel/antiparallel to the last rotary basic axis. This machine data is only relevant for kinematic operations with more than 3 axes.	–	13.5.3
Basic axis lengths A and B [n] n = 0...1	n = 0: Basic axis length A n = 1: Basic axis length B [0.0, 500.0] = default value This machine data identifies basic axis lengths A and B. These lengths are specially defined for each type of basic axis.	[mm], [inches]	13.5.3

CS = coordinate system

Table 13-2 Machine data (parameters) for “Handling transformation”, continued

Parameters	Value range/meaning	Unit	See Section
Attachment of hand (position component) [n] n = 0...2	n = 0: X component n = 1: Y component n = 2: Z component [0.0, 0.0, 0.0] = default value This machine data identifies the position component of frame T_X3_P3 which connects the basic axes to the hand.	[mm], [inches]	13.5.2
Attachment of hand (rotation component) [n] n = 0...2	n = 0: Rotation through RPY angle A n = 1: Rotation through RPY angle B n = 2: Rotation through RPY angle C [0.0, 0.0, 0.0] = default value This machine data identifies the rotation component of frame T_X3_P3 which connects the basic axes to the hand.	[degrees]	13.5.2
Frame between hand point and flange coordinate system (position component) [n] n = 0...2	n = 0: X component n = 1: Y component n = 2: Z component [0.0, 0.0, 0.0] = default value This machine data identifies the position component of frame T_FL_WP which connects the flange to the last internal hand point coordinate system permanently defined via the handling transformation.	[mm], [inches]	13.5.2
Frame between hand point and flange coordinate system (rotation component) [n] n = 0...2	n = 0: Rotation through RPY angle A n = 1: Rotation through RPY angle B n = 2: Rotation through RPY angle C [0.0, 0.0, 0.0] = default value This machine data identifies the rotation component of frame T_FL_WP which connects the flange to the last internal hand point coordinate system permanently defined via the handling transformation.	[degrees]	13.5.2
Frame between foot point and internal coordinate system (position component) [n] n = 0...2	n = 0: X component n = 1: Y component n = 2: Z component [0.0, 0.0, 0.0] = default value This machine data identifies the position component of frame T_IRO_RO which connects the foot point coordinate system defined by the user to the internal robot coordinate system.	[mm], [inches]	13.5.2

CS = coordinate system

Table 13-2 Machine data (parameters) for “Handling transformation”, continued

Parameters	Value range/meaning	Unit	See Section
Frame between foot point and internal coordinate system (rotation component) [n] n = 0...2	n = 0: Rotation through RPY angle A n = 1: Rotation through RPY angle B n = 2: Rotation through RPY angle C [0.0, 0.0, 0.0] = default value This machine data identifies the rotation component of frame T_IRO_RO which connects the foot point coordinate system defined by the user to the internal robot coordinate system.	[degrees]	13.5.2
Offset between mathematical and mechanical zero [axis no.] Axis no.: 0...3	This MD can be used to adapt the zero point for a axis to the mathematical zero point defined via the transformation. rotary axis [degrees], linear axis [mm], [inch] [0.0, 0.0, 0.0, 0.0] = default value	–	13.5.2
Adaptation of physical and mathematical direction of rotation [axis no.] Axis no.: 0...3	+1: Same direction of rotation –1: Different direction of rotation [1, 1, 1, 1] = default value This machine data can be used to adapt the mathematical direction of rotation of the axes to the physical direction of rotation.	–	13.5.2
Reordering of axes [n] n = 0...3	1...4 [1, 2, 3, 4] = default value This machine data can be used to change the order of axes, in order to transform the kinematic system to a standard kinematic system.	–	13.5.2
Cartesian velocities [n] n = 0...2	n = 0: Velocity in X direction n = 1: Velocity in Y direction n = 2: Velocity in Z direction [600000.0, 600000.0, 600000.0] = default value This machine data can be used to define a velocity command for the Cartesian directions for traversing blocks with G0.	[mm/min], [inch/min]	13.5.2
Cartesian accelerations [n] n = 0...2	n = 0: Acceleration in X direction n = 1: Acceleration in Y direction n = 2: Acceleration in Z direction [0.5, 0.5, 0.5] = default value This machine data can be used to define an acceleration command for the Cartesian directions for traversing blocks with G0.	[m/s ²] [Inch/s ²]	13.5.2

CS = coordinate system

Table 13-2 Machine data (parameters) for “Handling transformation”, continued

Parameters	Value range/meaning	Unit	See Section
Orientation angle velocity	Velocity angle A [1.6667] = default value This machine data can be used to define a velocity command for the orientation angle for traversing blocks with G0. (for 4 axes only)	[rev/min]	13.5.2
Orientation angle acceleration	Acceleration angle A [0.0028] = default value This machine data can be used to define an acceleration command for the orientation angle for traversing blocks with G0. (for 4 axes only)	[rev/s ²]	13.5.2

CS = coordinate system

13.2 Programming the standard function blocks for “Handling transformation” with the HPU or HT 6

General

The procedure for programming the standard function blocks is described in Chapter 6. This section describes only the blocks you need in addition to the basic functions for “handling transformation” **with** the HPU or HT 6.

The signals for “handling transformation” without the HPU or HT 6 are described in Section 6.6.

In addition to the blocks described in Chapter 6, you will need internal data blocks DB 7 (only HT 6), DB 11 (HPU only), DB 17, DB 19 (internal DBs), FC 21 and user BD 21 for operation with the HPU or HT 6 (these blocks are supplied).

Notice

Read the supplied **readme.wri** file in the directory **[STEP7 directory]\S7libs\FM357_2L\readme.wri**.

Using a HPU

To be able to work with the HPU, you must copy all blocks (DB 11, DB 17, DB 19, DB 21, FC 21, system data) from the directory “**PHU > Blocks**” into your project under **SIMATICxxx > CPUxxx > S7 Program > Blocks** in addition to the basic function blocks. During this copying process, you will be asked whether you want to overwrite the existing system data. Confirm this interrogation with **YES**. Since these system data contain the hardware configuration (SDB 0) and the configuration of the global data communication (SDB 210), your original hardware has been overwritten.

Now reopen the hardware configuration (start HW Config) in your project. Save and recompile the hardware project using the menu commands **Station > Save and compile** so that your hardware configuration is refreshed and updated in the system data (SDB 0).

When working with the symbolic programming, copy the symbolic designations from the PHU S7 program from the **Symbols** container into your project. A call example for the PHU block FC21 is to be found in the supplied awl source “fm357ob_phght6” (S7 program **PHU > Sources**). Copy the source and adapt the calls in your project accordingly (OB 1).

Now you can continue as described in Section 6.1.7, Point 13., load your blocks into the CPU and start the CPU.

Using an HT 6

To be able to work with the HT6, you must copy all blocks (DB 7, DB 17, DB 19, DB 21, FC 21, system data) from the directory "PHU > **Blocks**" in to your project under **SIMATICxxx > CPUxxx > S7 Program > Blocks** in addition to the basic function blocks. During this copying process, you will be asked whether you want to overwrite the existing system data. Confirm this interrogation with **YES**. Since these system data contain the hardware configuration (SDB 0) and the configuration of the global data communication (SDB 210), your original hardware has been overwritten.

Now reopen the hardware configuration (start HW Config) in your project. Save and recompile the hardware project using the menu commands **Station > Save and compile** so that your hardware configuration is refreshed and updated in the system data (SDB 0).

When working with the symbolic programming, copy the symbolic designations from the PHU S7 program from the **Symbols** container into your project. A call example for the PHU block FC21 is to be found in the supplied awl source "fm357ob_phght6" (S7 program **PHU > Sources**). Copy the source and adapt the calls in your project accordingly (OB 1).

Now you can continue as described in Section 6.1.7, Point 13., load your blocks into the CPU and start the CPU.

Overview of data communication HPU or HT 6 / CPU / FM 357-2

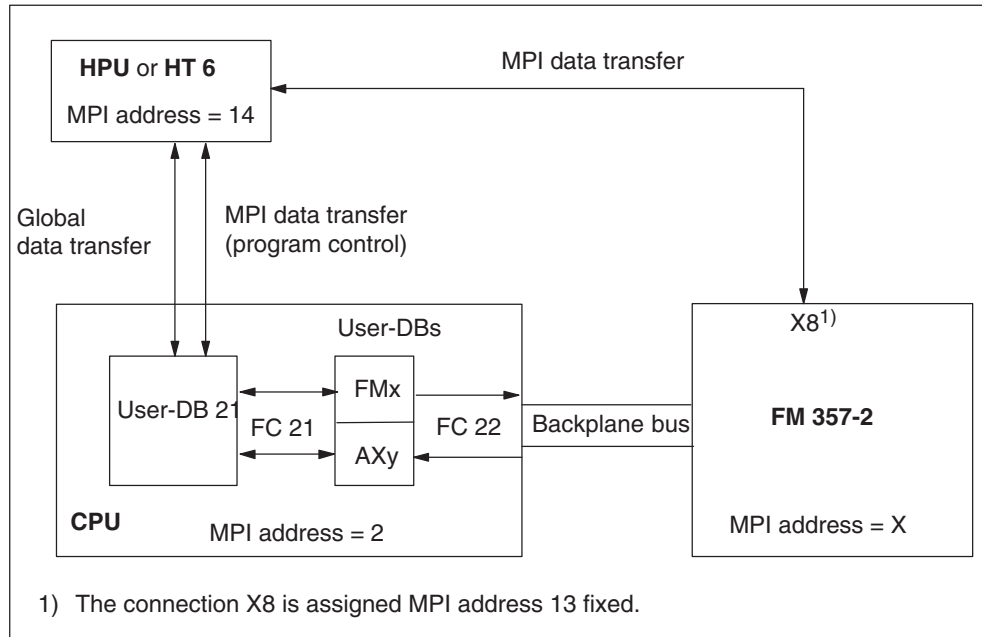


Fig. 13-2 Overview of data communication HPU or HT 6 / CPU / FM 357-2

The overview shows how data communication takes place between the HPU or HT 6, CPU and FM 357-2.

The HPU or HT 6 communicates with the CPU across two paths:

- Global data transfer via SDB 210
- Data transfer via the MPI interface (automatic)

Global data are transferred from the CPU to the HPU or HT 6 via signals DBB2 to 9 in user DB 21. Signals DBB16 to 23 in user DB 21 are transferred from the HPU or HT 6 to the CPU. For signal transfer, it is necessary to include SDB 210 in your project. You will find the SDB in the “FM357_2L” library supplied in directory

HPU > Blocks > System Data

or

HT 6 > Blocks > System Data.

The above data can only be exchanged if SDB 210 has been loaded into the CPU.

Signals DBB 24 to 26 (program control signals) are transferred to user DB 21 by means of MPI data communication.

The task of FC 21 is to monitor data exchange (global data) and, if input parameter “UDB_TO_IF” = TRUE, to synchronize the “HPU” or “HT 6” interface with user DBs “FMx” and “AXy” (gray fields in user DB 21).

Data transfer of the signals from user DBs “FMx” and “AXy” to and from the FM is performed cyclically in FC 22 (see Section 6.3.3).

13.2.1 FC 21: HPUHT6 – Transfer of the HPU / HT 6 signals to and from the interface (user DB)

General

The MMC and MCP signals described in the *Handheld Programming Unit* or the *Terminal HT 6 Operator's Guides* are combined in one unit in the following block description (HPU/HT 6 signals).

Task

The following signals are transferred from the HPU/HT 6 to the interface (user DBs) using FC 21:

- Operating modes
- Workpiece (WCS) / machine coordinate system (MCS) switchover
- Travel keys
- Override

The transfer of HPU/HT 6 signals to the interface (user DBs "FMx" and AXy") can be deactivated by setting FC 21 parameter "UDB_TO_IF" = FALSE.

As soon as the block is called, monitoring of the signals between the CPU and HPU/HT 6 is active. This monitoring routine cannot be deactivated. If the communication link is lost, this is indicated by an error code (see Table 13-4).

If the HPU/HT 6 malfunctions, the global data interface (user DB 21) is cleared.

The following specifications are valid for the override, INC and axis travel keys, independent of the active mode or the selected coordinate system:

- **Override:**

The override is transferred to the interface of channel 1 (user DB "FMx") and to the interface of the axes (user DB "AXy").

In addition, the following signals are set:

- "Path override, feedrate override active" (user DB "FMx", DBX107.7)
- "Activate override" (user DB "AXy", DBX1.7+m)

- **INC and axis travel key machine functions:**

When the MCS is active, the signals are transferred to the interface of the selected machine axis (user DB "AXy", DBB4+m).

When the WCS is active, the signals of the first three axes are transferred to the geometry axis interface of the configured channel (user DB "FMx", DBB114, 115, 116).

Internal flow of FC 21

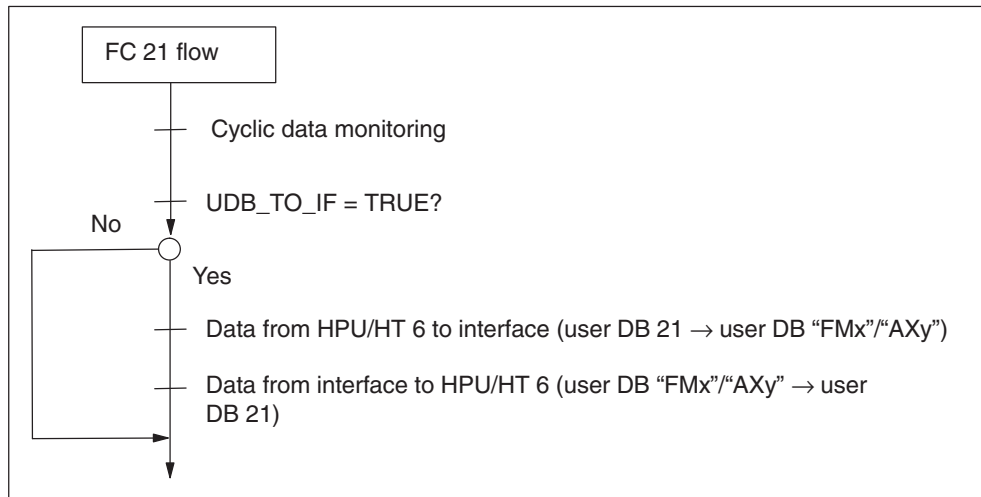


Fig. 13-3 Internal flow of FC 21

Call options

Block FC 21 must only be called once in cyclic mode.

It is not possible to connect one HPU/HT 6 to several FMs.

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)												
<div style="border: 1px solid black; padding: 5px; width: fit-content; margin: auto;"> <p style="text-align: center; margin: 0;">FC 21</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%; border-right: 1px solid black; padding: 2px;">EN</td> <td style="padding: 2px;"></td> <td style="width: 30%; padding: 2px;">ENO</td> <td style="width: 30%; border-right: 1px solid black; padding: 2px;"></td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px;">FMDB_NO</td> <td style="padding: 2px;"></td> <td style="padding: 2px;">RET_VAL</td> <td style="border-right: 1px solid black; padding: 2px;"></td> </tr> <tr> <td style="border-right: 1px solid black; padding: 2px;">UDB_TO_IF</td> <td style="padding: 2px;"></td> <td style="padding: 2px;"></td> <td style="border-right: 1px solid black; padding: 2px;"></td> </tr> </table> </div>	EN		ENO		FMDB_NO		RET_VAL		UDB_TO_IF				<pre> CALL FC 21 FMDB_NO := UDB_TO_IF := RET_VAL := </pre>
EN		ENO											
FMDB_NO		RET_VAL											
UDB_TO_IF													

Description of parameters

The following table shows the parameters of FC 21.

Table 13-3 FC 21 parameters

Name	Data type	I/O type	Value range	Significance
FMDB_NO	INT	I	1... see CPU	No. of assigned user DB "FMx"
UDB_TO_IF	BOOL	I	TRUE = yes FALSE = no	Transfer of HPU/HT 6 data to user DBs
RET_VAL	WORD	R	–	Return value of FC in the event of an error

Parameter types: I = input parameter, Q = output parameter

FC 21 troubleshooting

Table 13-4 FC 21 troubleshooting

Error code in output parameter RET_VAL	Significance	FC 21 error response	Error cause
W#16#0201	Breakdown of communication between HPU/HT 6 and CPU	Global data interface in user DB 21 is cleared	Connection between HPU/HT 6 and CPU lost (< 1 s)

Selection signals from HPU to interface (user DB)

Table 13-5 Selection signals from HPU to interface (user DB)

Source: HPU	Destination: User DB...
AUTOMATIK	... "FMx", DBX100.0
MDA (MDI)	... "FMx", DBX100.1
JOG	... "FMx", DBX100.2
TEACH_IN	... "FMx", DBX101.0
REPOS	... "FMx", DBX101.1
REF (reference point approach)	... "FMx", DBX101.2
INC variable	... "FMx", DBX102.5
Direction keys, rapid traverse override for WCS	
Direction key +	... "FMx", DBX114.7
Direction key –	... "FMx", DBX114.6
Direction key +	... "FMx", DBX115.7
Direction key –	... "FMx", DBX115.6
Direction key +	... "FMx", DBX116.7
Direction key –	... "FMx", DBX116.6

Table 13-5 Selection signals from HPU to interface (user DB), continued

Source: HPU	Destination: User DB...
Direction keys, rapid traverse override for MCS	
Direction key +	... "AXy", DBX4.7+m
Direction key -	... "AXy", DBX4.6+m
Override	... "FMx", DBB106
	... "AXy", DBB0+m
Start	... "FMx", DBX108.1
Stop	... "FMx", DBX108.3
Reset	... "FMx", DBX108.7
Step (single block)	... "FMx", DBX103.4

Checkback signals from the interface (user DB) to the HPU/HT 6

Table 13-6 Checkback signals from the interface (user DB) to the HPU

Destination: HPU	Source: User DB...
AUTOMATIK	... "FMx", DBX120.0
MDA (MDI)	... "FMx", DBX120.1
JOG	... "FMx", DBX120.2
TEACH_IN	... "FMx", DBX 121.0
REPOS	... "FMx", DBX121.1
REF (reference point approach)	... "FMx", DBX121.2
Reset	... "FMx", DBX125.7

WCS/MCS and Step output is controlled by pressing the key.

Program control to the interface (user DB)

Table 13-7 Program control from the HPU/HT 6 to the interface

Source: HPU/HT 6	Target: User DB...
M01 selected	... "FMx", DBX103.5
Dry run feedrate selected	... "FMx", DBX103.6
Rapid traverse override selected	... "FMx", DBX107.6
Program test selected	... "FMx", DBX 104.7
Skip block selected	... "FMx", DBX105.0

Notice

Do not write signals/data to reserved or hidden areas (internally reserved) of user DB 21.

13.2.2 User data block (user DB 21)

The following table describes the structure of user DB 21.

The fields with a grey background are processed and controlled by FC 21.

All other signals can be processed and controlled by you.

User data block for HPU (user DB 21: PHG_IF)

Table 13-8 User DB 21 for HPU

User DB 21		Signals to/from HPU						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB0 to DBB1	Reserved							
Global data transfer (signals from CPU to HPU)								
DBB2	Function keypad							
	Reference-point approach	TEACH IN	Automatic	MDI (MDA)	JOG	QUIT	Reset	WCS/MCS
DBB3	Function keypad							
		U4	U3	Switchover key	U2	U1	INC	REPOS
DBB4	Direction plus (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB5	Direction minus (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB6	Big keys							
	Signal	Diagno	Service	System	Param	Correct	Program	Machine
DBB7	Big keys							
	F5	F4	F3	F2	F1	Step	Modify	Insert
DBB8	Start keypad							
			VAL + (Override+)	VAL - (Override-)	S2	S1	Start	Stop
DBB9	Reserved							
DBB10 to DBB11	Reserved							
DBB12 to DBB15	GD Status (global status data)							

Table 13-8 User DB 21 for HPU, continued

User DB 21		Signals to/from HPU						
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Global data transfer (signals from HPU to CPU)								
DBB16	Function keypad							
	Reference point approach	TEACH IN	Automatic	MDI (MDA)	JOG	QUIT	Reset	WCS/MCS
DBB17	Function keypad							
		U4	U3	Switchover key	U2	U1	INC	REPOS
DBB18	Direction plus (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB19	Direction minus (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB20	Big keys							
	Signal	Diagno	Service	System	Param	Correct	Program	Machine
DBB21	Big keys							
	F5	F4	F3	F2	F1	Step	Modify	Insert
DBB22	Start keypad							
		(HW 0)	VAL + (Override+)	VAL - (Override-)	SF2	SF1	Start	Stop
DBB23	Reserved							
MPI data transfer (program control)								
DBB24		Dry run feed selected	M01 selected		DRF selected 1)			
DBB25	Program test selected							
DBB26								Skip block selected
DBB27 to DBB29	Reserved							

Signals put in brackets do not exist in the hardware, but are emulated by the software.

HW 0 = FALSE (HPU)

1) not FM 357-2

User data block for HT 6 (user DB 21: PHG_IF)

Table 13-9 User DB 21 for HT 6

AW-DB 21								
Signals to/from HPU								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB0 to DBB1	reserved							
Global data transfer (signals from CPU to HPU)								
DBB2	Function keypad							
	Reference point approach	TEACH IN	Automatic	MDI (MDA)	JOG	QUIT	Reset	WCS/MCS
DBB3	Function keypad							
	Control Panel Function	U4	U3		U2	U1	INC	REPOS
DBB4	Plus direction (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB5	Minus direction (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB6	reserved							
DBB7	Big keys							
		U8	U7	U6	U5	Step		
DBB8	Start keypad							
					S2	S1	Start	Stop
DBB9	reserved							
DBB10 to DBB11	reserved							
DBB12 to DBB15	GD status (global status data)							

Table 13-9 User DB 21 for HT 6, continued

AW-DB 21								
Signals to/from HPU								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Global data transfer (signals from HPU to CPU)								
DBB16	Function keypad							
	Reference point approach	TEACH IN	Automatic	MDI (MDA)	JOG	QUIT (Cancel)	Reset	WCS/MCS
DBB17	Function keypad							
	Control Panel Function	U4	U3		U2	U1	INC	REPOS
DBB18	Plus direction (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB19	Minus direction (Jog)							
					Axis 4	Axis 3	Axis 2	Axis 1
DBB20	reserved							
DBB21	Big keys							
		U8	U7	U6	U5	Step		
DBB22	Start keypad							
		(HW 0)			S2	S1	Start	Stop
DBB23	Path override							
				E	D	C	B	A
MPI data transfer (program control)								
DBB24		Dry run feed selected	M01 selected		DRF selected 1)			
DBB25	Program test selected				Rapid traverse override selected			
DBB26								Skip block selected
DBB27 to DBB29	reserved							

Signals put in brackets do not exist in the hardware, but are emulated by the software.

HW 0 = FALSE (HPU)

1) not FM 357-2

Description of signals

Address	Name/function
MPI data transfer (program control)	
DBX24.5	M01 selected M01 program control has been selected from the HPU/HT 6.
DBX24.6	Dry run feed selected Dry run feed program control has been selected from HPU/HT 6.
DBX25.3	Rapid traverse override selected Rapid traverse override program control has been selected from the HPU/HT 6.
DBX25.7	Program test selected Program test program control has been selected from the HPU/HT 6.
DBX26.0	Skip block selected Skip block program control has been selected from the HPU/HT 6.

13.3 User alarms and user messages

This Section describes how to generate user alarms and user messages during the operation and how these are displayed on the PHG or HT 6.

To this end, you will need the blocks FC 20 and DB 20. These blocks are included in the supplied library "FM357_2L" under the directory **PHG > Blocks** or **HT6 > Blocks** enthalten.

General

The signaling functions are based on the system diagnosis functions integrated in the operating system of the CPU. These functions provide the following features:

- All important system states and system transitions are entered by the CPU operating system in a **diagnosis status list**. In addition, communication events and (I/O) module diagnostic data (with modules featuring diagnostic functions) are entered.
- In addition, the diagnostic events leading to a system stop are entered in the time order of occurrence with time stamp in a **diagnostic buffer** (ring buffer).
- The events entered in the diagnostic buffer are transmitted automatically to HMI systems (e.g. PHG/HT 6) via the MPI interface provided that they did not log in as ready to receive (signaling service). The transmission to the station logged in is a function of the CPU operating system. Both the reception and the integration of the messages are provided by the PHG/HT 6 software.
- The events are entered in the diagnostic buffer in coded form. The appropriate texts must be configured in the PHG or HT 6 (see Sections 13.3.3 and 13.3.4).

Together with the basic program, an FC is provided for acquiring the messages (FC 20); it acquires the events to be signaled divided into signal groups and signals them via the diagnostic buffer to the PHG or HT 6. The message acquisition provides the following features:

- The bit fields are evaluated by FC 20 several times.
 - **Evaluation 1: Acquisition of user alarms**
It is fixed which signals of a group generate an alarm during their transition from "FALSE" to "TRUE" (see Table 13-11, structure of DB 20).
 - **Evaluation 2: Acquisition of user messages**
It is fixed which signals of a group generate a message during their transition from "FALSE" to "TRUE" (see Table 13-11, structure of DB 20).
- By default, the range of the user areas is fixed to 5 areas comprising 8 bytes each. Each user area comprises a total of 64 messages, with 32 of them alarms and 32 messages.

Acknowledgement concept

The following acknowledgement concepts are used with regard to the user alarms:

- **User messages**

The user messages are characterized by the fact that they are intended to indicate normal operating states of the machine as an information for the operator. Therefore, they do not require any acknowledgement signals. Both the coming and the going of the event is acquired and an entry made into the diagnostic buffer. Using the identifiers "Message has come" and "Message has gone", the PHG/HT 6 creates an up-to-date mapping of the existing messages.

- **User alarms**

User alarms are used to indicate error conditions on the machine which normally result in machine standstill. If several errors occur at the same time, it is important for troubleshooting to know the order of their occurrence. This is indicated, on the one hand, by the order of the entries in the diagnostic buffer and, on the other hand, by the time stamp contained in each entry.

If the error cause does no longer exist, the associated error message will only be cleared if an acknowledgement by the user is provided (e.g. by pressing a customer key). As a response to this signal, the FC 20 will try to find out which of the errors already signaled have disappeared and will make an appropriate entry in the diagnostic buffer with the identification "Error has gone". Thus, the PHG/HT 6 can also generate an up-to-date mapping of the existing error messages. With the messages still pending, the time of their occurrence is kept (in contrast to a repeated request).

- **User program**

The user program must call the block FC 20 in the cyclic program part and set or reset the appropriate bits in the user areas in DB 20.

13.3.1 FC 20: AL_MSG – user alarms and user messages

Task

FC 20 is used to evaluate the signals entered in DB 20 (user area from 0 to 4, DBX 0.0 to DBX 39.7) and to display coming and going user alarms and user messages on the PHG/HT 6.

Both in the case of user alarms and of user messages, the coming signals (positive edge) are displayed immediately.

Going normal signals (negative edge) will only be cleared immediately in the case of user messages. User alarms will only be cleared if you have reset the appropriate signal in DB 20 and have acknowledged it with the FC 20 parameter "QUIT".

The internal (reserved) area of DB 20 must be initialized with setting the "INIT" parameter (positive edge) of FC 20 after each transition of the operating status of the CPU from STOP to RUN. If DB 20 is not initialized, any user alarms/messages that have been set, but have not been acknowledged, will not be signaled again.

The user alarms and messages are set/reset by the user in data block DB 20. For the detailed structure of this DB and the numbers of the user alarms or messages, see Table 13-11.

Call possibilities

The block FC 20 may be called cyclically only once.

Call in LAD (ladder diagram)		Call in STL (statement list)
FC 20	ENO	CALL FC 20 INIT := QUIT :=

Description of the parameters

The parameters of FC 20 are described in the Table below.

Table 13-10 FC 20 parameters

Name	Data type	P type	Range of values	Meaning
INIT	BOOL	I	TRUE FALSE	Initialization of the internal areas of DB 20
QUIT	BOOL	I	TRUE FALSE	User alarm acknowledgement

Parameter types: I = input parameters

Call example

Call in OB 100 for initializing DB 20

```
STL
-----
CALL  "AL_MSG"                // Call of FC 20 in OB 100
      INIT   := TRUE           // Initialize DB 20
      QUIT   := FALSE
```

Call in the cyclic block (OB 1)

```
STL
-----
CALL  "AL_MSG"                // Call of FC 20
      INIT   := FALSE         // No initialization of DB 20
      QUIT   := M 2.0         // Free input for error
                               acknowledgement
```

13.3.2 Message signals in DB 20

Task

You can display messages for individual signals on the PHG or HT 6 via the DB 20. The signals are divided into predefined groups. When setting or resetting a message, the message number is transmitted to the PHG/HT 6. A text can be stored in the PHG/HT 6 for each message number (see Sections 13.3.3 and 13.3.4).

The DB 20 is divided into a user and into an internal structure.

The internal structure consists of the five user areas which you can also access using symbols (bytes 0...39). The area from byte 40 to 149 is an internal area required for the FC 20. This area is to be considered as a reserved area for the user. Any inadvertent wrong access to this reserved area on the side of the user will result in malfunctions of the FC 20.

Symbolic access (in the symbol browser: AL_MSG_DB = DB 20)

e.g.:

```
S  AL_MSG_DB.A7000xx[2]      // Set alarm 700.002
R  AL_MSG_DB.A7000xx[63]     // Reset message 700.063
```

Notice

In DB 20, the bit of the user alarm or of the user messages must be set for several OB1 cycles to make sure that a message can also be displayed on the PHG/HT 6.

User areas in DB 20

Table 13-11 Structure of DB 20

DB 21								
User areas								
User area 0								
DBB0	Alarm							
	700.007	700.006	700.005	700.004	700.003	700.002	700.001	700.000
DBB1	Alarm							
	700.015	700.014	700.013	700.012	700.011	700.010	700.009	700.008
DBB2	Alarm							
	700.023	700.022	700.021	700.020	700.019	700.018	700.017	700.016
DBB3	Alarm							
	700.031	700.030	700.029	700.028	700.027	700.026	700.025	700.024
DBB4	Message							
	700.039	700.038	700.037	700.036	700.035	700.034	700.033	700.032
DBB5	Message							
	700.047	700.046	700.045	700.044	700.043	700.042	700.041	700.040
DBB6	Message							
	700.055	700.054	700.053	700.052	700.051	700.050	700.049	700.048
DBB7	Message							
	700.063	700.062	700.061	700.060	700.059	700.058	700.057	700.056
User area 1								
DBB8	Alarm							
	700.107	700.106	700.105	700.104	700.103	700.102	700.101	700.100
DBB9	Alarm							
	700.115	700.114	700.113	700.112	700.111	700.110	700.109	700.108
DBB10	Alarm							
	700.123	700.122	700.121	700.120	700.119	700.118	700.117	700.116
DBB11	Alarm							
	700.131	700.130	700.129	700.128	700.127	700.126	700.125	700.124
DBB12	Message							
	700.139	700.138	700.137	700.136	700.135	700.134	700.133	700.132
DBB13	Message							
	700.147	700.146	700.145	700.144	700.143	700.142	700.141	700.140
DBB14	Message							
	700.155	700.154	700.153	700.152	700.151	700.150	700.149	700.148
DBB15	Message							
	700.163	700.162	700.161	700.160	700.159	700.158	700.157	700.156

Table 13-11 Structure of DB 20, continued

DB 21								
User areas								
User area 2								
DBB16	Alarm							
	700.207	700.206	700.205	700.204	700.203	700.202	700.201	700.200
DBB17	Alarm							
	700.215	700.214	700.213	700.212	700.211	700.210	700.209	700.208
DBB18	Alarm							
	700.123	700.222	700.221	700.220	700.219	700.218	700.217	700.216
DBB19	Alarm							
	700.231	700.230	700.229	700.228	700.227	700.226	700.225	700.224
DBB20	Message							
	700.239	700.238	700.237	700.236	700.235	700.234	700.233	700.232
DBB21	Message							
	700.247	700.246	700.245	700.244	700.243	700.242	700.241	700.240
DBB22	Message							
	700.255	700.254	700.253	700.252	700.251	700.250	700.249	700.248
DBB23	Message							
	700.263	700.262	700.261	700.260	700.259	700.258	700.257	700.256
User area 3								
DBB24	Alarm							
	700.307	700.306	700.305	700.304	700.303	700.302	700.301	700.300
DBB25	Alarm							
	700.315	700.314	700.313	700.312	700.311	700.310	700.309	700.308
DBB26	Alarm							
	700.323	700.322	700.321	700.320	700.319	700.318	700.317	700.316
DBB27	Alarm							
	700.331	700.330	700.329	700.328	700.327	700.326	700.325	700.324
DBB28	Message							
	700.339	700.338	700.337	700.336	700.335	700.334	700.333	700.332
DBB29	Message							
	700.347	700.346	700.345	700.344	700.343	700.342	700.341	700.340
DBB30	Message							
	700.355	700.354	700.353	700.352	700.351	700.350	700.349	700.348
DBB31	Message							
	700.363	700.362	700.361	700.360	700.359	700.358	700.357	700.356

Table 13-11 Structure of DB 20, continued

DB 21								
User areas								
User area 4								
DBB32	Alarm							
	700.407	700.406	700.405	700.404	700.403	700.402	700.401	700.400
DBB33	Alarm							
	700.415	700.414	700.413	700.412	700.411	700.410	700.409	700.408
DBB34	Alarm							
	700.423	700.422	700.421	700.420	700.419	700.418	700.417	700.416
DBB35	Alarm							
	700.431	700.430	700.429	700.428	700.427	700.426	700.425	700.424
DBB36	Message							
	700.439	700.438	700.437	700.436	700.435	700.434	700.433	700.432
DBB37	Message							
	700.447	700.446	700.445	700.444	700.443	700.442	700.441	700.440
DBB38	Message							
	700.455	700.454	700.453	700.452	700.451	700.450	700.449	700.448
DBB39	Message							
	700.463	700.462	700.461	700.460	700.459	700.458	700.457	700.456
Internal area								
DBB40	reserved							
...								
DBB149								

13.3.3 Configuring alarm and message texts for PHG

Prerequisite

- PC with “MS DOS” (6.x or higher)
- V.24 cable between the COM 1 interface of the PHG and the COM 1 or COM 2 interface of your PC/PG (programming device).
- Memory requirements on the PC/PG hard disk: approx. 3 MB
- You have installed the system software for the PHG on your PG/PC and have completed this process.

Amending alarm and message texts

To edit the alarm and message texts, use the DOS editor **edit**. The standard texts contained in the text files can be overwritten or added by user-specific texts.

The alarm and message texts in the five standard languages with the SIEMENS standard entries are to be found in the directory **C:\proj_hpu\text\al\...**

The following abbreviations are used for the individual languages:

D	for German
G	for English
F	for French
E	for Spanish
I	for Italian

To edit the alarm and message texts, proceed as follows:

1. Start the editor **edit**.
2. Edit the alarm and text files **alp.txt** in the desired 1st and 2nd display languages.

Note:

Make sure that number and order of the alarm and message texts (number of lines) of the selected languages coincide.

The following numbers are available for the alarm and message texts: 700000 to 700463 (see Table 13-11).

Format of the text file alp.txt for the alarm and message texts

The file is structured as follows:

Message no.	Display ¹⁾	Help ID ²⁾	Text on the PHG
700000	0	0	"User alarm text"
...			...
700032	0	0	"User message text"
// Text file for user alarms and user messages			

- 1) 0 = display in the alarm line; 1 = display in a dialog box
- 2) Help ID no longer applicable to the FM 357-2

Notice

The associated text is specified together with the position parameters in inverted commas.

The characters " and # may not be used for message texts.

The character % is reserved for the display of the parameters.

3. Quit the editor.
4. Call the file **hpusetup.bat**, which is to be found under the directory **C:\hpu_dvk**.
Follow the statements displayed by the installation program step by step.

13.3.4 Configuring alarm and message texts for HT 6

Prerequisite

- PC/PG with “MS DOS” (6.x or higher) and “Windows”.
- PC/PG and HT 6 are connected using a V.24 (zero modem) cable.
- HT 6 with communication to a ready FM 357-2.
- PCIN is installed on your PC/PG (programming device).

PCIN is a program for transmitting and receiving user data via the serial (V.24) interface

Amending alarm and message texts

To edit the alarm and message texts, proceed as follows:

1. Start the “PCIN” tool from your PC/PG.
2. On the HT 6, choose the area **Services > Data out**. The dialog box **Select data** will appear.
3. Choose **Texts** from the **Name:** block. Press the Input key (yellow) to open the **Texts** directory.
4. Select **Language 1** and click **Start** to transfer them to the PC/PG. Then select **language 2** and transfer this also to the PC/PG.
5. After successful transfer to the PC/PG, you can amend these files on the PC/PG by using the “PCIN” tool or another ASCII editor.
6. Edit the alarm and text files **language1.txt** or **language2.txt** in the desired 1st and 2nd display languages.

Note:

Make sure that number and order of the alarm and message texts (number of lines) of the selected languages coincide.

The following numbers are available for the alarm and message texts: 700000 to 700463 (see Table 13-11).

Format of the language file language1.txt or language2.txt for the alarm and message texts

The file is structured as follows:

%E:\ALPUTX.TO!				
//CP = 1252				
//ANSI = YES				
700000	0	0		"User alarm text"
...				...
700032	0	0		"User message text"
// language1.txt				

Column 1 – message number

Column 2 – display; 0 = display in the alarm line; 1 = display in a dialog box

Column 3 – help ID does not apply to FM 357-2 (always write 0)

Column 5 – text on HT 6

Notice

The associated text is specified together with the position parameters in inverted commas.

The characters “ and # may not be used for message texts.

The character % is reserved for the display of the parameters.

7. Retransfer the files you have edited back to the HT 6. To do so, select the area **Services > Data in** on the HT 6 and click **Start**.

On your PC/PG, use the “PCIN” tool to select the files you want to transfer and start the transfer.

13.3.5 Diagnostic buffer of the CPU

General

Diagnostic information of the CPU operating system (can be read using STE 7) is entered in the diagnostic buffer of the CPU. Entries in the diagnostic buffer are made by the FC 20.

Below you will find some notes with regard to the interpretation of the user alarms/messages in the diagnostic buffer. These messages do not have an explaining text as the diagnostic buffer entry; they only consist of an event ID and supplementary information.

User alarms/messages

First, the event ID must be interpreted in the diagnostic buffer. The basic program creates the following number combinations in the first two decades of the event ID:

- A1: Alarm set
- A0: Alarm cleared
- B1: Message set
- B0: Message cleared

The following two data of the event ID correspond to the message number. This two-decade number of the event ID must be converted by the displayed hexadecimal value to a decimal value.

In addition, the field "Suppl. information 1/2/3" must also be evaluated; only the last two decades of suppl. information 1 and the last two decades of suppl. information 3 are valid pieces of information to the message number. All numbers contained in supplementary information 1/2/3 are hexadecimal values and must be converted to decimal values for composing the message number.

Example:

Display of the diagnostic buffer of the CPU

Event ID = 16#A146

Supplementary information 1/2/3 = 4900 0000 0002

Evaluation

A1	= set alarm
46 (hex)	= 70 (dec)
from suppl. info 1	= 00
from suppl. info 3	= 02

Result

User alarm: 700.002

13.4 Changing the global data communication (SDB 210)

Overview

if you have already set up such a communication in your project and now additionally want to set up the communication for the PHU/HT 6, you can view the original configuration in the project "GDSDB210". This archived project is contained on the supplied CD in the directory "GD_Proj".

Procedure

Dearchive the project and open the global data definition (to do so, select the project name in the project GDSDB210 from the SIMATIC manager, select MPI network; definition via the menu commands **Tools > Global data**).

Since a PHU/HT 6 is not contained in the hardware selection, a GD communication can only be created using a virtual CPU. The global data configuration contains two stations which exchange documents with each other. These stations are the CPU (CPU 314) and the PHU/HT 6 (CPU 314). The MPI address of the CPU 314 station is "2". The station PHU/HT 6 possesses the MPI address "14" (CPU-MPI address).

From the view of the CPU stations, the global data exchange is carried out writing using the user data block (user DB 21, DBB2 with 8 bytes (GD 1.1.1) and reading via the range user DB 21, DBB16 also with 8 bytes (GD 1.2.1). The global data status (GDS 1.2) is to be found in user DB 21, DBD12.

Do **not** change these data; otherwise, the data exchange between PHU/HT 6 interface (user DB 21) and user interfaces (user DB "FMx" and AW-DB "AXy") will no longer function.

After saving and compiling the configuration, an SDB 210 has been created with each station in the "Blocks" container of the SIMATIC manager. Then copy the system data from the CPU station, "Blocks" container into the appropriate user project. The system data of the PHU/HT 6 station are no longer needed, since the GD configuration is parameterized in the PHU/HT 6 with fixed values.

13.5 Functions

General

The function description is based on the example of a robot.

Axis identifiers in the coordinate systems

The abbreviations used in the diagrams and descriptions are listed below:

- $X_{TCS}, Y_{TCS}, Z_{TCS}$ – Tool coordinate system
- $X_{WCS}, Y_{WCS}, Z_{WCS}$ – Workpiece coordinate system (WCS)
- X_{RO}, Y_{RO}, Z_{RO} – Basic (robot), foot point coordinate system (BCS)
- $A1, A2, A3$ – Machine coordinate system (MCS)
- X_{FL}, Y_{FL}, Z_{FL} – Flange coordinate system
- X_{HP}, Y_{HP}, Z_{HP} – Hand point coordinate system
- $X_{IRO}, Y_{IRO}, Z_{IRO}$ – Internal robot coordinate system
- X_3, Y_3, Z_3 – Coordinates of last hand axis
- p_3, q_3, r_3 – Coordinates of last basic axis

13.5.1 Definition of terms

Units and directions

All lengths in the transformation machine data are specified in millimeters or inches and, unless stated otherwise, all angles are specified in degrees with an interval of $[-180^\circ, 180^\circ]$; RPY angle B has an interval of $[-90^\circ, +90^\circ]$.

The angles in the drawings are always accompanied by arrows indicating the positive mathematical direction of rotation.

Position and orientation description using frames

In order to distinguish the term “frame” from the usage defined in the NC language, the following description explains the meaning of the term “frame” in relation to “handling transformation”.

Frame

A frame can be used to transform one coordinate system to another. A distinction is made here between translation and rotation. Translation shifts the coordinate system, while rotation rotates the coordinate system in relation to the reference system. Frames are description in Section 10.3.

Translation

Coordinates X, Y and Z are used as identifiers for the translation. They are defined such that the coordinate system is rectangular.

The translation is always specified with reference to the coordinate directions of the initial system. The directions are assigned to the machine data as follows:

- X direction: ..._POS[0]
- Y direction: ..._POS[1]
- Z direction: ..._POS[2]

Rotation

RPY angles A, B and C are used for rotation (RPY stands for Roll Pitch Yaw). The positive direction of rotation is defined by the right-hand rule, i.e. when the thumb of the right hand is pointing in the direction of the rotary axis, the other fingers indicate the positive angle direction. It should be noted that A and C are defined with the interval [-180; +180] and B with the interval [-90; +90].

The RPY angles are defined as follows:

- Angle A: 1st rotation around Z axis of initial system
- Angle B: 2nd rotation around rotated Y axis
- Angle C: 3rd rotation around doubly rotated X axis

The RPY angles are assigned to the machine data as follows:

- Angle A: ..._RPY[0]
- Angle B: ..._RPY[1]
- Angle C: ..._RPY[2]

The figure 13-4 shows an example of a rotation through the RPY angles. In this example, the initial coordinate system X₁, Y₁, Z₁, is first rotated through angle A around axis Z₁; then through angle B around axis Y₂; and finally through angle C around axis X₃.

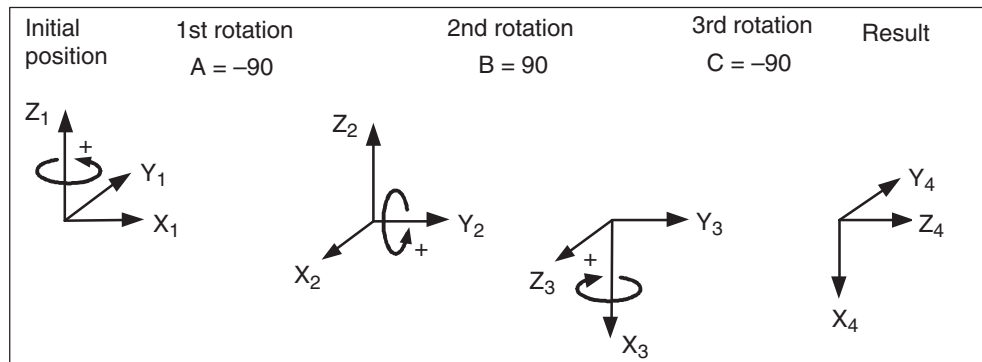


Fig. 13-4 Example of rotation through the RPY angles

Articulated joint definition

An articulated joint either a translation or rotation axis.

The basic axis identifiers are determined by the arrangement and order of the individual joints. These are specified by the letters (S, C, R, N), which are explained below.

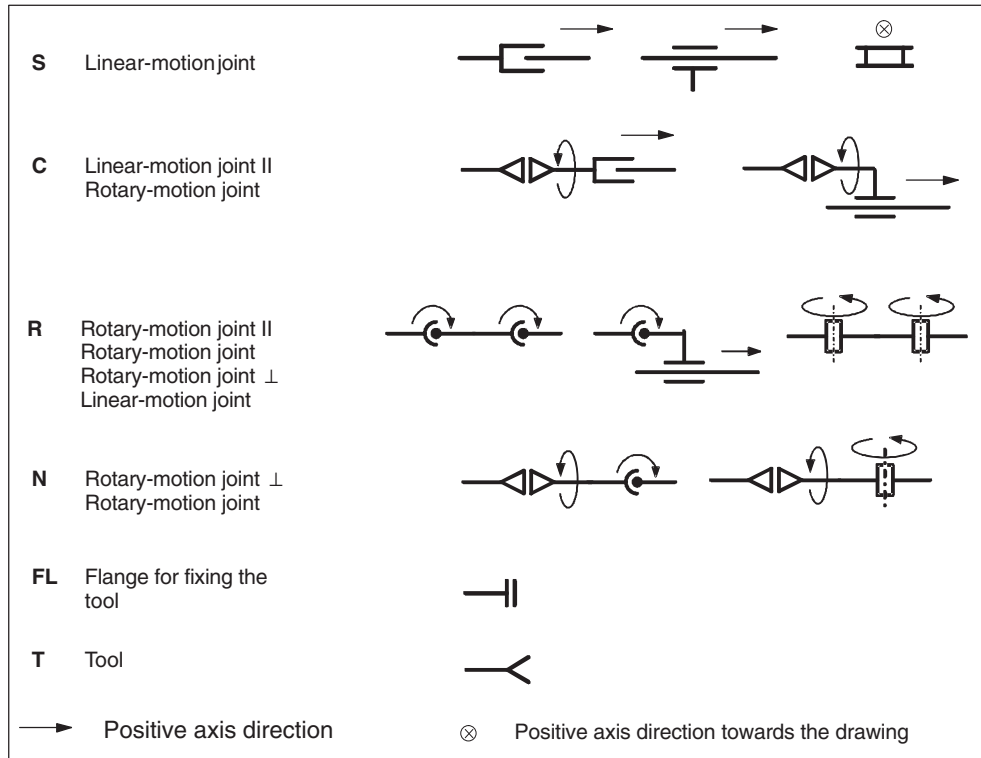


Fig. 13-5 Articulated joint names

13.5.2 Configuring the kinematic transformation

General

The system requires certain information about the mechanical configuration of the machine to enable the kinematic transformation to convert the programmed values into axis movements. The following information is stored in machine data during the startup phase:

- Axis assignments
- Geometry information

The configuration (parameterization) of the machine geometry is based on a type of building block principle. The machine is configured from its foot to the tool tip using geometry parameters, resulting in a closed kinematic sequence. Frames (see Section 13.5.1) are used here to describe the geometry. When the control is powered up, the machine data are verified, and an error message is displayed if a configuration error is detected.

As you can see in 13-6, the kinematic transformation converts the working point of the tool (tool coordinate system: X_{TCS} , Y_{TCS} , Z_{TCS}), which is specified with reference to the basic coordinate system (BCS: X_{RO} , Y_{RO} , Z_{RO}), into machine axis values (MCS positions: $A1$, $A2$, $A3$, ...). The working point (X_{TCS} , Y_{TCS} , Z_{TCS}) is specified in the NC program with reference to the workpiece to be machined (WCS: X_{WCS} , Y_{WCS} , Z_{WCS}). The programmable frames can be used to shift the WCS with reference to the BCS.

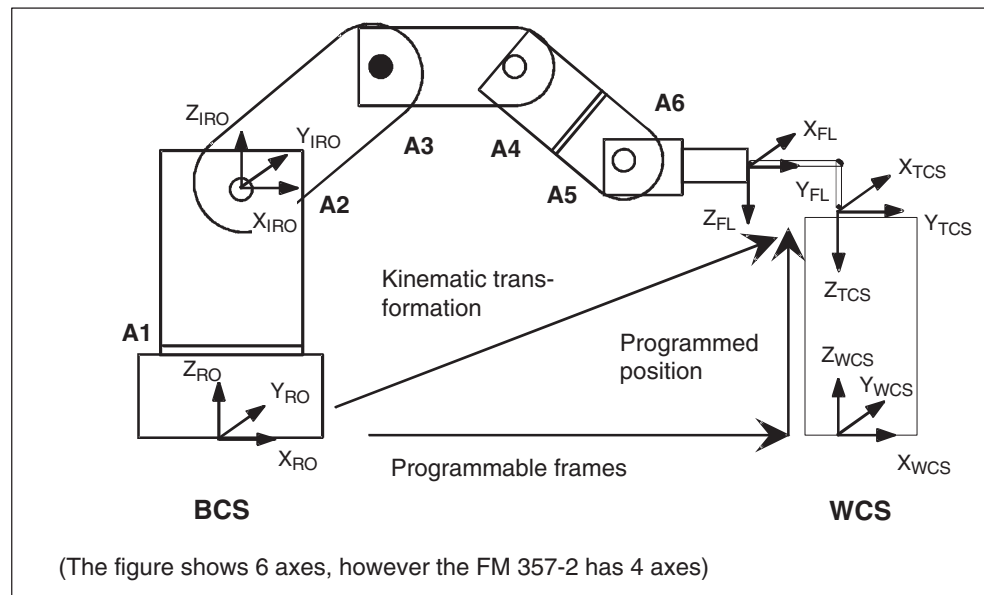


Fig. 13-6 Robot example for kinematic sequence

Overview of machine data for kinematic transformation

The following machine data are required in order to configure the kinematic transformation:

- Basic axes
- Hand axis
- Connection frames
- Further configuration data

Basic axes

The basic axes are usually the first 3 axes involved in the transformation. They must always be positioned parallel or perpendicular to each other. Each of the following basic axis arrangements is assigned an identifier (see Section 13.5.1, Articulated joint definition).

Parameters	Value range/meaning	Unit
Basic axis identifier	1: SS (gantry) (default value) 2: CC (SCARA) 3: NR (articulated arm) 4: SC (SCARA) 5: RR (articulated arm) 6: CS (SCARA) 7: NN (articulated arm)	–

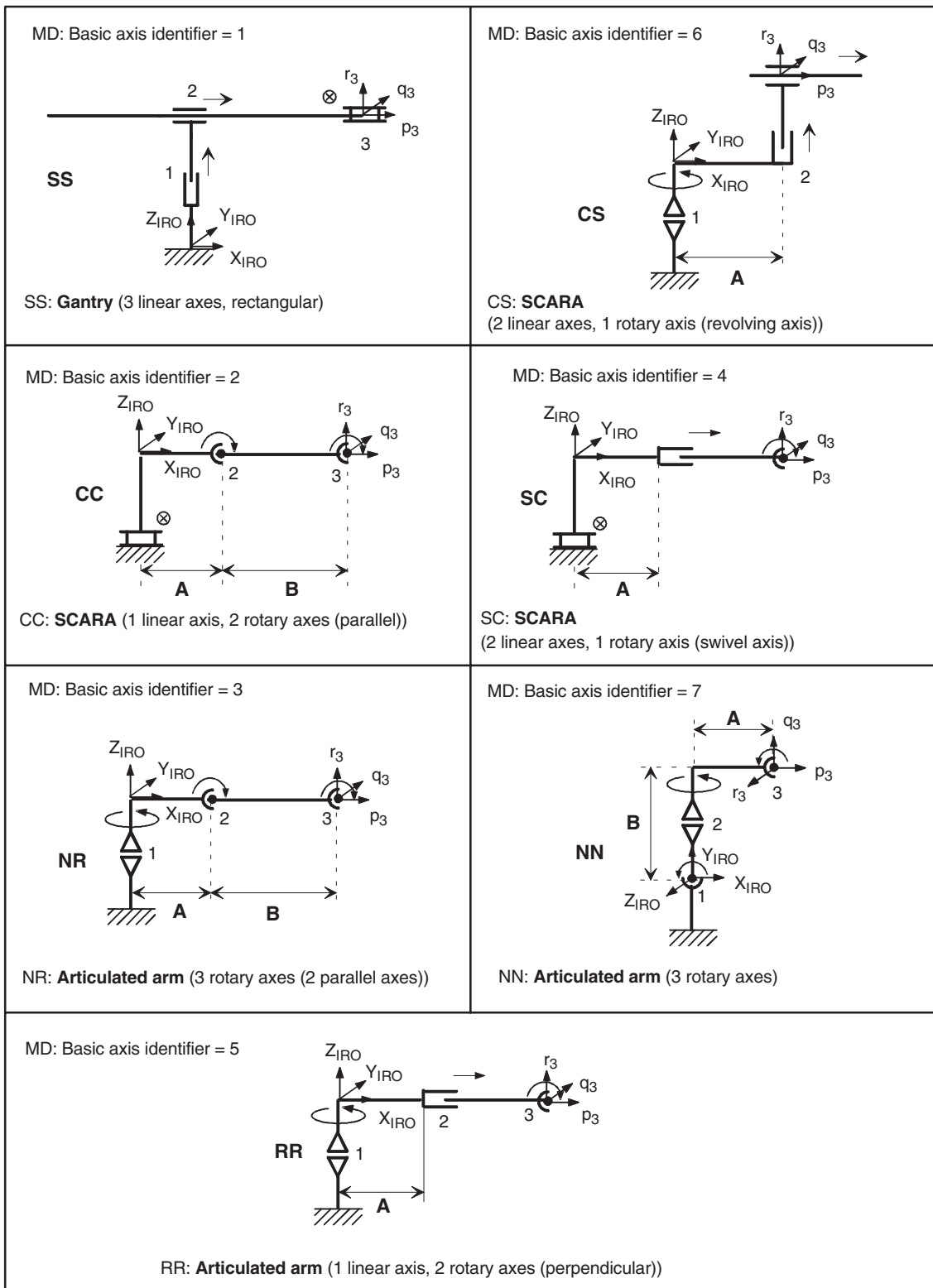


Fig. 13-7 Overview of basic axis configurations contained in handling transformation package

Hand axis

Two hand axes are required for a hand. These axes are required in addition to the basic axes. FM357-2 has 4 axes and thus does not have a complete hand, but up to one single hand axis.

Connection frames

The following connection frames are possible:

- T_IRO_RO (frame between foot point and internal coordinate system)
- T_X3_P3 (attachement of the hand)
- T_FL_WP (frame between hand point and flange coordinate system)

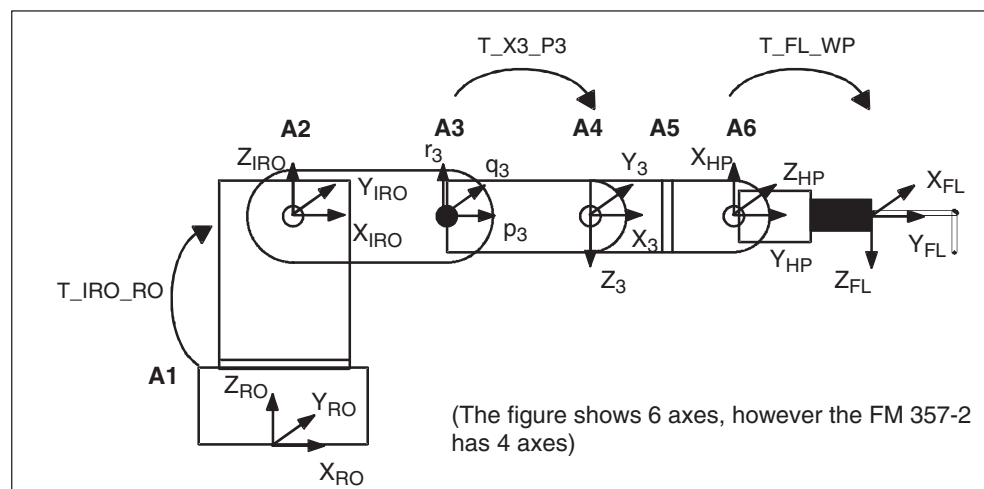


Fig. 13-8 Connection frames

T_IRO_RO

Frame T_IRO_RO connects the foot point coordinate system (RO) defined by the user with the internal robot coordinate system (IRO). The internal robot coordinate system is permanently defined by "Handling transformation" for each type of basic axis and is sketched into the kinematic diagrams for the basic axis arrangements. The foot point coordinate system is located at the Cartesian machine zero. It is equivalent to the basic coordinate system. If no frames are programmed, the basic coordinate system is equivalent to the workpiece coordinate system.

Notice

The first rotary axis must always be parallel/antiparallel to one of the coordinate axes of the foot point coordinate system (Z_{RO}).

There are no further restrictions for basic axis types CC, CS and SC if axis 4 is parallel to the last rotary basic axis.

For all other basic axes and for basic axis types CC, CS and SC where axis 4 is perpendicular to the last rotary basic axis, the Z axis of the foot point coordinate system (Z_{RO}) must be parallel to the Z axis of the internal robot coordinate system (Z_{IRO}).

T_X3_P3

Frame T_X3_P3 describes the relationship between the hand and the basic axes. Frame T_X3_P3 connects the coordinate system of the last basic axis ($p_3-q_3-r_3$ coordinate system) to the coordinate system applied to the first hand axis ($X_3-Y_3-Z_3$ coordinate system). The $p_3-q_3-r_3$ coordinate system is sketched into the kinematic diagrams for the basic axis arrangements.

The Z_3 axis is always located on the 4th axis.

In the case of **4-axis** systems, it should be noted that the Z_3 axis is always parallel/antiparallel or perpendicular to the last basic axis.

Frame T_X3_P3 is calculated for 4-axis systems. The origin of the $X_3-Y_3-Z_3$ coordinate system coincides with $X_{HP}-Y_{HP}-Z_{HP}$ of the hand point coordinate system.

In the case of **3-axis** systems, the origins of the coordinate systems $p_3-q_3-r_3$, $X_3-Y_3-Z_3$ and $X_{HP}-Y_{HP}-Z_{HP}$ coincide.

T_FL_WP

Frame T_FL_WP connects the flange coordinate system ($X_{FL}-Y_{FL}-Z_{FL}$) to the last internal $X_{HP}-Y_{HP}-Z_{HP}$ coordinate system (hand point coordinate system) defined permanently by "Handling Transformation".

More detailed information on frame T_FL_WP for 4-axis systems is contained in the section "4-axis kinematics".

Parameters	Value range/meaning	Unit
Attachment of hand (position component) [n] n = 0...2	n = 0: X component n = 1: Y component n = 2: Z component [0.0, 0.0, 0.0] = default value	[mm], [inches]
Attachment of hand (rotation component) [n] n = 0...2	n = 0: Rotation through RPY angle A n = 1: Rotation through RPY angle B n = 2: Rotation through RPY angle C [0.0, 0.0, 0.0] = default value	[de- grees]
Frame between hand point and flange coordinate system (position component) [n] n = 0...2	n = 0: X component n = 1: Y component n = 2: Z component [0.0, 0.0, 0.0] = default value	[mm], [inches]
Frame between hand point and flange coordinate system (rotation component) [n] n = 0...2	n = 0: Rotation through RPY angle A n = 1: Rotation through RPY angle B n = 2: Rotation through RPY angle C [0.0, 0.0, 0.0] = default value	[de- grees]
Frame between foot point and internal coordinate system (position component) [n] n = 0...2	n = 0: X component n = 1: Y component n = 2: Z component [0.0, 0.0, 0.0] = default value	[mm], [inches]
Frame between foot point and internal coordinate system (rotation component) [n] n = 0...2	n = 0: Rotation through RPY angle A n = 1: Rotation through RPY angle B n = 2: Rotation through RPY angle C [0.0, 0.0, 0.0] = default value	[de- grees]

Further configuration data

Machine data: Number of transformed axes

This machine data defines the number of axes involved in the transformation. The number of transformed axes can be between 2 and 4.

Machine data: Reorder axes (change the order of the axes) [n]

Notice

In certain kinematic sequences it is possible to interchange the axes without producing a different kinematic response. MD "Reorder axes" is available for such kinematic transitions. The axes on the machine are numbered from 1 to 4 and must be entered in their internal order in MD "Reorder axes[0]...[3]".

Table 13-12 Changing the axis order

Basic axis kinematics	Interchange options
SS, CC	Any
CS	2 and 3
SC	1 and 2

Example 1

Two kinematic sequences are used (in Fig. 13-9). Kinematic sequence 1 is contained directly in the "handling transformation". It is equivalent to a CC kinematic sequence with a hand axis parallel to the last rotary basic axis. Kinematic sequence 2 is equivalent to kinematic 1, since it is irrelevant for the resulting robot movement whether the translational axis is axis 1 or axis 4. In this case, the data for kinematic sequence 2 must be entered as follows in MD "Reorder axes":

- Reorder axes[0] = 4
- Reorder axes[1] = 1
- Reorder axes[2] = 2
- Reorder axes[3] = 3

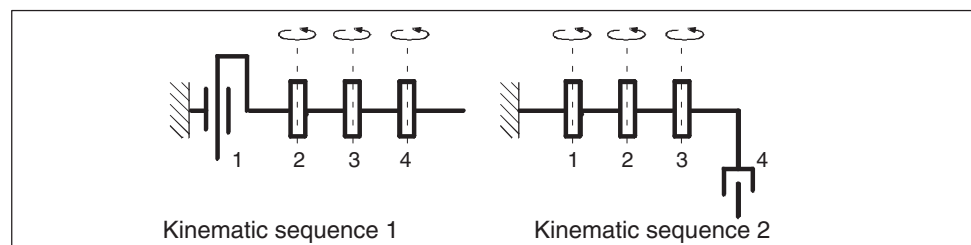


Fig. 13-9 Changing the axis order (example 1)

Example 2

In the SCARA kinematic configuration shown in Fig. 13-10, the axes can be interchanged as desired. Kinematic sequence 1 is contained directly in the “handling transformation”. It is equivalent to a CC kinematic sequence. It is relevant for axis interchange how many hand axes are involved.

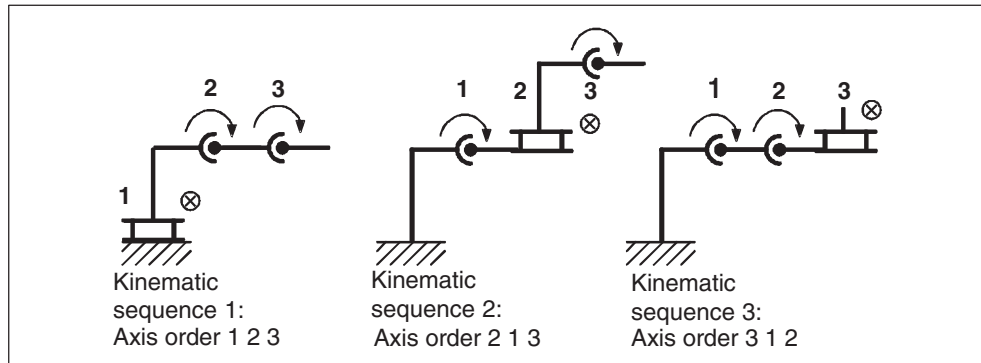


Fig. 13-10 Changing the axis order (example 2)

Machine data: Adapt physical and mathematical direction of rotation (change axis direction) [axis no.]

A fixed direction of rotation or linear motion is defined for each axis in “Handling transformation”. This direction may not match the direction on the machine. The directions can be matched in MD “Adapt physical and mathematical direction of rotation[]” by entering **-1** for each axis if the direction is to be inverted or **+1** if the directions already match up.

Machine data: Offset between mathematical and mechanical zero (adapt zero points of axes) [axis no.]

The mathematical zero points of the axes are permanently defined in “Handling transformation”. However, the mathematical zero point may differ from the mechanical zero point (alignment position) of the axes. The zero points can be matched in MD “Offset between mathematical and mechanical zero[]” by entering the offset between the mathematical zero and the alignment point for each axis. The difference must be entered starting at the mechanical zero point with reference to the positive mathematical direction of rotation of the axis.

Example

The example (see Fig. 13-11) shows an articulated arm kinematic sequence. The mathematical zero point of axis 2 is at 90° . This value must be entered for axis 2 in MD "Offset between mathematical and mechanical zero[1]". Axis 3 is counted relative to axis 2 and thus has a value of -90° at the mathematical zero point.

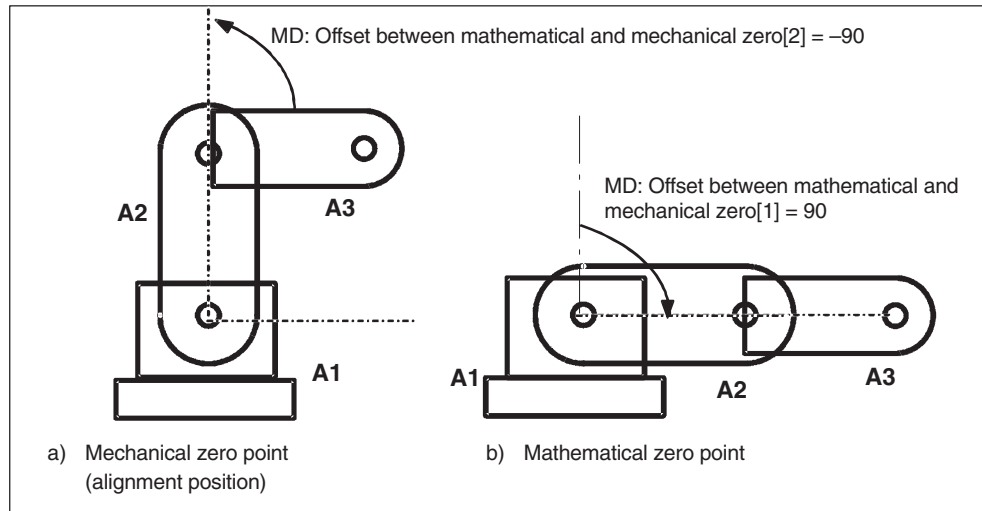


Fig. 13-11 Adapt to mechanical zero point

Machine data: Axis type for transformation [axis no.]

This MD specifies the axis type. The following types of axis are used in transformations:

- Linear axis
- Rotary axis

Machine data: Cartesian velocities [n]

This MD can be used to define the velocities for the individual translational directions of movements when traversing with G0.

Machine data: Cartesian acceleration [n]

This MD can be used to define the acceleration rates for the individual translational directions of movements when traversing with G0.

Machine data: Orientation angle velocity

This MD can be used to define the velocity for the orientation direction when traversing with G0.

Machine data: Orientation angle acceleration

This MD can be used to define the acceleration for the orientation direction when traversing with G0.

Parameters	Value range/meaning	Unit
Number of transformed axes	1...4 (3 = default value)	–
Reordering of axes	1...4 [1, 2, 3, 4] = default value	–
Adapt physical and mathematical direction of rotation [axis no.] Axis no.: 0...3	+1: Same direction of rotation –1: Different direction of rotation [1, 1, 1, 1] = default value	–
Offset between mathematical and mechanical zero [axis no.] Axis no.: 0...3	rotary axes [degrees] linear axes [mm], [inch] [0.0, 0.0, 0.0, 0.0] = default value	–
Axis type for transformation [axis no.] Axis no.: 0...3	1: Linear axis 3: Rotary axis [1, 1, 1, 3] = default value	–
Cartesian velocities [n] No.: 0...2	n = 0: Velocity in X direction n = 1: Velocity in Y direction n = 2: Velocity in Z direction [600000.0, 600000.0, 600000.0] = default	[mm/min], [inch/min]
Cartesian accelerations [n] No.: 0...2	n = 0: Acceleration in X direction n = 1: Acceleration in Y direction n = 2: Acceleration in Z direction [0.5, 0.5, 0.5] = default value	[m/s ²]
Orientation angle velocity	Velocity, angle A [1.6667] = default value	[rev/min]
Orientation angle acceleration	Acceleration, angle A [0.0028] = default value	[rev/s ²]

13.5.3 Kinematic descriptions

General

The following kinematic descriptions for 3 and 4-axis kinematic sequences first describe the general procedure for configuring and then explain how to configure the machine data for each type of kinematic sequence, using an example configuration for reference purposes. Not all of the possible lengths and offsets are depicted in these examples. The direction parameters refer to the positive directions of linear and rotary movement for the transformation. The axis positions correspond to the zero axis positions for each transformation.

3-axis kinematics

3-axis kinematic sequences have 3 translational degrees of freedom. They have no degree of freedom for the orientation. That means they only have basic axes.

Configuration

1. Enter kinematic class "Standard" in MD "Kinematic class".
2. Enter the number of axes for the transformation in MD "Number of transformed axes = 3".
3. Compare the basic axes with those contained in the "handling transformation". Enter the basic axis identifier in MD "Basic axis identifier".
4. If the axis order differs from the normal axis order, you must correct the order in MD "Reorder axes".
5. Enter the axis types for the transformation in MD "Axis type for transformation".
6. Compare the directions of rotation of the axes with the directions of rotation defined by "handling transformation" and match the directions in MD "Adapt physical and mathematical direction of rotation".
7. Enter the mechanical zero offset in MD "Offset between mathematical and mechanical zero".
8. Enter the basic axis lengths in MD "Basic axis lengths A and B".
9. Define frame T_IRO_RO and enter the offset in MD "Frame between foot point and internal coordinate system (position component)" and enter the rotation in MD "Frame between foot point and internal coordinate system (rotation component)".
10. Define the flange coordinate system. The $X_{HP_Y_{HP_Z_{HP}}$ coordinate system should be regarded as the initial system for this purpose. The offset is entered in MD "Frame between hand point and flange coordinate system (position component)" and the rotation is entered in MD "Frame between hand point and flange coordinate system (rotation component)".

3-axis Gantry kinematics

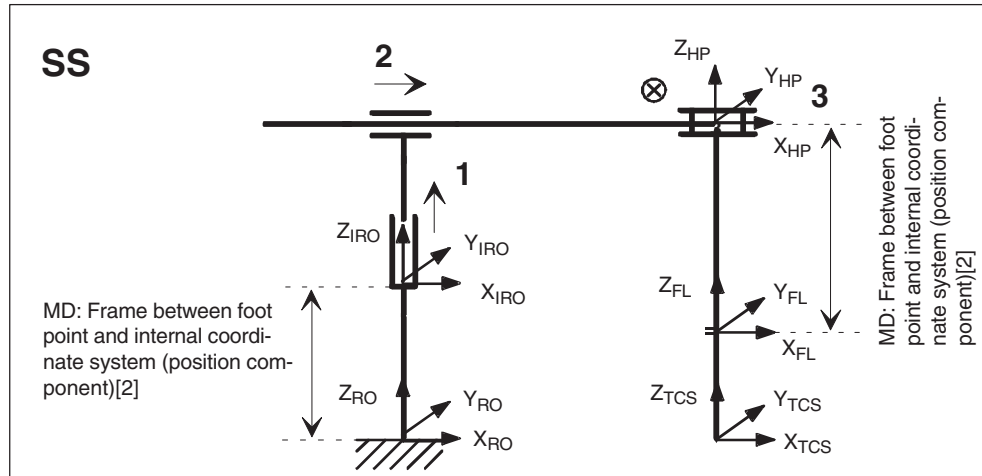


Fig. 13-12 3-axis SS kinematics

Table 13-13 Configuration data for 3-axis SS kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	3
Basic axis identifier	1
Axis type for transformation	[1, 1, 1]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[0.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 0.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (position component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

3-axis SCARA kinematics

SCARA kinematic sequences use both translational and rotary axes. They are subdivided into 3 groups, according to the relationship between the positions of the basic axes.

- CC types
- CS types
- SC types (see Fig. 13-7)

CC types

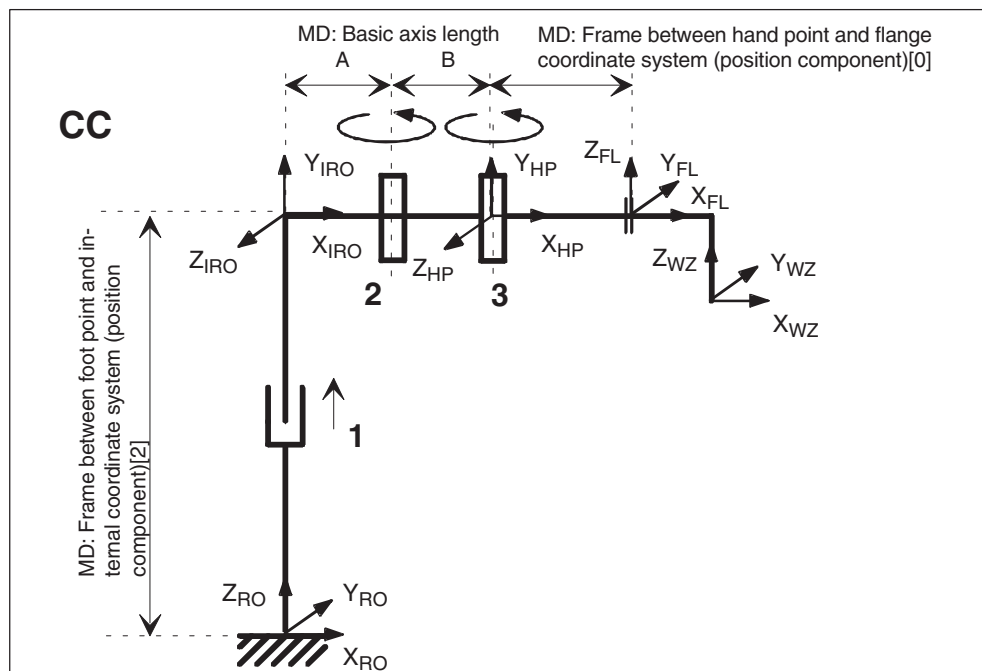


Fig. 13-13 3-axis CC kinematics

Table 13-14 Configuration data for 3-axis CC kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	3
Basic axis identifier	2
Axis type for transformation	[1, 3, 3]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[0.0, 300.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 500.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 90.0]
Frame between hand point and flange CS (position component)	[200.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, -90.0]

CS = coordinate system

SC types

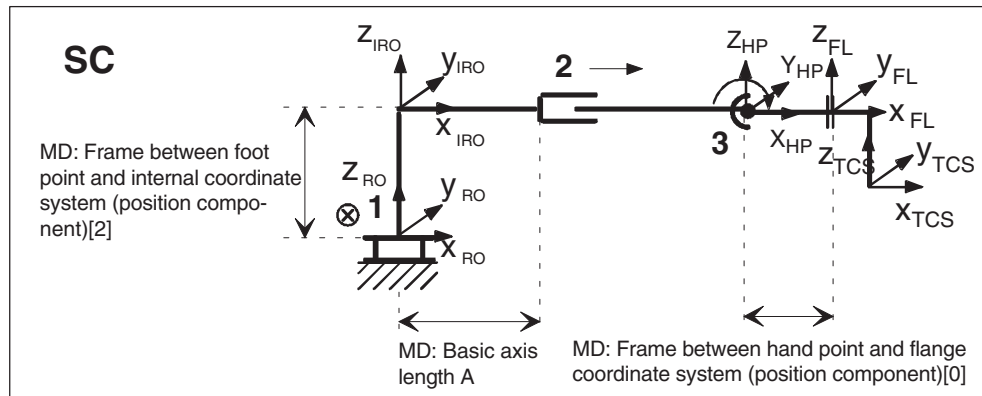


Fig. 13-14 3-axis SC kinematics

Table 13-15 Configuration data for 3-axis SC kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	3
Basic axis identifier	4
Axis type for transformation	[1, 1, 3]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[500.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 500.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (position component)	[300.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

CS types

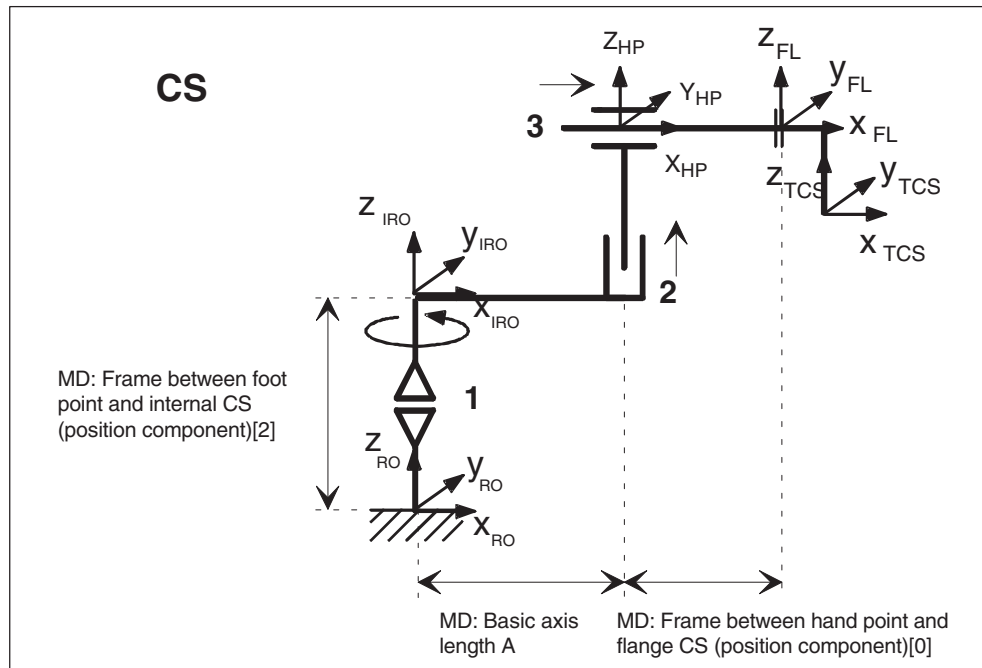


Fig. 13-15 3-axis CS kinematics

Table 13-16 Configuration data for 3-axis CS kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	3
Basic axis identifier	6
Axis type for transformation	[3, 1, 1]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[500.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 500.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (position component)	[300.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

3-axis articulated arm kinematics (NR)

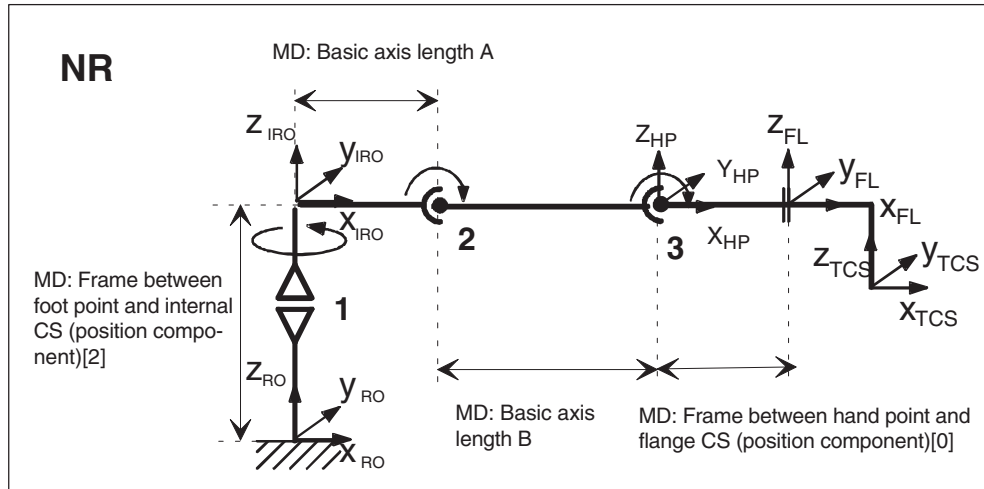


Fig. 13-16 3-axis NR kinematics

Table 13-17 Configuration data for 3-axis NR kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	3
Basic axis identifier	3
Axis type for transformation	[3, 3, 3]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[300.0, 500.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 500.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (position component)	[300.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

3-axis RR kinematics

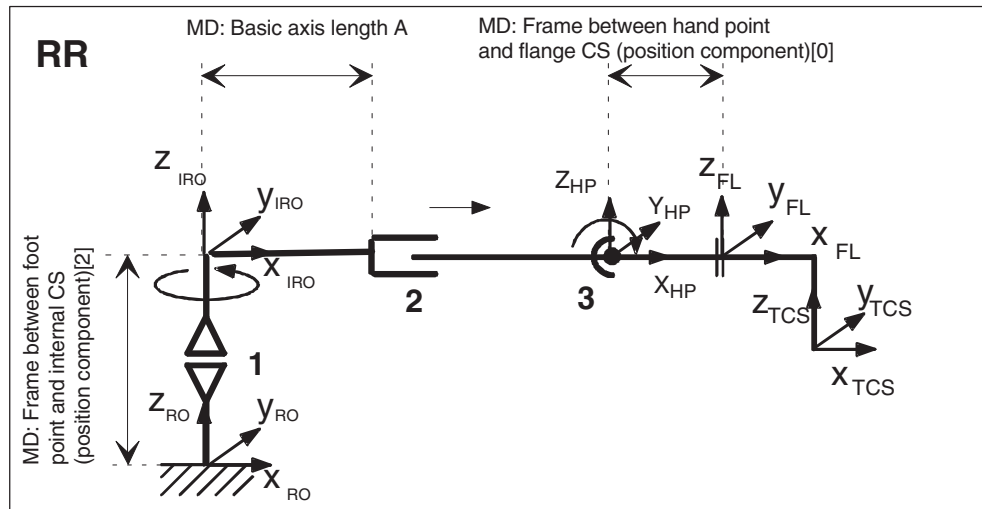


Fig. 13-17 3-axis RR kinematics

Table 13-18 Configuration data for 3-axis RR kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	3
Basic axis identifier	5
Axis type for transformation	[3, 1, 3]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[300.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 300.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (position component)	[200.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

3-axis NN kinematics

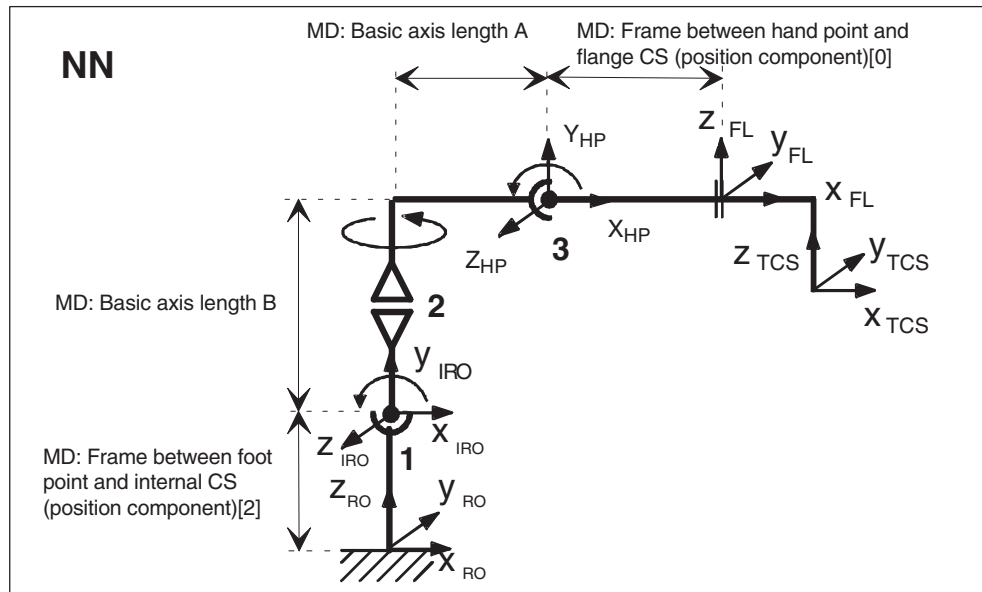


Fig. 13-18 3-axis NN kinematics

Table 13-19 Configuration data for 3-axis NN kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	3
Basic axis identifier	7
Axis type for transformation	[3, 3, 3]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[300.0, 500.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 300.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 90.0]
Frame between hand point and flange CS (position component)	[400.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, -90.0]

CS = coordinate system

4-axis Gantry kinematics

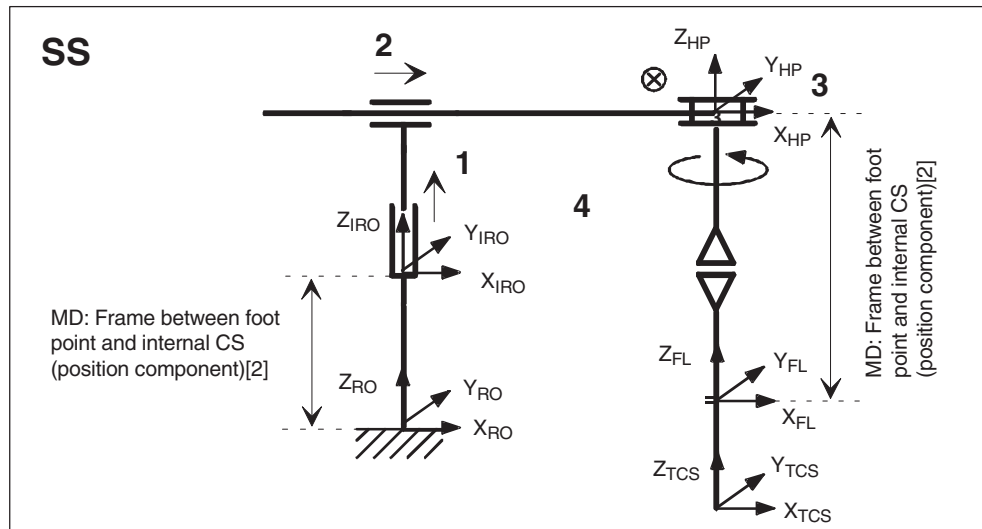


Fig. 13-19 4-axis SS kinematics

Table 13-20 Configuration data for 4-axis SS kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	4
Basic axis identifier	1
Axis 4 parallel/antiparallel to last basic axis	1
Axis type for transformation	[1, 1, 1, 3]
Reordering of axes	[1, 2, 3, 4]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[0.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 0.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (position component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

4-axis SCARA kinematics

CC types

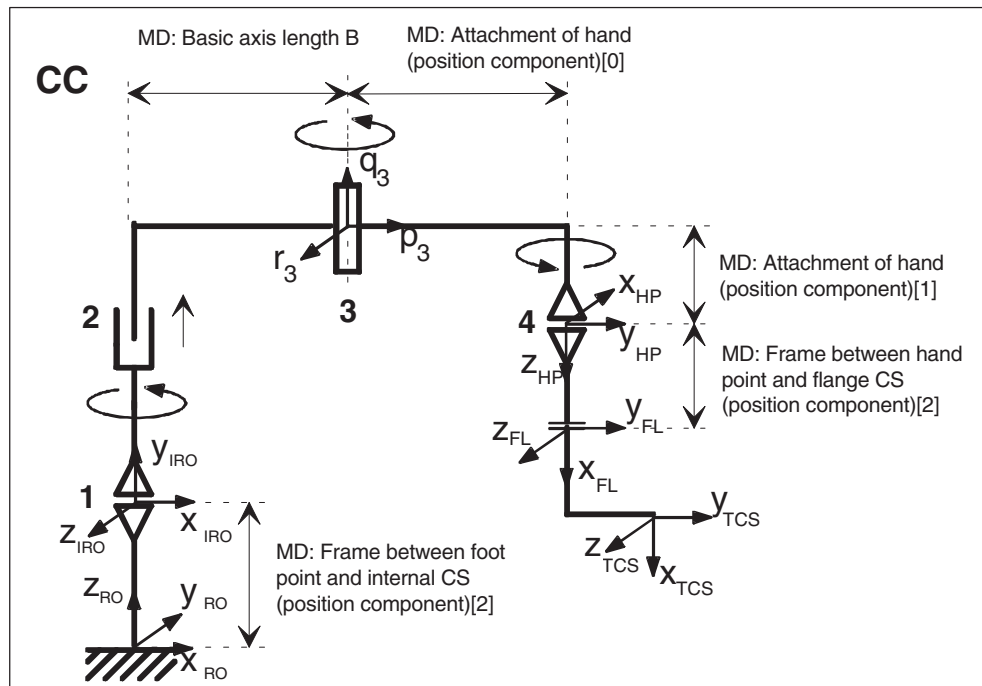


Fig. 13-20 4-axis CC kinematics

Table 13-21 Configuration data for 4-axis CC kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	4
Basic axis identifier	2
Axis 4 parallel/antiparallel to last basic axis	1
Axis type for transformation	[3, 1, 3, 3]
Reordering of axes	[2, 1, 3, 4]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0, 0.0]
Basic axis lengths A and B	[0.0, 300.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 500.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 90.0]
Attachment of hand (position component)	[300.0, 0.0, -200.0]
Attachment of hand (rotation component)	[-90.0, 90.0, 0.0]
Frame between hand point and flange CS (position component)	[0.0, 0.0, 200.0]
Frame between hand point and flange CS (rotation component)	[0.0, -90.0, 0.0]

CS = coordinate system

SC types

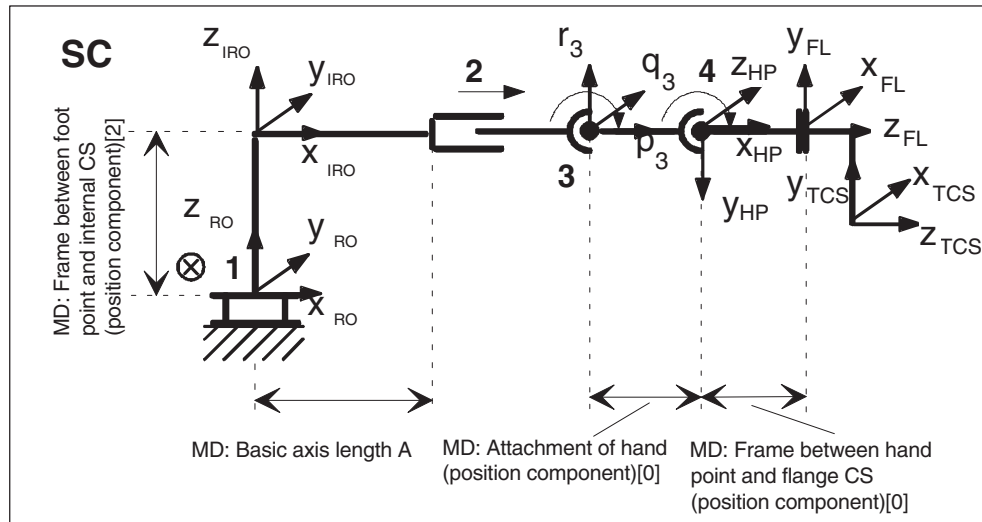


Fig. 13-21 4-axis SC kinematics

Table 13-22 Configuration data for 4-axis SC kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	4
Basic axis identifier	4
Axis 4 parallel/antiparallel to last basic axis	1
Axis type for transformation	[1, 1, 3, 3]
Reordering of axes	[1, 2, 3, 4]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0, 0.0]
Basic axis lengths A and B	[300.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 300.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Attachment of hand (position component)	[200.0, 0.0, 0.0]
Attachment of hand (rotation component)	[0.0, 0.0, -90.0]
Frame between hand point and flange CS (position component)	[200.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, -90.0, 180.0]

CS = coordinate system

CS types

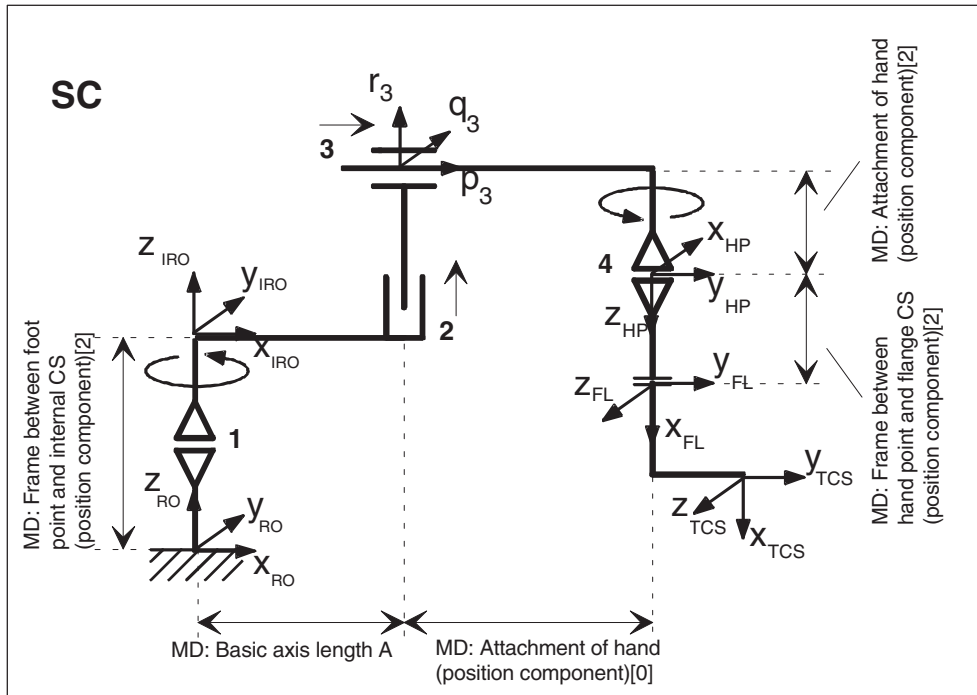


Fig. 13-22 4-axis CS kinematics

Table 13-23 Configuration data for 4-axis CS kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	4
Basic axis identifier	6
Axis 4 parallel/antiparallel to last basic axis	1
Axis type for transformation	[3, 1, 1, 3]
Reordering of axes	[1, 2, 3, 4]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0, 0.0]
Basic axis lengths A and B	[400.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 400.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Attachment of hand (position component)	[500.0, 0.0, -200.0]
Attachment of hand (rotation component)	[90.0, 0.0, 180.0]
Frame between hand point and flange CS (position component)	[200.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, -90.0, 0.0]

CS = coordinate system

4-axis articulated arm kinematics (NR)

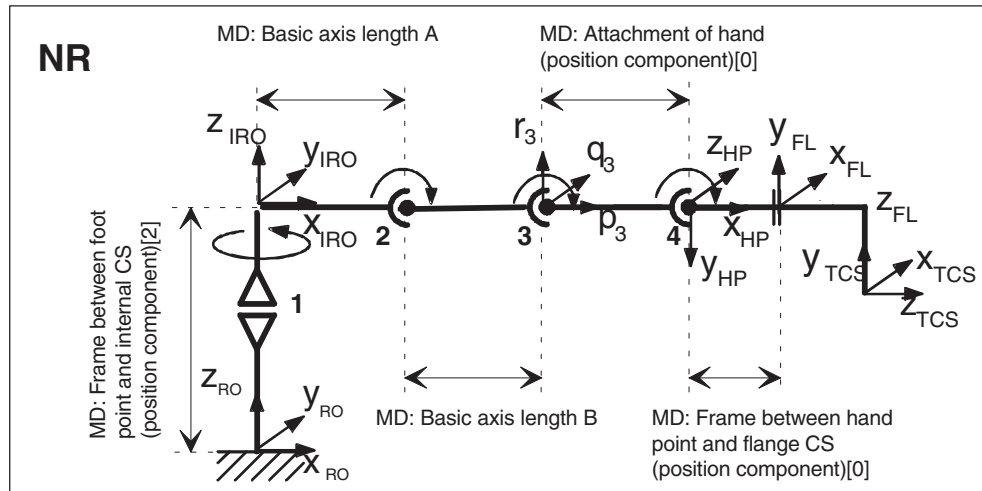


Fig. 13-23 4-axis NR kinematics

Table 13-24 Configuration data for 4-axis NR kinematics

Machine data	Value
Kinematic class	1
Number of transformed axes	4
Basic axis identifier	3
Axis 4 parallel/antiparallel to last basic axis	1
Axis type for transformation	[3, 3, 3, 3]
Reordering of axes	[1, 2, 3, 4]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0, 0.0]
Basic axis lengths A and B	[300.0, 300.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 500.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Attachment of hand (position component)	[300.0, 0.0, 0.0]
Attachment of hand (rotation component)	[0.0, 0.0, -90.0]
Frame between hand point and flange CS (position component)	[200.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, -90.0, 180.0]

CS = coordinate system

Special kinematics

Special kinematics are kinematic sequences which are not contained directly in the “Handling transformation” building block system. They are frequently characterized by a missing degree of freedom or the existence of mechanical connections between the axes or with the tool. MD “Kinematic class= 2” must be set for these types of kinematic sequences. MD “Special kinematic type” specifies the actual type of the special kinematics.

2-axis SC special kinematics

This special kinematic class is characterized by the fact that the tool is always held in the same orientation by a mechanical rod. It has 2 Cartesian degrees of freedom. This kinematic type has the identifier MD “Special kinematic type = 3”.

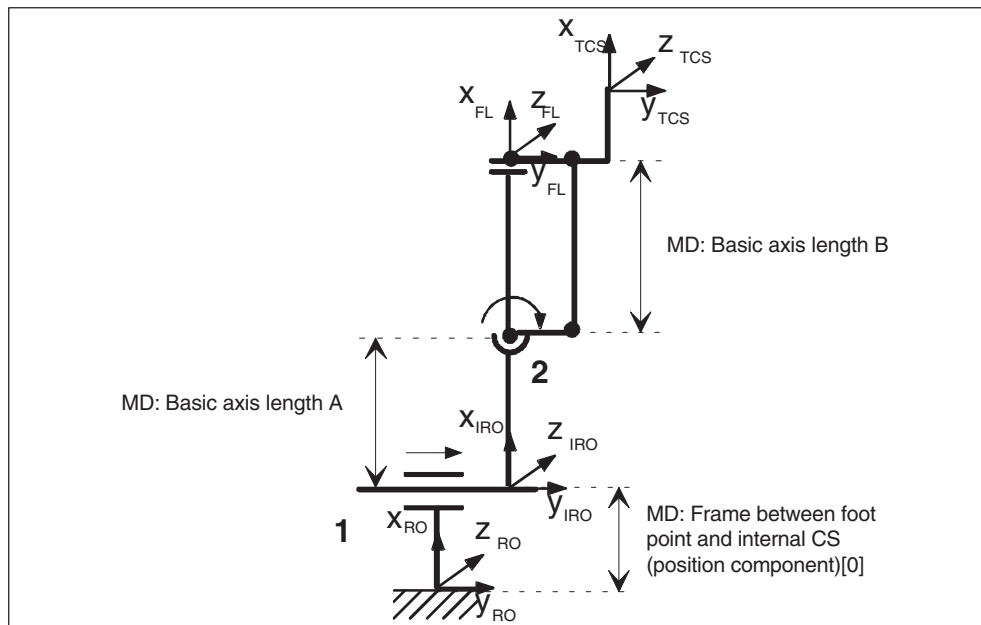


Fig. 13-24 2-axis SC special kinematics

Table 13-25 Configuration data for 2-axis SC special kinematics

Machine data	Value
Kinematic class	2
Special kinematic type	3
Number of transformed axes	2
Basic axis identifier	2
Axis type for transformation	[1, 3]
Reordering of axes	[1, 2]
Adaptation of physical and mathematical direction of rotation	[1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0]

Special kinematics

Special kinematics are kinematic sequences which are not contained directly in the “Handling transformation” building block system. They are frequently characterized by a missing degree of freedom or the existence of mechanical connections between the axes or with the tool. MD “Kinematic class= 2” must be set for these types of kinematic sequences. MD “Special kinematic type” specifies the actual type of the special kinematics.

2-axis SC special kinematics

This special kinematic class is characterized by the fact that the tool is always held in the same orientation by a mechanical rod. It has 2 Cartesian degrees of freedom. This kinematic type has the identifier MD “Special kinematic type = 3”.

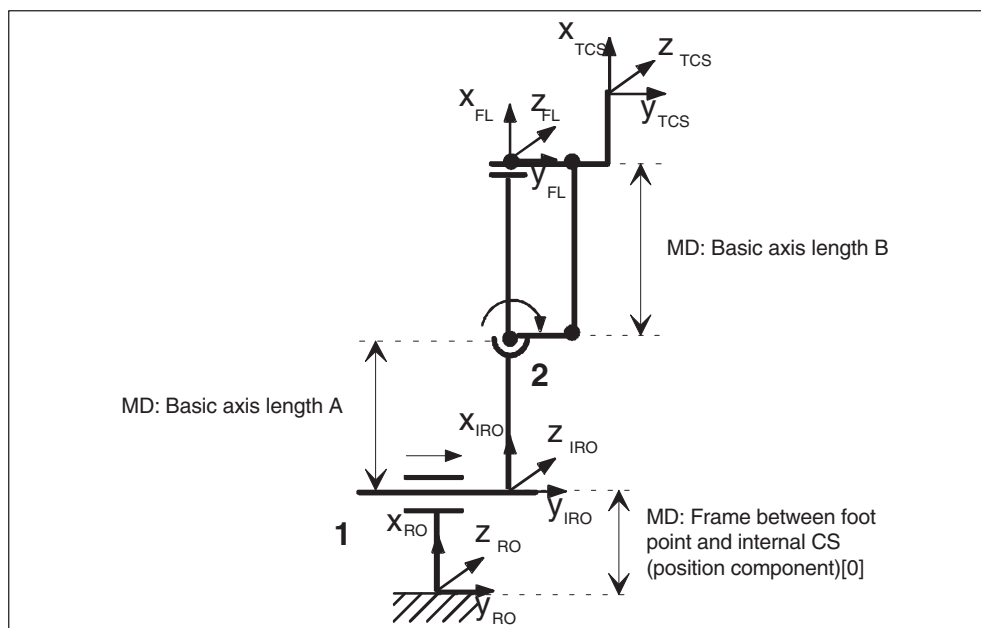


Fig. 13-24 2-axis SC special kinematics

Table 13-25 Configuration data for 2-axis SC special kinematics

Machine data	Value
Kinematic class	2
Special kinematic type	3
Number of transformed axes	2
Basic axis identifier	2
Axis type for transformation	[1, 3]
Reordering of axes	[1, 2]
Adaptation of physical and mathematical direction of rotation	[1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0]

Special kinematics

Special kinematics are kinematic sequences which are not contained directly in the “Handling transformation” building block system. They are frequently characterized by a missing degree of freedom or the existence of mechanical connections between the axes or with the tool. MD “Kinematic class= 2” must be set for these types of kinematic sequences. MD “Special kinematic type” specifies the actual type of the special kinematics.

2-axis SC special kinematics

This special kinematic class is characterized by the fact that the tool is always held in the same orientation by a mechanical rod. It has 2 Cartesian degrees of freedom. This kinematic type has the identifier MD “Special kinematic type = 3”.

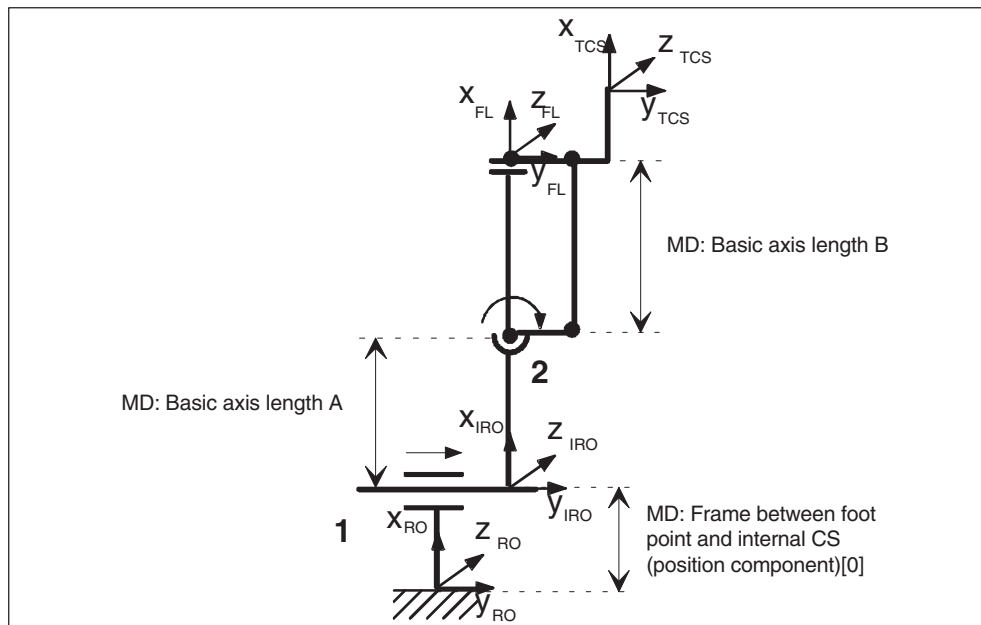


Fig. 13-24 2-axis SC special kinematics

Table 13-25 Configuration data for 2-axis SC special kinematics

Machine data	Value
Kinematic class	2
Special kinematic type	3
Number of transformed axes	2
Basic axis identifier	2
Axis type for transformation	[1, 3]
Reordering of axes	[1, 2]
Adaptation of physical and mathematical direction of rotation	[1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0]

Table 13-25 Configuration data for 2-axis SC special kinematics, continued

Machine data	Value
Basic axis lengths A and B	[400.0, 500.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 300.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

3-axis SC special kinematics

This special kinematic sequence has 2 Cartesian degrees of freedom and one degree of freedom for the orientation. It has the identifier MD "Special kinematic type = 4".

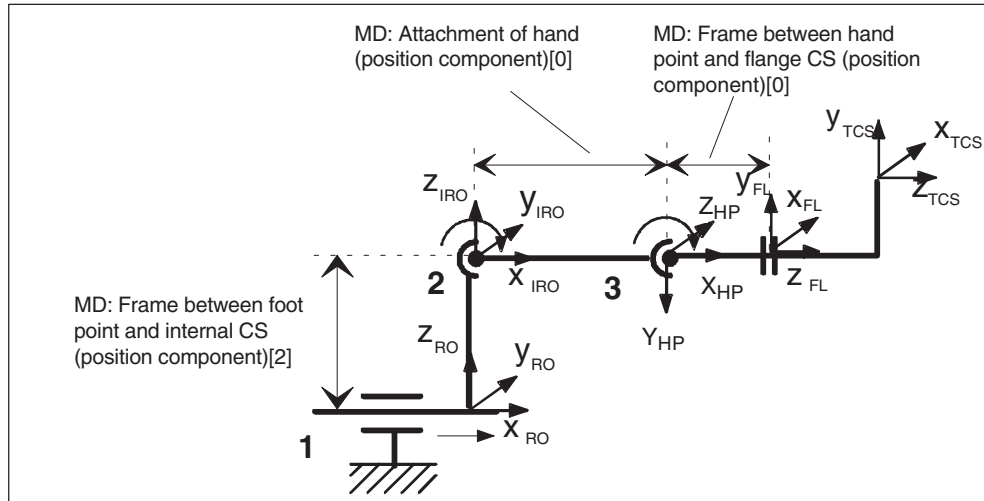


Fig. 13-25 3-axis SC special kinematics

Table 13-26 Configuration data for 3-axis SC special kinematics

Machine data	Value
Kinematic class	2
Special kinematic type	4
Number of transformed axes	3
Basic axis identifier	2
Axis type for transformation	[1, 3, 3]
Reordering of axes	[1, 2, 3]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0]
Basic axis lengths A and B	[0.0, 0.0]
Frame between foot point and internal CS (position component)	[0.0, 0.0, 400.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Attachment of hand (position component)	[400.0, 0.0, 0.0]
Attachment of hand (rotation component)	[0.0, 0.0, -90.0]
Frame between hand point and flange CS (position component)	[200.0, 0.0, 0.0]
Frame between hand point and flange CS (rotation component)	[0.0, -90.0, 180.0]

CS = coordinate system

4-axis SC special kinematics

This special kinematic class is characterized by the existence of a mechanical connection between axis 1 and axis 2. Axis 2 is always maintained at a constant angle while axis 1 is swiveled. With kinematic type, axes 3 and 4 are always maintained in a vertical position, regardless of the position of axes 1 and 2. It has the identifier MD "Special kinematic type = 7".

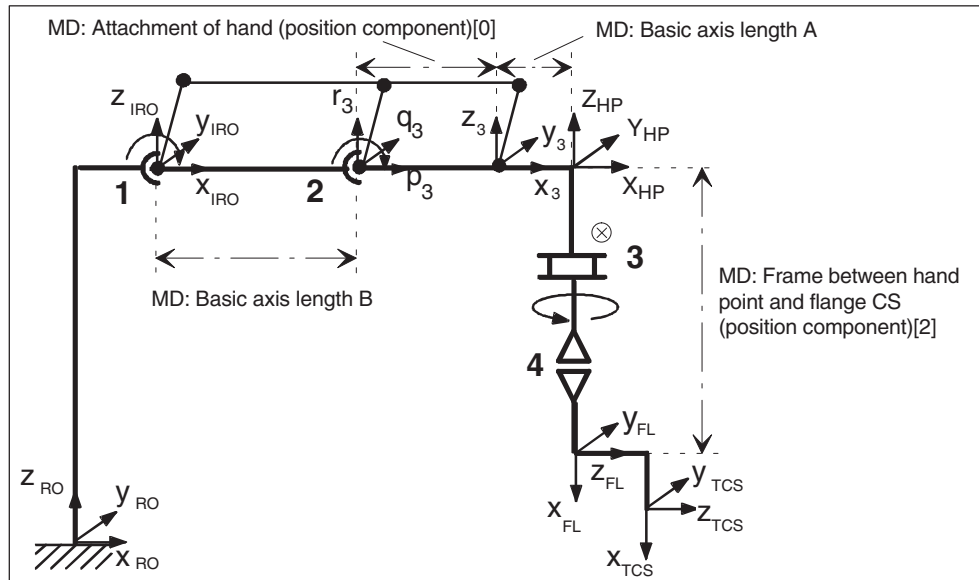


Fig. 13-26 4-axis SC special kinematics

Table 13-27 Configuration data for 4-axis SC special kinematics

Machine data	Value
Kinematic class	2
Special kinematic type	7
Number of transformed axes	4
Basic axis identifier	2
Axis type for transformation	[3, 3, 1, 3]
Reordering of axes	[1, 2, 3, 4]
Adaptation of physical and mathematical direction of rotation	[1, 1, 1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0, 0.0, 0.0]
Basic axis lengths A and B	[100.0, 400.0]
Frame between foot point and internal CS (position component)	[100.0, 0.0, 1000.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, 0.0]
Attachment of hand (position component)	[300.0, 0.0, 0.0]
Attachment of hand (rotation component)	[0.0, 0.0, 0.0]
Frame between hand point and flange CS (position component)	[0.0, 0.0, -600.0]
Frame between hand point and flange CS (rotation component)	[0.0, 90.0, 0.0]

CS = coordinate system

2-axis NR special kinematics

This special kinematic class is characterized by the existence of a mechanical connection between axis 1 and axis 2. The tool exhibits a further special characteristic. It maintains its orientation in space, independent of the position of the other axes. It has the identifier MD "Special kinematic type = 5".

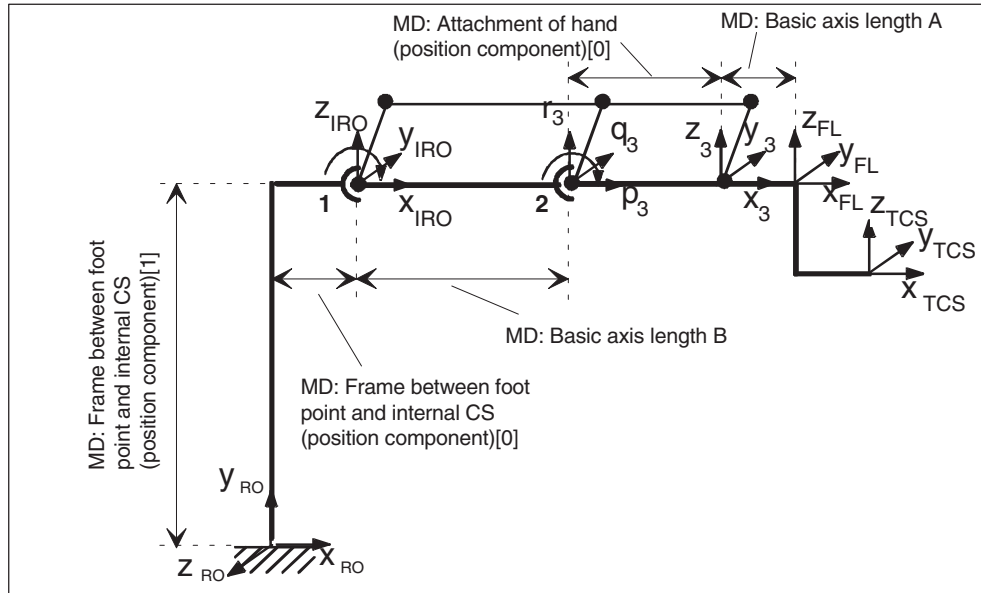


Fig. 13-27 2-axis NR special kinematics

Table 13-28 Configuration data for 2-axis NR special kinematics

Machine data	Value
Kinematic class	2
Special kinematic type	5
Number of transformed axes	2
Basic axis identifier	3
Axis type for transformation	[3, 3]
Reordering of axes	[1, 2]
Adaptation of physical and mathematical direction of rotation	[1, 1]
Offset between mathematical and mechanical zero	[0.0, 0.0]
Basic axis lengths A and B	[100.0, 400.0]
Frame between foot point and internal CS (position component)	[100.0, 500.0, 0.0]
Frame between foot point and internal CS (rotation component)	[0.0, 0.0, -90.0]
Attachment of hand (position component)	[400.0, 0.0, 0.0]
Attachment of hand (rotation component)	[0.0, 0.0, 0.0]

CS = coordinate system

13.5.4 Operating modes

General

The following modes are used for “Handling transformation”:

Table 13-29 Operating modes and their properties

Mode	Property
Jog JOG for HPU	see Section 9.12
Reference point approach Submode of “Jog”	see Section 9.12
MDI (Manual Data Input) MDA for HPU Control and checkback signals	<p>In “MDI (MDA)” mode, you can create and run NC programs block-by-block. You can enter the desired movements one at a time on the keyboard (HPU) in the form of individual NC program blocks.</p> <p>The control executes the blocks for channel 1 when you press the “Start” key.</p> <p>Caution</p> <p>The same safety interlocks must be implemented as for fully automatic mode. The same conditions must be met as fully automatic mode.</p> <p>The automatic functions (block execution) are active in “MDI” mode.</p> <p>Select mode: “MDI” control signal (user DB “FMx”, DBX100.1)</p> <p>Mode checkback: “MDI” checkback signal (user DB “FMx”, DBX120.1)</p> <p>Start NC block execution: “Start” control signal (user DB “FMx”, DBX108.1)</p>
Automatic	see Section 9.12
Automatic Single Block	see Section 9.12
TEACH IN Submode of “Automatic” Control and checkback signals	<p>The “TEACH IN” function can be used to create NC programs (main programs and subprograms) for motion sequences or simple workpieces by moving the axes and saving the positions.</p> <p>(see <i>Handheld Programming Unit Operator’s Guide</i> “TEACH IN” section)</p> <p>Select mode: “Automatic” and “TEACH IN” control signal (user DB “FMx”, DBX100.0 and DBX101.0)</p> <p>Mode checkback: “Automatic” and “TEACH IN” checkback signal (user DB “FMx”, DBX120.0 and DBX121.0)</p>

Table 13-29 Operating modes and their properties, continued

Mode	Property
<p>REPOS (repositioning) Submode of "Jog" or "MDI"</p> <p>Control and checkback signals</p>	<p>When a program running in Automatic mode is interrupted (e.g. in the event of a tool breakage), the "Direction plus/minus" keys are used in "Jog" mode to move away from the interruption point (in order to replace the tool). "REPOS" can then be used, again in association with the "Direction plus/minus" keys, to move back to the exact previous position in order to resume the program in "Automatic" mode.</p> <p>(see <i>Handheld Programming Unit Operator's Guide</i> "Repositioning" section)</p> <p>REPOS offsets are matched when the mode is changed to Automatic and Start is activated with program feed and linear interpolation.</p> <p>Select mode: "Jog" and "REPOS" control signal or "MDI" and "REPOS" (user DB "FMx", DBX100.2 and DBX101.1) or (user DB "FMx", DBX100.1 and DBX101.1)</p> <p>Mode checkback: "Jog" and "REPOS" checkback signal or "MDI" and "REPOS" (user DB "FMx", DBX120.2 and DBX121.1) or (user DB "FMx", DBX120.1 and DBX121.1)</p>

13.6 NC programming

Overview

In this section, you can find information about:

- Tool orientation
- Singular positions and their handling
- Handling transformation call
- Tool programming
- Cartesian PTP traversing

13.6.1 Tool orientation

General

Only one degree of freedom exists for orientation in 4-axis kinematics.

For this reason, it is only possible to implement orientation programming using **orientation angle A** (not a machine axis identifier).

This angle corresponds to RPY angle C of the robotic definition. That is a rotation around the Z_{RO} axis, as shown in the diagram below.

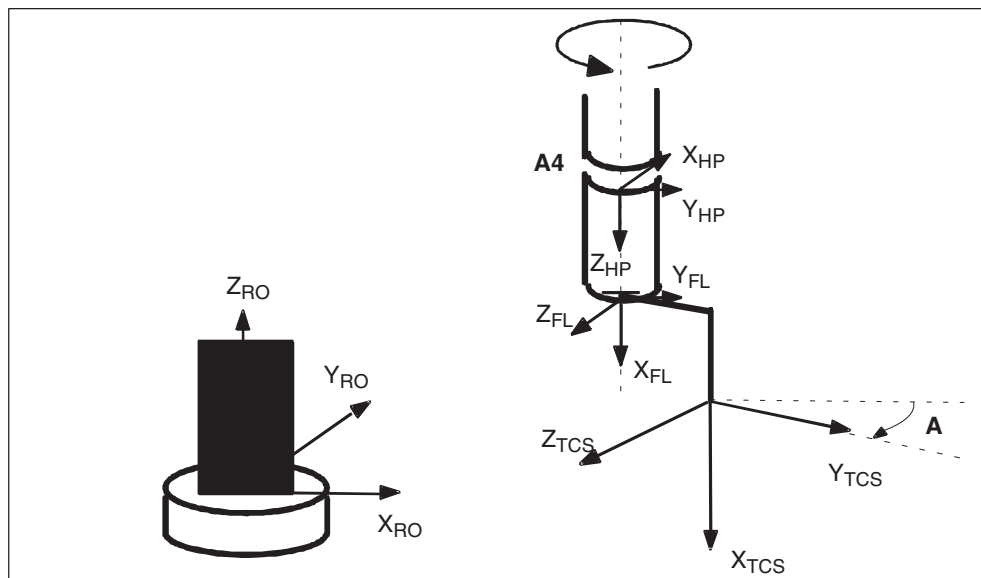


Fig. 13-28 Orientation angle for 4-axis system

Programming

A... ; Orientation angle A in degrees

13.6.2 Singular positions and their handling

General

The calculation of the machine axes at a defined position, i.e. a position with orientation, does not always yield a unique result. Depending on the kinematic response of the machine, positions with an infinite number of solutions can occur. These positions are called “singular”.

Singular positions

A singular position also occurs with articulated arm and SCARA kinematics when the **third axis** is positioned at **0°** or **180°**. These positions are known as **stretch/bend singularity**.

A further singular position also occurs with articulated arm kinematics when the hand point is above the axis of rotation of axis 1. This position is known as **over-head singularity**.

Extreme velocity excesses

If the path runs in the vicinity of a pole (singularity), one or more axes may traverse at extremely high velocity. In this case, error 10910 “Excessive velocity of one axis” is output.

Behavior at pole

The undesired behavior of rapid compensating movements can be improved by reducing the velocity in the vicinity of the pole. It is not usually possible to cross the pole when a transformation is active.

13.6.3 Calling the handling transformation

Programming

TRAORI(1) ; Activate handling transformation

TRAFOOF() ; Deactivate handling transformation
or
TRAFOOF

Notice

When the transformation is activated with TRAORI(1), the “Transformation active” signal (user DB “FMx”, DBX124.6) is set to “1”.

If the machine data are not defined for a transformation group which has been called, the NC program stops and the control outputs error 14100 “Orientation transformation does not exist”.

When the “Handling transformation” is deactivated, an implicit preprocessor stop is executed (see also Section 10.20).

When the transformation is deactivated, the “Transformation active” signal (user DB “FMx”, DBX124.6) is set to “0”.

Illegal programming

When transformation is active, activation of the following functions triggers error 17630 “Referencing not possible for transformed axis” or 17620 “Fixed point approach not possible for transformed axis”:

- G74 Reference point approach
- G75 Fixed point approach

Reset/end of program

The behavior of the control with reference to transformations after power-up, end of program or reset depends on the reset response MD “Active kinematic transformation”.

13.6.4 Tool programming

General

The tool lengths are specified with reference to the flange coordinate system. 3-dimensional tool offsets are possible. Additional restrictions apply to the tool for 4-axis kinematics, depending on the kinematic operation involved.

Automatic tool radius compensation is not possible.

The direction of the tool depends on the initial setting of the machine specified by G codes G17, G18 and G19.

Tool lengths refer to the zero position specified by G17. This should not be changed in the program.

Example

Let us assume a 2-dimensional tool as an example. Type 200 (drill or similar tool) is to be used as the tool identifier. The tool lengths are derived as shown in Figures 13-29. X-TOOL must be entered in the tool parameters as tool length x, Y-TOOL as tool length y.

```

$TC_DP1[1,1] = 200      ; Tool type (200), with tool length compensation only
$TC_DP3[1,1] = 0.0     ; Z length compensation vector
$TC_DP4[1,1] = Y-TOOL ; Y length compensation vector
$TC_DP5[1,1] = X-TOOL ; X length compensation vector
    
```

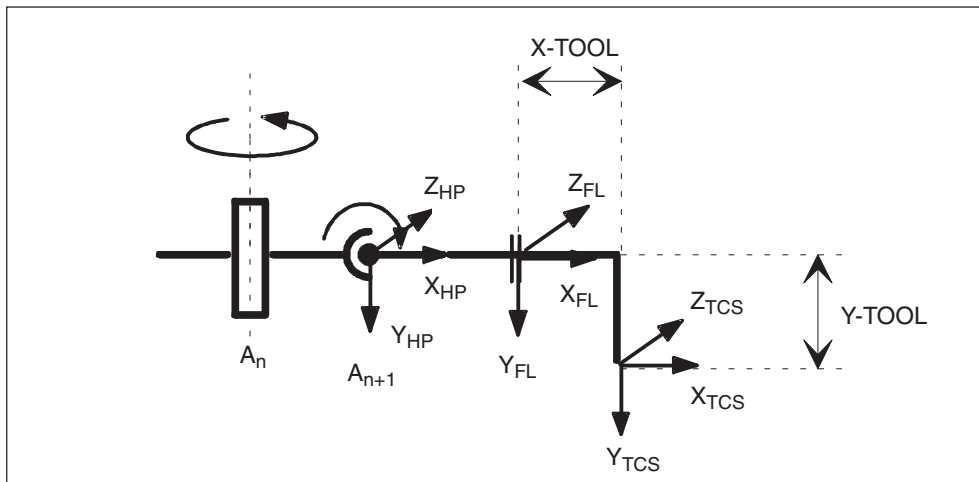


Fig. 13-29 Tool length programming

13.6.5 Cartesian PTP traversing

General

The “Cartesian PTP traversing” function (point-to-point traversing) allows you to program a position in a Cartesian coordinate system (WCS) while implementing the movement of the machine in machine coordinates.

A practical application of this function is traversing through a singularity with transformation TRAORI active. The direct traversing of the machine axes yields an optimized time response compared with Cartesian traversing with a programmed feedrate.

The position and turn information are required in addition to programming a unique point.

PTP traversing is only allowed with G0 and G1.

If an interpolation type other than G0 or G1 is set, and PTP traversing is programmed, error 14144 “PTP traversing not possible” is output.

Programming

PTP	; Point-to-point traversing
CP	; Cartesian path movement
STAT=...	; Position of joints
TU=...	; Turn information

PTP

When PTP is active, only one transformation of the target point is performed, and the machine axes are traversed in synchronism.

The motion with the PTP setting is executed as a synchronized axis movement where the slowest axis involved in the movement determines the velocity.

CP

If CP is active, a Cartesian path movement is performed.

STAT

A machine position is generally not uniquely defined by the position in Cartesian coordinates and the orientation of the tool. Various different joint positions exist, according to the kinematic type used. These different joint positions are transformation-specific.

In order to convert a Cartesian position unambiguously into the axis angle, the **STAT** address is used to identify the position of the joints.

The STAT address contains a binary value (max. 32 bits) with one bit for each of the possible positions. The meaning of the bits is defined as follows for the handling transformation:

- Bit 0: Overhead position
- Bit 1: Position of axis 2 to 3

Programming of the position is only practical for “Cartesian PTP traversing”, because a position change is not usually possible when traversing with transformation active. When traversing with active CP, the position for the destination point is therefore taken from the start point.

TU

Since the transformation core modules only process axis angles in the range between -180 degrees and $+180$ degrees, additional information must be introduced for Cartesian positions, in order to enable unambiguous approach of axis angles greater than ± 180 degrees.

This information is specified in the turn address TU. It represents the sign of the axis angle. By programming the turn information, an axis angle of $|\theta| < 360$ degrees can be approached unambiguously.

The TU variable contains a bit for each axis involved in the transformation. The bit indicates the travel direction.

Bit 0 → Axis 1

Bit 1 → Axis 2, etc.

TU bit =0 : $0^\circ \leq \theta < 360^\circ$

TU bit =1 : $-360^\circ < \theta < 0^\circ$

TU bit = 0 is set for linear axes.

The TURN information (TU) is not modal. If no TU parameter is specified for a position, the position is always approached across the shortest path.

Axes with a traversing range of $> \pm 360^\circ$ are also traversed across the shortest path, because the axis position cannot be determined unambiguously from the TURN information.

PTP/CP switchover in the program

The switchover between the Cartesian method and the machine axis method is set with the language commands **PTP** and **CP**.

The commands are modal and belong to the same G group. The default is CP.

Programming example

Example program for ambiguity:

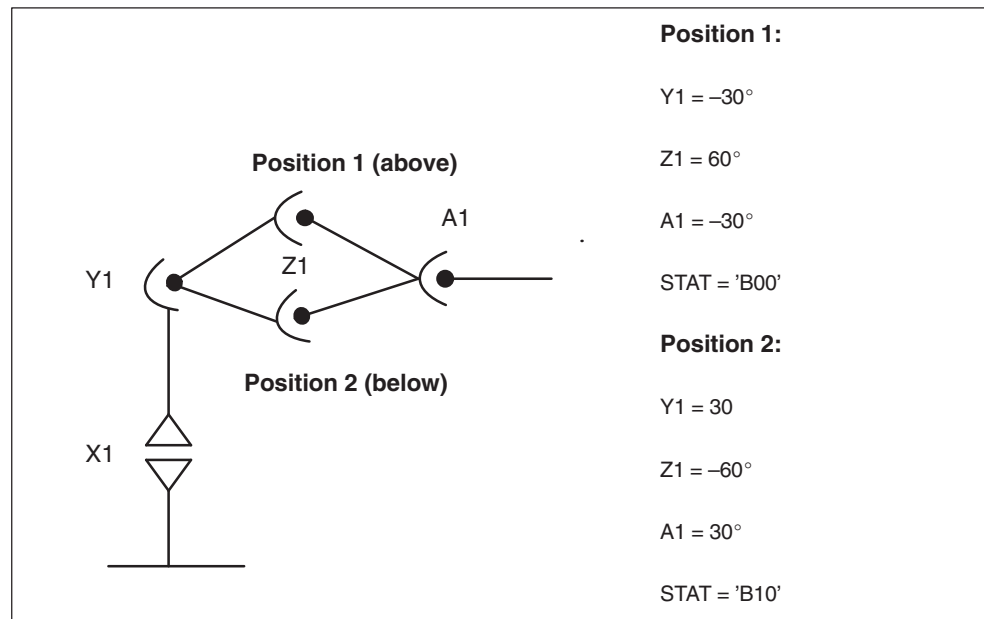


Fig. 13-30 Ambiguity, elbow above or below

Program:

```

N10 G0 X0 Y-30 Z60 A-30 F10000 ; Initial position → elbow above
N20 TRAORI(1) ; Transformation on
N30 X1000 Y0 Z400 A0
N40 X1000 Z500 A0 STAT='B10' TU='B100' PTP ; Reorientation without
; transformation → elbow below
N50 X1200 Z400 CP ; Transformation active again
N60 X1000 Z500 A20
N70 M30
    
```

PTP/CP switchover in Jog mode

The “Activate point-to-point traversing” signal (user DB “FMx”, DBX584.4) provides a simple means of activating or deactivating the transformation in Jog mode. This control signal is only active in Jog mode if a transformation was activated in the program.

When you switch back to Automatic mode, the status which was active before the switchover to Jog is restored.

The active status is reported by the “Point-to-point traversing active” signal (user DB “FMx”, DBX626.6).

Mode change

The “Cartesian PTP traversing” function is only appropriate in Automatic and MDI (MDA) modes. Traversing is performed with the CP setting when the mode is changed to Jog. The mode which was last active is restored when you switch back to Automatic or MDI (MDA).

Submode “REPOS”

The setting for “Cartesian PTP traversing” is not changed during repositioning. If PTP was set during the interruption block, PTP is also used for the repositioning operation.

System variables

The following system variables can be used in association with the “Cartesian PTP traversing” function.

System variables	Significance	Access to NC programs	Access to synchronized actions	Type
\$AC_IWSTAT	Current position information of machine, for description of ambiguities (bit-coded) $0 \leq \$AC_IWSTAT \leq 2^{32}-1$	Read	Read	INTEGER
\$AC_IWTU	Current position of axes Bit n = 0 : Axis n positive Bit n = 1 : Axis n negative	Read	Read	INTEGER

13.6.6 Example program for handling transformation

Simple motion program example:

Parts are fetched from a device and inserted in a machine. When the part has been machined, it should be removed from the machine and stored again. The machine is started via an output signal and uses an input signal to indicate that machining has finished. An input also indicates whether a part to be machined is available in the pickup device.

```

... ; Handling transformation is already active
N10 G1 G90 G53 F20000 ; Linear interpolation with feed F,
; Absolute dimensions, zero offset OFF
N20 $AC_OVR=50 ; Set path feed to 50 %
N30 G0 X... Y... Z... A... ; Rapid traverse (velocity and acceleration
; stored in machine data), wait position –
; taught or programmed
N40 M80 ; Open gripper (let M80 be declared for this function)
N50 WHENEVER $A_IN[1]==0 DO RDISABLE ; Wait for part:
; Read-in disable for next block but one,
; as long as input 1 = 0 (synchronized action)
N60 G4 F0.1 ; Dwell 0.1 s
; only continue if 1 = 1:
N70 X... Y... Z... A... ; Approach part
N80 M81 ; Close gripper (let be defined for this function)
N90 X... Y... Z... A... ; Approach machine
N100 G1 X... Y... Z... A... ; Insert part
N110 M80 ; Open gripper
N120 X... Y... Z... A... ; Move arm out of machine
N130 M90 ; Start machine (M90 – declared for
; setting output signal)
N140 WHENEVER $A_IN[2]==0 DO RDISABLE ; Wait for finished signal:
; Read-in disable for next block but one,
; as long as input 2 = 0 ist (synchronized action)
N150 G4 F0.1 ; Dwell 0.1 s
; only continue if 2 = 1:
N160 X... Y... Z... A... ; Approach finished part
N170 M81 ; Close gripper, pick up part
N180 X... Y... Z... A... ; Move arm out of machine
N190 G0 X... Y... Z... A... ; Approach store with rapid traverse
N200 M80 ; Open gripper, deposit part
N210 M30 ; End of program

```

13.7 Error handling

Table 13-30 "Handling transformation" error list

Error no.	Error message, error analysis and remedy	
10 910	Channel %1 block %2 Excessive velocity of one path axis	
	Cause	%1 = channel number; %2 = block number, label While transformation was selected, an extreme velocity increase occurred in or more axes, e.g. because the path runs in the vicinity of the pole.
	Effect	Warning
	Elimination	Subdivide the NC block into several blocks (e.g. 3), so that the path section with the excessive velocity, and thus the duration, is as short as possible. The other blocks are then traversed with the programmed velocity.
	Acknowl.	Clear error with the CANCEL key.
12 140	Channel %1 block %2 Functionality %3 not contained in this release	
	Cause	%1 = channel number; %2 = block number, label; %3 = Function not relevant
	Effect	NC block preparation stop
	Elimination	Correct the NC program
	Acknowl.	Clear error with "Start".
14 100	Channel %1 block %2 Orientation transformation not available	
	Cause	%1 = channel number; %2 = block number, label Machine data defined incorrectly
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Change the parameters
	Acknowl.	Clear error with "Reset".
14 140	Channel %1 block %2 Programming of position without transformation	
	Cause	%1 = channel number; %2 = block number, label Positional information was programmed for an axis position, but a transformation is not active.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
	Acknowl.	Clear error with "Start".

Table 13-30 "Handling transformation" error list, continued

Error no.	Error message, error analysis and remedy	
14 144	Channel %1 block %2 PTP movement not possible	
	Cause	%1 = channel number; %2 = block number, label The PTP G code is programmed for a movement other than G0 or G1.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
Acknowl.	Clear error with "Start".	
14 146	Channel %1 block %2 CP movement not possible	
	Cause	%1 = channel number; %2 = block number, label The CP G code was programmed for a movement, but a transformation is not active.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
Acknowl.	Clear error with "Start".	
17 620	Channel %1 block %2 Fixpoint cannot be approached for transformed axis %3	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name Fixed point approach (G75) has been programmed for an axis in the block indicated, but the axis is involved in the active transformation. The fixed point is not approached.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Remove the G75 instruction from the NC block or deselect the transformation first with TRAFOOF.
Acknowl.	Clear error with "Reset".	
17 630	Channel %1 block %2 Referencing not possible for transformed axis %3	
	Cause	%1 = channel number; %2 = block number, label; %3 = axis name Reference point approach (G74) has been programmed for an axis in the block indicated, but the axis is involved in the active transformation. The reference point is not approached.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Remove the G74 instruction or the machine axes involved in the transformation from the NC block or deselect the transformation first with TRAFOOF.
Acknowl.	Clear error with "Reset".	

Table 13-30 "Handling transformation" error list, continued

Error no.	Error message, error analysis and remedy	
75 200	Channel %1 Incorrect MD configuration, error in %2	
	Cause	%1 = channel number; %2 = machine data The machine data were configured incorrectly
	Effect	No Ready signal
	Elimination	Check the parameters
Acknowl.	Switch the FM off/on	
75 250	Channel %1 Tool parameter error	
	Cause	%1 = channel number The tool parameters do not match the specifications for handling transformation.
	Effect	NC block preparation stop
	Elimination	Correct the NC program
Acknowl.	Clear error with "Reset".	
75 255	Channel %1 Working area error	
	Cause	%1 = channel number The programmed point is outside the kinematic working area.
	Effect	NC block preparation stop
	Elimination	Correct the NC program
Acknowl.	Clear error with "Reset".	
75 260	Channel %1 block %2 Tool parameter error	
	Cause	%1 = channel number; %2 = block number, label The tool parameters do not match the specifications for handling transformation.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
Acknowl.	Clear error with "Reset".	
75 265	Channel %1 block %2 Working area error	
	Cause	%1 = channel number; %2 = block number, label The programmed point is outside the kinematic working area.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
Acknowl.	Clear error with "Reset".	

Table 13-30 "Handling transformation" error list, continued

Error no.	Error message, error analysis and remedy	
75 270	Channel %1 Tool parameter error	
	Cause	%1 = channel number The tool parameters do not match the specifications for handling transformation.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
	Acknowl.	Clear error with "Reset".
75 275	Channel %1 block %2 Working area error	
	Cause	%1 = channel number; %2 = block number, label The programmed point is outside the kinematic working area.
	Effect	<ul style="list-style-type: none"> • Start disable • NC block preparation stop
	Elimination	Correct the NC program
	Acknowl.	Clear error with "Reset".



Technical Specifications

A

General

This chapter describes the technical data of the FM 357-2 multi-axis module.

- General technical data
- Dimensions and weight
- Load memory
- Encoder inputs
- Drive port
- Digital inputs/outputs

General technical data

The general technical data are as follows:

- Electromagnetic compatibility
- Shipping and storage conditions
- Ambient mechanical and climate conditions
- Data on insulation testing, protection class and degree of protection

This information contains standards and test values incorporated into the S7-300 with which it is also in compliance with, or according to whose criteria the S7-300 was tested.

The general technical specifications are described in the manual *S7-300, M7-300 Programmable Controller, Module Data*.

UL/CSA approvals

The following approvals have been granted for the FM 357-2:

UL Recognition Mark
Underwriters Laboratories (UL) in compliance with
UL Standard 508, File E 164110

CSA Certification Mark
Canadian Standard Association (CSA) in compliance with
Standard C 22.2 No. 142

FM approval

The following FM approval has been granted for the FM 357-2:
 FM Approval according to Factory Mutual Approval Standard Class Number 3611,
 Class I, Division 2, Group A, B, C, D.



Warning

Potential for personal injury and property damage.

In areas where there is a risk of explosion, personal injury and property damage may occur if you disconnect plugs during operations.

In areas where there is a risk of explosion, always cut off power to the S7-300 before disconnecting plugs.



Warning

WARNING - DO NOT DISCONNECT WHILE CIRCUIT IS ALIVE
 UNLESS LOCATION IS KNOWN TO BE NON-HAZARDOUS

CE marking

Our products meet the general and safety-related requirements of the following EC guidelines and conform to the uniform standards (EN) for programmable controllers published in the official gazettes of the European Union:

89/336/EEC "Electromagnetic Compatibility" (EMC guideline)

The EC declaration of conformity is contained in this manual.

EMC Directive

SIMATIC products are designed for industrial applications.

Application	Requirement concerning	
	Noise emission	Noise immunity
Industry	EN 50081-2 : 1993	EN 61000-6-2 : 1999

Declaration of conformity

The current Declaration of Conformity is on the Internet at

<http://support.automation.siemens.com/WW/view/en/15257461>

Please follow installation guidelines

SIMATIC products will fulfill the relevant requirements if they are installed and operated in accordance with the installation guidelines specified in the product manuals.

Connection data

Supply voltage	20.4 to 28.8 V
Power consumption from 24 V	1 A
Power loss	15 W
Startup current	2.6 A
Power consumption from 5 V backplane bus	100 mA
Encoder power supply 5 V max. output current	1.35 A
Encoder power supply 24 V max. output current	1.0 A

Dimensions and weight

Dimensions W × H × D (mm)	200 × 125 × 118
Weight (g)	approx. 1 200

User data memory

Non-volatile RAM, approx. 750 KB (for exact information see parameterizing tool via menu **Target system > Properties > Index card “Memory utilization”**)

System cycles

Position control cycle: 3 ms; interpolation cycle (IPO cycle): 9 ms

Drive port

Analog drive

Setpoint signal	
Rated voltage range	-10.5...10.5 V
Output current	-3...3 mA
Servo enable relay contact	
Switching voltage	Max. 50 V
Switching current	max. 1 A
Switching capacity	max. 30 VA
Cable length	35 m

Stepper drive

5 V output signals in conformity with RS 422 standard		
Differential output voltage	V_{OD}	min. 2 V ($R_L = 100 \text{ } \Omega$)
Output voltage "1"	V_{OH}	3.7 V ($I_O = -20 \text{ mA}$) 4.5 V ($I_O = -100 \text{ mA}$)
Output voltage "0"	V_{OL}	max. 1 V ($I_O = 20 \text{ mA}$)
Load resistance	R_L	min. 55 Ω
Output current	I_O	max. $\pm 60 \text{ mA}$
Pulse frequency	f_P	max. 750 kHz
Cable length		max. 50 m in hybrid configurations with analog axes 35 m with asymmetrical transfer 10 m

Encoder inputs

Position detection	<ul style="list-style-type: none"> • Incremental • Absolute (SSI)
Signal voltages	Inputs: 5 V, RS 422-compliant
Encoder supply voltage	<ul style="list-style-type: none"> • 5 V/300 mA • 24 V/300 mA
Input frequency and line length for incremental encoder	<ul style="list-style-type: none"> • max. 1 MHz with 10 m conductor length shielded • max. 500 kHz with 35 m conductor length shielded
Data transmission rates and line length for absolute encoder (SSI)	<ul style="list-style-type: none"> • max. 1.5 Mbit/s with 10 m conductor length shielded • max. 187.5 kbit/s with 250 m conductor length shielded
Cable length with incremental encoder <ul style="list-style-type: none"> • 5 V encoder supply • 24 V encoder supply 	<ul style="list-style-type: none"> • max. 25 m at max. 300 mA (tolerance 4.75...5.25 V) • max. 35 m at max. 210 mA (tolerance 4.75...5.25 V) • max. 100 m at max. 300 mA (tolerance 20.4...28.8 V) • max. 300 m at max. 300 mA (tolerance 11...30 V)
Cable length with absolute encoder (SSI)	See data transfer rate

Digital inputs

Number of inputs	18
Supply voltage	24 V DC (permissible range: 20.4...28.8 V)
Electrical isolation	Yes
Input voltage	<ul style="list-style-type: none"> • 0 signal: -3 to 5 V • 1 signal: 11 to 30 V
Input current	<ul style="list-style-type: none"> • 0 signal: ≤ 2 mA • 1 signal: 6 to 30 mA
Input delay (I0 to I11)	<ul style="list-style-type: none"> • 0 → 1-Signal: type 15 ms • 1 → 0-Signal: type 150 ms
Connecting a 2-conductor sensor	Possible

Digital outputs

Number of outputs	8
Supply voltage	24 V DC (permissible range: 20.4...28.8 V)
Electrical isolation	Yes
Output voltage	1 signal: $V_L^{1)} - 3...V_L^{1)}$ V
Max. output current <ul style="list-style-type: none"> • at ambient temperature (40°C) <ul style="list-style-type: none"> – rated value – permitted range – lamp load • at ambient temperature (55°C) <ul style="list-style-type: none"> – rated value – permitted range – lamp load 	1 signal: 0,6 A 0.5 A 5 mA...0.6 A (via supply voltage) max. 5 W 0.25 A 5 mA...0.3 A (via supply voltage) max. 2.5 W
Max. leakage current	0 signal: 2 mA
Output delay (Q0...Q7)	<ul style="list-style-type: none"> • 0 → 1-signal: type 500 ms • 1 → 0-signal: type 500 ms

1) VL – Supply voltage of outputs



B

List of Abbreviations

“A/ASBL” mode	“Automatic/Automatic Single Block” mode
ASCII	American Standard Code for Information Interchange
ASUB	Asynchronous Subroutine
AT	Advanced Technology
AW-DB	User Data Block
BCD	Binary Coded Decimals
BD	Basic Data
BR	Binary Result
COM	Communication Module
CPU	Central Processing Unit of the SIMATIC S7
CRC	Cutter Radius Compensation
CTS	Clear To Send (for serial interfaces)
DAC	Digital–Analog Converter
DB	Data Block
DBB	Data Block Byte
DBD	Data Block Double Word
DBW	Data Block Word
DBX	Data Block Bit
DCE	Data Communications Equipment
DP	Distributed Peripherals (I/Os)
DPR	Dual-Port RAM
DRAM	Dynamic memory (volatile)
DRF	Differential Resolver Function
DRV	Driver Module
DRY	Dry Run
DSC	Dynamic Servo Control
DSG	Dimension System Grid
DSR	Data Send Ready (on serial interfaces)
DW	Data Word
DTE	Data Terminal Equipment
EG	Electronic gear
EMC	ElectroMagnetic Compatibility

EN	Enable (input parameter in ladder diagram)
ENO	Enable Output (output parameter in ladder diagram)
EPROM	Erasable Programmable Read-Only Memory with permanently stored program
ESD	Components endangered by ElectroStatic Discharge
EXE	External pulse shaper electronics
FB	Function Block
FC	Function Call, function block in CPU
FDD	Feed Drive
FEPROM	Flash EPROM: Read/write memory
FIFO	First in First Out: Memory that operates without addresses where the data are always read out in the same order in which they were stored.
FIPO	Fine Interpolator
FM	Function Module
FST	Feed Stop
GEO	Geometry
GND	Signal ground (reference point)
HEX	Abbreviation for hexadecimal number
HMI	Unit for operation and monitoring of a process
I	Input parameter
I/O	In/out parameter (activation parameter)
I/RF	Infeed/Regenerative Feedback module
IM	Interface Module (SIMATIC S7)
INC	Increment
INI	I nitializing data
INTM	Internal Multiplication
IPO cycle	Interpolation cycle
“ITR” mode	“Incremental Travel Relative” mode
“J” mode	“Jog” mode
K bus	Communication bus
LAD	Ladder Diagram (type of representation in STEP 7)
LED	Light Emitting Diode
MCS	Machine Coordinate System
MDI	Manual Data Input
“MDI” mode	“ <u>M</u> anual <u>D</u> ata <u>I</u> nput” mode
MLFB	Machine-readable product designation (order no.)
MPI	Multi Point Interface

MS	Mains Supply
MSD	Main Spindle Drive
NC	Numerical Control
NCK	Numerical Control Kernel
O	Output parameter
OB	Organization Block of CPU
OMP	Operating Mode Parameter
OP	Operator Panel
PCMCIA	Personal Computer Memory Card International Association
PELV	Protective Extra Low Voltage
PG	Programming Device
PLC	Programmable Logic Controller
PRS	Position Reached, Stop
PS	Power Supply (SIMATIC S7)
PWM	Pulse Width Modulation
RAM	Random Access Memory in which data can be read and written
“REF” mode	“Reference Point Approach” mode
REPOS	Repositioning (submode of Jog or MDI)
RFG	Controller enable
ROV	Rapid Override
RPA	R Parameter Active
RPS	Reference Point Switch (cam)
S7-300	Programmable logic controller in medium performance range
SDB	System Data Block
SFC	System Function Call, system services (integrated functions)
SKP	Skip block
SM	Signal Module (SIMATIC S7, e.g. input/output module)
SPF	Sub Program File
SRAM	Static memory (non-volatile)
SSI	Synchronous Serial Interface
SSL	System Status List
STEP 7	Programmer software for SIMATIC S7
STL	Statement List (type of representation in STEP 7)
TC	Tool Compensation
TEA	Testing Data Active: Refers to machine data
TF	Technology Function

TO	Tool Offset
TOA	Tool Offset Active
TRC	Tool Radius Compensation
UDT	User-defined Data Type
UE	Unregulated supply
UP	User Program
VDI	Virtual Device Interface
VGA	Video Graphics Array
WCS	Workpiece Coordinate System
ZOA	Zero Offset Active



Index

Zahlen

- 3-axis kinematics, 13-48
- 4-axis kinematics, 13-56

A

- Absolute dimensioning G90, 10-14
- Absolute dimensioning, rotary axes, 10-15
- Absolute encoder (SSI), 4-23
- Absolute encoders (SSI), 9-22, 9-65
- Absolute encoders SSI), Parameters, 9-22
- Acceleration, 5-29, 9-39, 9-41, 9-42, 9-43
 - Brisk acceleration, 9-41
 - Knee-shaped acceleration, 9-43
 - Soft acceleration, 9-42
- Acceleration pattern, 5-29, 9-41
- Acceleration response, 10-70
- Action list, 12-92
- Activate program test, 6-80
- Activate skip block, 6-80, 10-7
- Activating machine data (NEWCONF), 10-153
- Actual velocity monitoring, 9-49
- Alignment, 9-56
 - Parameters, 9-66
 - With absolute encoders, 9-65
- Analog drives, Signals, 4-14
- Analog inputs, on the local P bus, 9-82
- Analog outputs, on the local P bus, 9-82
- Application area, 1-2
- Application example, 11-1
- Application examples, 6-100
- Arithmetic parameters
 - Comparison operations, 10-104
 - Operators/arithmetic functions, 10-104
- Articulated joint definition, 13-37
- Asynchronous subprogram (ASUB), 9-107
- Asynchronous subprogram (ASUP), 6-58
- Asynchronous subroutine (ASUB), 10-149
- Automatic, 9-103
- Automatic single-block, 9-104
- Auxiliary functions, 6-57, 9-73
- Axes, 10-12
- Axis configuration, 9-6
- Axis control from the CPU, 6-49
- Axis errors, 12-47

- Axis module (fixed SDB), 9-9
- Axis motion, 10-31
 - Circular interpolation, 10-44
 - Linear interpolation with feed, 10-39
 - Linear interpolation with rapid traverse, 10-37
 - Positioning movements, 10-40
 - Programmable block change for positioning axes (FINEA, COARSEA, IPOENDA, IPOBRAKE), 10-41
 - Programming feeds, 10-32
- Axis motions, Positioning motion at rapid traverse, 10-38
- Axis name, 9-6
 - Channel axis, 9-6
 - Geometry axis, 9-6
 - Machine axis, 9-6
 - Special axis, 9-6
- Axis number, 9-6
- Axis replacement, 9-163, 10-192
- Axis type, 9-7
- Axis types, 10-11
 - Machine axes, 10-12
- Axis variable, 10-131
- Axis velocity, 5-29, 9-40

B

- Backlash compensation, 9-29
 - Parameters, 5-28, 9-30
- Backup battery, 4-37
 - Disposal, 4-39
 - Inserting, 4-37
 - Replacing, 4-38
 - Rules for handling, 4-39
 - Storage, 4-38
 - Type, 4-38
- Basic axis configuration, 13-40
- Battery compartment, 4-37
- Battery type, 4-38

C

- Calling the handling transformation, 13-71
- Cartesian PTP traversing, 13-73

- CE marking, ii, A-2
- Chamfer and rounding (CHF, CHR, RND, RNDM), 10-62
- Channel assignment, 9-14
- Channel axis, 9-6
- Channel configuration, 9-14
- Channel error, 12-13
- Checkback signals, 6-75
- Circular interpolation, 10-44
- Clamping monitor, 9-48
- Configuration, 9-3
 - Axis configuration, 9-6
 - Channel configuration, 9-14
 - Parameters, 9-15
- Configuring, 5-5
 - sample, 5-6
- configuring, sample, 5-12, 5-15
- Configuring the kinematic transformation, 13-38
- Connecting cable, 4-6
 - PROFIBUS DP cable, 4-6
 - Setpoint cable, 4-6
- Connecting cables
 - Measuring system cable, 4-6
 - Measuring system cables, 4-28
 - MPI cable, 4-6
- Connecting the drive units, 4-18
- Connection data, A-3
- Continuous-path mode, 10-67
- Control signals, 6-75
- Control structures, 10-126, 10-129
- Controlling, 9-13, 9-157
 - Activation, 9-157
 - Example, 9-158
 - Motion profile, 9-158
- Coordinate systems, 10-10
- Coupled motion, 9-114, 10-74, 10-76
- CPU/FM 357-2 communication, 6-7
- Creation of the user data blocks
 - by FC 1, 6-4
 - offline, 6-5
- Creep acceleration, 5-29, 9-43
- Creep velocity, 5-29, 9-43
- CSA approval, A-1
- Curve table, 9-126, 10-182
 - Non periodic, 10-183
 - Parameters, 5-35, 9-126, 9-127
 - Periodic, 10-182
- Cycle time, 9-4

D

- DB 121: VAR_ADDR – Extract from FM variable list, 6-45
- Description of signals, 6-75
- Description of the standard function blocks, 6-16
- Diagnostic errors, 12-9
- Digital inputs, On-board inputs, 4-31, A-5
- Digital outputs, On-board outputs, 4-32, A-5
- Dimensions of the FM 357-2, A-3
- Direction reversal actual value, 9-29
- Distributed installation, 1-5
- Distributed installations, 6-47
- Distribution of equipment, 1-4
- Drift compensation, 9-34
- Drift limit value, 9-34
- Drive, 9-8
- Drive interface, pin assignment, 4-13
- DSC (Dynamic Servo Control), 9-13
- Dwell, 10-73
- Dynamic Servo Control (DSC), 9-37

E

- Electronic gear, 9-131, 10-187
- EMC guidelines, 4-2
- EMERGENCY OFF concept, 4-1
- EMERGENCY STOP, 6-76, 6-77, 9-155
 - Parameters, 5-29, 9-155
 - Sequence, 9-155
- Encoder
 - Parameters, 5-27, 9-19
 - Selection, 9-18
- Encoder alignment, 9-67
- Encoder inputs, A-4
- Encoder power supply, 4-25
- Encoders, 4-22, 9-18
 - Absolute encoder, 4-23
 - Absolute encoders, 9-22
 - Connecting the encoders, 4-27
 - Incremental encoders, 4-23, 9-20
- Error list
 - Axis errors, 12-47, 12-64
 - Channel error, 12-13
 - Diagnostic errors, 12-9
- Error lists, 12-8, 12-66
- Error messages, LED indicators, 12-2
- Error messages and their effect, 12-6

Event-controlled program calls, 9-68
 External master value, 9-13
 Extreme velocity , 13-70

F

FB 2: FMGET – Read FM variable, 6-24
 FB 3: FMPUT – Write FM variable, 6-32
 FB 4: FMPI – General services, 6-38
 FC 1: RUN_UP – Startup / initialization, 6-17
 FC 22: BFCT – Basic functions and operating modes, 6-22
 FC 5: BF_DIAG – Diagnostic alarm and FM restart, 6-19
 Feed interpolation, 10-33
 Field of application, A-2
 FIFO- variable (\$AC_FIFO1[...] to \$AC_FIFO10[...]), 10-113
 Firmware, 3-4
 Firmware update, 3-4
 FM approval, A-2
 FM READY output, 4-33
 FM STEPDRIVE, connection, 4-19
 Following error monitoring, 9-47
 Frames, 13-35
 Front connector, 4-6
 Front panel elements, 1-10
 LED indicators, 1-10
 Functional errors, 12-64
 Fundamentals of variable NC programming, 10-97
 Data types, 10-98
 Operations/arithmetic functions, 10-98
 Priorities of operators and execution, 10-101
 type conversion, 10-100
 Variable types, 10-97

G

Gantry, 9-117
 Commissioning, 9-123
 Initial start-up, 9-122
 Interface signals, 9-118
 Parameters, 5-35, 9-117
 Geometry axis, 9-6, 10-12

H

H functions, 6-57, 10-90
 Output, 6-57, 9-74

Handheld programming unit (HPU), 13-2
 Handling Transformation, 13-1
 Handling transformation
 3-axis kinematics, 13-48
 4-axis kinematics, 13-56
 Articulated joint definition, 13-37
 Basic axes, 13-39
 Calling the handling transformation, 13-71
 Cartesian PTP traversing, 13-73
 Configuration, 13-38
 Configuring alarm and message texts, 13-29
 Configuring alarm and message texts (HT 6), 13-31
 Connection frames, 13-41
 Diagnostic buffer of the CPU, 13-33
 Error handling, 13-78
 FC 20: AL_MSG – user alarms and user messages, 13-24
 Frames, 13-35
 Functions, 13-35
 Further configuration data, 13-44
 Hand axis, 13-41
 Information, 13-4
 Kinematic descriptions, 13-48
 Machine data list, 13-5
 Message signals in DB 20, 13-25
 NC programming, 13-69
 Parameterization, 13-5
 Programming example, 13-77
 Programming the standard function blocks, 13-10
 Rotation, 13-36
 Singular positions and their handling, 13-70
 Terms, 13-35
 Tool orientation, 13-69
 Tool programming, 13-72
 Translation, 13-36
 User alarms and user messages, 13-22
 Hot-spot measurement, 9-89, 9-98

I

Incremental dimensioning G91, 10-14
 Incremental encoders, 4-23, 9-20, 9-58
 Parameters, 9-21
 Incremental relative, 9-102
 Independent CPU axes, 9-13
 Inputs/outputs
 on local P bus, 9-79
 On-board I/Os, 9-76
 Installing the FM 357-2, 3-3

Instructions (complete list), 10-205

Interface signals

Axis signals, 6-72, 6-90

FM signals, 6-61, 6-75

Interfaces, 1-10, 4-11, 4-12, 4-22, 4-29

Connecting the power supply, 4-7

Drive interface, 1-10

Drive interface for analog and stepper drives, 4-12

I/O interface, 1-10, 4-29

Measuring system interface, 1-10, 4-22

Memory module interface, 1-10

Power supply port, 1-10

PROFIBUS DP drive interface, 4-11

PROFIBUS DP interface, 1-10

SIMATIC bus connector interface, 1-10

Interpolation cycle (IPO cycle), 9-4, A-3

Involute geometry, 10-60

Involute interpolation (INVCW, INVCCW), 10-58

J

Jerk, 5-29, 9-42

Jerk filter, 9-28

Parameters, 5-28

JOG, 13-67

Jog , 9-102

K

Kinematic descriptions, 13-48

Kinematic sequence, 13-38

Kv factor, 9-31

L

Limit signals (software cams), Parameters, 5-33

Limit switching signals (software cams)

Linked output, 9-95

Output, 9-90

Parameters, 9-85

Separated output, 9-92

Linear axes, 9-7

Linear interpolation with feed, 10-39

Linear interpolation with rapid traverse, 10-37

List-based parameterization, 5-36

Local P bus, 1-5, 9-79, 9-82

Location of ports, 1-9

Logic address (external SDB), 9-10

LUD-Viewer, 7-12

M

M functions, 6-57, 10-88

Output, 6-57, 9-73

Machine axes, 10-12

Machine axis, 9-6

Machine data (parameters), 5-23

Online project, 5-23

Value ranges, 5-25

Master axis, 9-124

Master value coupling, 9-124, 10-181

Curve table, 9-126, 10-182

Master axis, 9-124

Parameters, 9-124, 9-125, 9-130, 9-135

Slave axis, 9-124

System variable, 10-190

System variables, 10-186

Master value link, Parameters, 5-34

Maximum velocity, 9-39

MDA, 13-67

MDI, 9-103

Measurement, 9-142

Measurement (programming), 10-78

Axial (MEAS, MEAW), 10-80

Block-specific (MEAS, MEAW), 10-78

Measurement specification, 10-20

Measuring, Global measuring, 9-144

Message frame type (external SDB), 9-11

Module power supply, 4-9

Modulo rotary axes, 9-7

Monitoring, 9-45

Actual velocity, 9-49

Encoder, 9-51

Hardware limit switch, 9-54

Monitoring timeout, 9-46

Software limit switch, 9-54

Speed setpoint, 9-48

Target range coarse, 9-46

Target range fine, 9-46

Monitoring functions

Clamping operation, 9-48

Following error, 9-47

Motion control, 2-1

Motion synchronous actions

Principle, 10-174

Structure, 10-154

MPI connection, 1-14

Multi-tier configuration, 1-4

N

NC program execution, 9-105
 NC programming, 10-1
 Block structure, 10-6
 Program structure, 10-3
 Special characters, 10-9
 Statements, 10-4
 NC programs, 5-38
 NC-VAR Selector, 6-29, 6-36
 Number of electronic gears, Parameters, 9-131

O

Offset compensation, 9-34
 OP 17 menu tree, 8-4
 Operating modes, 9-102
 Operator control and monitoring, 8-1, 8-3
 Optimization, 7-7
 Oscillation, 10-176
 Overlaid motion in synchronized actions, 9-140
 Parameters, 5-34, 9-141
 Override, 6-81
 Override-coding, 9-3

P

Parameter data, 5-21
 Offline editing, 5-22
 Online editing, 5-21
 Parameterization, 5-1
 List-based parameterization, 5-36
 Machine data (parameters), 5-23
 NC programs, 5-38
 Parameterization wizard, 5-24
 R parameters, 5-37
 Tool offset values, 5-38
 Zero offset, 5-37
 Parameterization wizard, 5-24
 Path acceleration, 5-29, 9-44
 Path action, Exact stop, 10-65
 Path axes, 10-12
 Path group, 10-36
 Path jerk, 5-29, 9-44
 Path override, 6-81
 Path response, 9-44, 10-64
 Acceleration response, 10-70
 Continuous-path mode, 10-67
 Target range, 10-65
 Path switching signals (software cams), 9-85
 Hot-spot measurement, 9-98
 Timer-controlled, 9-87
 Path-time cam, 9-89

Performanceanzeige, 7-11
 Plane selection, 10-21
 Polar coordinates, 10-17
 Polynomial interpolation (POLY), 10-54
 Position control, 9-27
 Position control cycle, 9-4
 Position control gain, 9-31
 Position loop gain, Parameters, 9-31
 Positioning motion at rapid traverse, 10-38
 Positioning velocity, 5-29, 9-39
 Power supply, 4-7
 Powering up the FM 357-2, 3-4, 3-7, 7-3
 Product versions, iii
 PROFIBUS DP interface, pin assignment, 4-11
 PROFIBUS drive, 9-8
 Program coordination, 10-139
 Program jumps, 10-124
 Programmable dynamic limitation, 10-72
 Programming
 NC programs, 10-1
 User program, 6-1
 Programming feeds, 10-32
 Programming the FB/FCs, Fundamentals, 6-3
 Programming the standard function blocks
 CPU / FM 357-2 communication, 6-7
 Interface, user data blocks, 6-4
 Standard function blocks, overview, 6-5
 Structure of a user program, 6-9
 Symbolic programming, 6-8
 Testing the user program, 6-11
 Writing a user program, 6-14
 Protection zones, 9-109, 10-93
 Number, 9-110

R

R parameters, 5-37
 R parameters (arithmetic parameters), 10-103
 Rapid traverse override, 5-29, 9-40
 Reading, writing and deleting a file, 10-134
 Reference point approach, 9-103
 Reference point switch, 9-58
 Referencing, 9-56
 Incremental encoders, 9-58
 Parameters, 5-30, 9-60
 Stepper motor without encoder, 9-64
 With reference point switch, 9-58
 Without reference point switch, 9-59
 Removing and replacing the FM, 3-6
 Replacing the battery, 4-38
 REPOS, 13-68
 Rotary axes, 9-7
 Rotation monitoring, 9-53

S

- Safety rules, 4-1
 - EMERGENCY STOP devices, 4-1
- Sample project, 5-6, 5-12, 5-15
- Scaling system, 9-3
- Servicing data, 7-8
- Servo cycle, A-3
- Servo drive (analog), 9-8
- Servo-cycle, 9-4
- Set actual value, 10-30
- Set break point, 6-11
- Signal circuit of the stepper motor interface, 4-17
- SIMATIC Manager, 5-5
- SIMODRIVE 611-A connection, 4-18
- SIMODRIVE 611-U connection, 4-20
- Simulated master value, 10-186
- Single-tier configuration, 1-4
- Singular positions, 13-70
- Singular positions and their handling, 13-70
- Slave axis, 9-124
- Software cams, 9-85
 - Parameters, 5-33, 9-85
- Special axis, 9-6, 10-12
- Speed control loop, 9-35
- Speed feedforward control, 5-32, 9-35
- Speed feedforward control (FFWON, FFWOF), 10-195
- Speed setpoint monitoring system, 9-48
- Spline, 10-47
- Standard function blocks, 6-5
 - Application examples, 6-100
 - DB 121: VAR_ADDR – Extract from FM variable list, 6-45
 - FB 2: FMGET – Read FM variable, 6-24
 - FB 3: FMPUT – Write FM variable, 6-32
 - FB 4 – General services, 6-38
 - FC 1: RUN_UP – Startup / initialization, 6-17
 - FC 22: BFCT – Basic functions and operating modes, 6-22
 - FC 5: BF_DIAG – Diagnostic alarm and FM restart, 6-19
- Start-up, 7-7
- Start-up switch, 7-3
- Startup, with the parameterization tool, 6-15
- Startup switch, 1-9
- Statements, 10-4
 - Overview, 10-196
- Stepper drive, 9-8
 - Signals, 4-15

- Stepper motor, 9-8, 9-26
 - Parameters, 5-28, 9-26
 - With/without encoder, 9-8
- Stop preprocessor, 10-86
- String operations, 10-132
- Structure of a user program, 6-9
 - Error evaluation (information), 6-12
 - Signal processing, 6-10
- Subroutine technique, 10-143
- Symbolic programming, 6-8
- Synchronized actions, 10-154
 - Operators, 10-167
 - System variables, 10-168
- Synchronized axes, 10-12
- System cycles, A-3
- System overview, 1-6
 - Components, 1-6
 - Data handling, 1-8
- System rates, 9-4
- System variables, 10-105

T

- T function, 6-57, 10-91
 - Output, 6-57, 9-74
- Tangential control, 9-137
 - Angular position of the slave axis, 9-138
 - Axis motion, 9-137
 - Axis types, 9-137
 - Behavior at contour corners, 9-139
 - Behavior in the case of direction reversal, 9-140
 - Behavior in the modes, 9-139
 - Coordinate system, 9-138
 - Programming, 9-138
- Target range coarse, 9-46, 10-65
- Target range fine, 9-46, 10-65
- TEACH IN, 13-67
- Technical data, runtime specifications, 6-110
- Testing, 7-7
- Testing the axis , 7-19
- Time constant , 9-35
- Timer-controlled path switching signals, 9-87
- Timing diagrams, communication, 6-99
- Tool offset values, 5-38, 10-91
- Tool orientation, 13-69
- Tool programming, 13-72
- Trace, 7-12, 7-13
- Travel direction reversal, 9-32
 - Parameters, 9-32

- Travel to fixed stop, 9-146, 10-84
 - Clamping torque, 10-85
 - Clock pulse diagrams, 9-152
 - Monitoring window, 10-85
 - Parameters, 5-35, 9-147
- Traversing response, Positioning axes, 10-69
- Troubleshooting , 7-8

- U**
- UL approval, A-1
- Unabhängige CPU-Achse, 6-55
- User data block (user DB 21), 13-13, 13-17
- User data blocks
 - AXy, 6-72
 - FMx, 6-61
 - HT 6, 13-19
 - PHG, 13-17
- User data blocks of the interface to FM 357-2,
 - Description of signals, 6-75
- User data blocks of the interface to the FM 357-2, 6-59
- User handling, function sequences, 6-48
- User notes, 6-48
- User tips
 - Auxiliary functions, 6-57
 - Axis control from the CPU, 6-49
 - Start ASUP program, 6-58
- User variable
 - Application example GUD 106 variable, 10-123
 - application example of GUD variable, 10-123
 - Application example of LUD variable, 10-122
 - Definition, 10-119
 - Range of validity, 10-118

- User variables, 10-118
 - Example for definition, 10-121
 - Number and memory allocation, 10-118
 - Programming, 10-120

- V**
- VDI output, 9-13
- Velocities, 9-39
- Velocity assignment, 9-32, 9-33
 - PROFIBUS DP drive, 9-33
 - Servo drive, 9-32
 - Parameters, 9-32
 - Stepper motor, 9-33
 - Parameters, 9-33

- W**
- Weight, A-3
- Weighting factor, 9-35
- Wiring, 4-1
- Wiring an FM 357-2, 4-3
- Wiring up the front connector, 4-33
- Working area limitations, 10-86
- Writing the user program , 6-14

- Z**
- Zero offset, 5-37
- Zero offsets, 10-22
 - Programmable, 10-25
 - Settable, 10-22

