

SIEMENS

SIMATIC

FM 357 Multi-Axis Module for Servo and Stepper Drives

Manual

04.98 Edition

**This Manual is supplied together with the Configuring Package,
Order No.: 6ES7 357-4AH02-7BG0.**

Siemens AG
Automation Group
Automation Systems Division
for Machine Tools, Robots
and Special-Purpose Machines
P. O. Box 3180, D-91050 Erlangen

Siemens quality for software and training
to DIN ISO 9001, Reg. No. 2160-01
This edition was printed on paper bleached using an
environmentally friendly chlorine-free method.

© Siemens AG 1997-98 All Rights Reserved
Subject to changes without prior notice



Progress
in Automation.
Siemens

Siemens Aktiengesellschaft

Order No.: GWE-570093101798
Printed in the Federal Republic of Germany

04.98 SIMATIC S7-300, FM 357 Multi-Axis Module for Servo and Stepper Drives

SIEMENS

SIMATIC

FM 357 Multi-Axis Module for Servo and Stepper Drives

Manual

Preface, Contents

User Information

Product Overview

Fundamental Principles of
Motion Control

Installation and Removal of the
FM 357

Wiring the FM 357

Parameterization of the FM 357

Programming the FM 357

Starting Up the FM 357

Human-Machine Interface

Reference Information

Description of Functions

NC Programming

Troubleshooting

Appendices

Technical Specifications

EC Declaration of Conformity

List of Abbreviations

List of Indices

1

2

3

4

5

6

7

8

9

10

11

A

B

C

Safety Information

This Manual contains information which you should carefully observe to ensure your own personal safety and the prevention of damage to the system. This information is highlighted by a warning triangle and presented in one of the following ways depending on the degree of risk involved:



Danger

indicates that death, severe personal injury or substantial property damage **will** result if proper precautions are not taken.



Warning

indicates that death, severe personal injury or substantial property damage **can** result if proper precautions are not taken.



Caution

indicates that minor personal injury or property damage can result if proper precautions are not taken.

Note

contains important information about the product, its operation or a part of the document to which special attention is drawn.

Qualified person

The unit may only be started up and operated by a **qualified person or persons**. Qualified persons as referred to in the safety guidelines in this document are those who are authorized to start up, earth and label units, systems and circuits in accordance with relevant safety standards.

Proper use

Please note the following:



Warning

The unit may be used only for the applications described in the catalog or the technical description, and only in combination with the equipment, components and devices of other manufacturers as far as this is recommended or permitted by Siemens.

It is assumed that this product be transported, stored and installed as intended and maintained and operated with care to ensure that the product functions correctly and safely.

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.

Other names in this publication might be trademarks whose use by a third party for his own purposes may violate the rights of the registered holder.

Copyright Siemens AG 1997-98 All rights reserved

The reproduction, transmission or use of this document or its contents is not permitted without express written authority. Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility model, are reserved.

Siemens Aktiengesellschaft
Automation Group
Industrial Automation Division
P.O. Box 4848, D-90327 Nuremberg

Exclusion of liability

We have checked that the contents of this publication agree with the hardware and software described herein. Nonetheless, differences might exist and therefore we cannot guarantee that they are completely identical. The information given in this publication is reviewed at regular intervals and any corrections that might be necessary are made in the subsequent printings. Suggestions for improvement are welcome at all times.

© Siemens AG 1997-98
Subject to technical changes without notice.

Preface

Purpose of this documentation

This Manual contains all information about the FM 357 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 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.
- 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."
- Programming of technology functions (Chapter 6)
Describes how technology functions can be programmed with STEP 7.
- Starting up (Chapter 7)
Describes startup procedures for the FM 357.
- Human-machine interface (Chapter 8)
Describes the available options for controlling and monitoring the FM 357 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 module.

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

- The SIMATIC S7
Installation Manual *S7-300 Programmable Controller, Hardware and Installation*
- Your programming device (PG)
- How to program with STEP 7
- Configuring the interface of an operator panel (e.g. OP 17)

FM 357 users

The information in this Manual is structured and represented in accordance with the field of application of the FM 357 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

CE marking

Our products comply with the requirements of EU Directive 89/336/EEC "Electromagnetic Compatibility" and the relevant harmonized European standards (EN).



The EC Declaration of Conformity in accordance with Article 10 of the EU Directive referenced above is contained in this manual (see Chapter B).

Contact person

If you should encounter problems in using this Manual or have any other queries, please contact the responsible department named on the query sheet at the back of the document.

Hotline

If you have any queries, please contact: Hotline: ++49-911 / 895 – 7000



Contents

1	Product Overview	1-1
1.1	The FM 357 in the S7-300 programmable controller	1-3
1.2	Module description	1-8
1.3	Overview of module functions	1-11
2	Fundamental Principles of Motion Control	2-1
3	Installation and Removal of the FM 357	3-1
3.1	Installation of the FM 357	3-3
3.2	Install firmware/firmware update	3-4
3.3	Removal and replacement of the FM 357	3-6
4	Wiring the FM 357	4-1
4.1	Wiring diagram of an FM 357	4-3
4.2	Connection of power supply	4-6
4.3	Description of drive interface	4-9
4.4	Connection of drive unit	4-15
4.5	Description of measuring system interface	4-19
4.6	Connecting encoders	4-23
4.7	Description of I/O interface	4-25
4.8	Wiring of front connector	4-28
4.9	Insertion and replacement of backup battery	4-31
5	Parameterization of the FM 357	5-1
5.1	Installation of "Parameterize FM 357"	5-3
5.2	Getting started with "Parameterize FM 357"	5-4
5.3	Adaptation to firmware	5-5
5.4	Parameterization data	5-7
5.4.1	Machine data (parameters)	5-9
5.4.2	User data	5-21
5.5	Menus of "Parameterize FM 357"	5-23
5.6	Settings of parameterization interface	5-27
6	Programming the FM 357	6-1
6.1	FB 1: RUN_UP – Basic function, startup section	6-5
6.2	FC 22: GFKT – Basic functions and operating modes	6-7

6.3	FC 24: POS_AX – Positioning of linear and rotary axes (CPU axis)	6-12
6.4	FB 2: GET – Read NC variable	6-16
6.5	FB 3: PUT – Write NC variable	6-22
6.6	FB 4: PI – Select program, acknowledge error	6-27
6.7	FC 5: GF_DIAG – Basic function, diagnostic alarm	6-31
6.8	FC 9: ASUP – Start of asynchronous subroutines	6-33
6.9	User data blocks (user DB)	6-36
6.9.1	User data block “NC signals”	6-37
6.9.2	User data block “Axis signals”	6-44
6.9.3	Description of signals	6-48
6.10	User handling procedures for controlling axes	6-60
6.11	Application examples	6-62
6.12	Technical data	6-66
7	Starting Up the FM 357	7-1
7.1	Installation and wiring	7-2
7.2	FM 357 power-up	7-3
7.3	Procedure for parameterization	7-4
7.4	Testing and optimization	7-6
8	Human-Machine Interface	8-1
8.1	Standard HMI for OP 17	8-3
8.2	Troubleshooting on OP 17 (example)	8-6
9	Description of Functions	9-1
9.1	Configuration	9-3
9.2	Encoders	9-8
9.2.1	Incremental encoders	9-10
9.2.2	Absolute encoders (SSI)	9-12
9.2.3	Stepper motor	9-14
9.3	Position control	9-15
9.4	Velocities and acceleration rates	9-24
9.5	Monitoring functions	9-30
9.5.1	Monitoring of movements	9-30
9.5.2	Encoder monitoring	9-35
9.5.3	Hardware and software limit switches	9-37
9.6	Referencing and alignment	9-39
9.6.1	Referencing with incremental encoders	9-41
9.6.2	Referencing for stepper motors without encoders	9-46
9.6.3	Alignment for absolute encoders	9-47
9.7	Output of M, T and H functions	9-49
9.8	Digital I/Os	9-52
9.8.1	Digital on-board inputs	9-52

9.8.2	Digital inputs/outputs on local P bus	9-53
9.9	Limit switching signals (software cams)	9-56
9.9.1	Parameterization	9-56
9.9.2	Generation of limit switching signals	9-59
9.9.3	Output of limit switching signals	9-61
9.10	Operating modes	9-62
9.11	NC program execution	9-64
9.12	Asynchronous subroutine (ASUB)	9-66
9.13	Motion coupling	9-69
9.13.1	Coupled motion	9-69
9.13.2	Gantry	9-72
9.13.3	Master value coupling	9-78
9.13.4	Overlaid motion in synchronized actions	9-84
9.14	Measurement	9-86
9.15	Travel to fixed stop	9-88
9.15.1	Parameter definition	9-89
9.15.2	Drive	9-91
9.15.3	Functional sequence	9-92
9.15.4	Further information	9-96
9.16	EMERGENCY STOP	9-97
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 format	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 dimension for rotary axes (DC, ACP, ACN)	10-15
10.2.5	Programming a polar coordinate (G110, G111, G112, RP, AP)	10-17
10.2.6	Dimensions inch and metric (G70, G71)	10-20
10.2.7	Plane selection (G17, G18, G19)	10-21
10.3	Zero offset (frames)	10-22
10.3.1	Settable zero offset (G54, G55, G56, G57, G500, G53)	10-22
10.3.2	Programmable zero offset (TRANS, ATRANS, ROT, AROT, RPL, MIRROR, AMIRROR)	10-24
10.4	Setting an actual value (PRESETON)	10-29
10.5	Programming axis movements	10-30
10.5.1	Programming of feedrates (F, FA, FL)	10-30
10.5.2	Feed interpolation (FNORM, FLIN, FCUB)	10-31
10.5.3	Linear interpolation with rapid traverse (G0)	10-34
10.5.4	Linear interpolation with feed (G1)	10-34
10.5.5	Positioning motions (POS, POSA, WAITP)	10-35
10.5.6	Circular interpolation (G2, G3, I, J, K, CR)	10-36

10.6	Spline (ASPLINE, CSPLINE, BSPLINE)	10-40
10.7	Path action	10-46
10.7.1	Exact stop (G60, G9), target range (G601, G602)	10-47
10.7.2	Continuous-path mode (G64, G641, ADIS, ADISPOS)	10-49
10.7.3	Acceleration patterns (BRISK, SOFT, DRIVE)	10-52
10.7.4	Programmable acceleration (ACC)	10-53
10.8	Dwell time (G4)	10-54
10.9	Motion coupling (TRAILON, TRAILOF)	10-54
10.10	Measurement	10-56
10.10.1	Block-specific measurement (MEAS, MEAW)	10-56
10.10.2	Axial measurement (MEASA, MEAWA)	10-58
10.11	Travel to fixed stop (FXST, FXSW, FXS)	10-60
10.12	Stop Ppreprocessor (STOPRE)	10-62
10.13	Working area limitations (G25, G26, WALIMON, WALIMOF)	10-62
10.14	M functions	10-64
10.15	H functions	10-66
10.16	Tool offset values (T functions)	10-67
10.17	R parameters (arithmetic parameters)	10-69
10.18	System variables (\$P_, \$A_, \$AC_, \$AA_)	10-72
10.19	Program jumps (GOTOF, GOTOB, LABEL, IF)	10-78
10.20	Subroutine system (L, P, RET)	10-80
10.21	Asynchronous subroutines (ASUB)	10-83
10.22	Synchronized actions	10-87
10.23	Oscillation	10-104
10.24	Master value coupling	10-108
10.25	Speed feedforward control (FFWON, FFWOF)	10-112
10.26	Overview of statements	10-113
11	Troubleshooting	11-1
11.1	Display by LEDs	11-3
11.2	Error messages and their effect	11-7
11.3	Error lists	11-9
A	Technical Specifications	A-1
B	EC Declaration of Conformity	B-1
C	List of Abbreviations	C-1
	List of Indices	Index-1

Bilder

1-1	Multi-tier configuration of a SIMATIC S7-300 with FM 357 (example)	1-3
1-2	System overview (diagrammatic)	1-5
1-3	Data storage concept	1-7
1-4	Position of interfaces and front-panel elements	1-8
1-5	Type plate of the FM 357	1-10
2-1	Servo system with converter, e.g. SIMODRIVE 611-A	2-1
2-2	Controlled stepper motor system with drive circuit, e.g. FM STEPDRIVE	2-2
2-3	Controlled stepper motor system with drive circuit, e.g. FM STEPDRIVE	2-2
4-1	Overview of connecting cable between FM 357 and servo drive (example)	4-3
4-2	Overview of connecting cable between FM 357 and stepper driv (example)	4-4
4-3	Module supply options	4-7
4-4	Position of X2 connector	4-9
4-5	Options for connecting signals on stepper motor interface	4-14
4-6	Connection of a SIMODRIVE 611-A drive unit	4-15
4-7	Connection of FM STEPDRIVE drive units	4-16
4-8	Assignment of connectors X3 to X6	4-19
4-9	Connection of encoders	4-23
4-10	Location of X1 connector	4-25
4-11	Wiring up the front connector	4-28
4-12	Connection overview for probes and proximity switches	4-30
4-13	Installing the backup battery	4-31
5-1	Overview of parameterization	5-1
5-2	Getting started with "Parameterize FM 357"	5-4
5-3	Adaptation to firmware	5-6
5-4	Online Edit selection dialog	5-8
5-5	Offline editing selection dialog	5-8
5-6	Machine data, e.g. controller data	5-10
5-7	Machine data	5-20
5-8	Entering values for R parameters	5-21
5-9	Entering values for zero offsets	5-21
5-10	Entering values for tool offsets	5-22
5-11	Input of values for NC programs	5-22
5-12	Settings of parameterization interface	5-27
6-1	Block diagram of communication between CPU and three FM 357	6-3
6-2	Timing diagram for FC 24	6-15
6-3	Timing diagram for the FC 24 (error case)	6-15
6-4	Timing diagram for FB 2	6-20
6-5	Timing diagram for FB 3	6-25
6-6	Timing diagram for FB 4	6-29
6-7	Timing diagram for FC 9	6-34
6-8	Example of auxiliary function	6-59
7-1	Operating and display elements for power-up	7-3
7-2	Startup interface (e.g. for "Automatic" mode)	7-6
7-3	Troubleshooting	7-7
7-4	Servicing data	7-7
7-5	Trace	7-8
7-6	Axis testing	7-10
8-1	Operator control and monitoring for the FM 357	8-1
8-2	Menu tree of operator interface on OP 17	8-4

8-3	Main screen MCS PIC_G	8-5
8-4	Input/output dialog	8-6
8-5	Symbol list-Text dialog	8-7
9-1	Rotary encoder on motor	9-11
9-2	Control loops	9-15
9-3	Overview of position controller	9-15
9-4	Jerk limitation on position controller level	9-16
9-5	Positive backlash (normal condition)	9-18
9-6	Negative backlash	9-18
9-7	Composition of the total compensation value	9-22
9-8	Velocity and acceleration plotted for brisk acceleration	9-26
9-9	Velocity and acceleration profiles with soft acceleration	9-27
9-10	Axial acceleration and velocity characteristic	9-28
9-11	Speed setpoint monitoring	9-34
9-12	Travel limits	9-37
9-13	Mounting a reference point switch (RPS)	9-41
9-14	Example for output of M, T and H functions	9-51
9-15	Limit switching signals for linear axis (minus cam < plus cam)	9-59
9-16	Limit switching signals for linear axis (plus cam < minus cam)	9-59
9-17	Limit switching signals for modulo rotary axis (plus cam – minus cam < 1805)	9-60
9-18	Limit switching signals for modulo rotary axis (plus cam – minus cam > 1805)	9-61
9-19	Execution of asynchronous subprograms	9-66
9-20	Example of scaling of master and slave axis positions	9-82
9-21	Example of synchronization	9-83
9-22	Example of travel to fixed stop	9-88
9-23	Hardware connections between FM 357, signal module and SIMODRIVE 611-A (FDD)	9-91
9-24	Diagram for “Fixed stop reached” with SIMODRIVE 611-A	9-94
9-25	Diagram showing “Fixed stop not reached” with SIMODRIVE 611-A	9-95
9-26	Diagram showing deselection “Fixed stop reached” with SIMODRIVE 611-A	9-95
9-27	EMERGENCY STOP sequence	9-98
10-1	Structure of statements with an address and numerical value	10-4
10-2	Block structure	10-6
10-3	Definition of axis directions	10-10
10-4	Machine and workpiece coordinate systems	10-11
10-5	Relationship between axis types	10-11
10-6	Absolute and incremental dimensioning	10-13
10-7	Move rotary axis across shortest path	10-15
10-8	Move rotary axis in positive direction to absolute position	10-16
10-9	Move rotary axis in negative direction to absolute position	10-16
10-10	Programming G110	10-18
10-11	Programming G110 (in polar coordinates)	10-18
10-12	Programming G111	10-18
10-13	Programming G112	10-19
10-14	Polar radius and polar angle	10-19
10-15	Plane and axis assignment	10-21
10-16	Settable zero offset G54 (shift and rotation)	10-23
10-17	Settable zero offset G57 (shift and mirror)	10-23
10-18	Directions of the angle of rotation	10-26
10-19	Order of rotation for three angle dimensions in one block	10-26

10-20	RPL – Shift, then rotate	10-27
10-21	RPL – Rotate, then shift	10-28
10-22	Mirror in X axis	10-28
10-23	Example of constant feed characteristic	10-32
10-24	Example of linear feed characteristic	10-32
10-25	Example of cubic feed characteristic	10-33
10-26	Example of feed interpolation	10-33
10-27	Linear interpolation with rapid traverse	10-34
10-28	Linear interpolation with feed	10-35
10-29	Direction of circle rotation in the planes	10-37
10-30	Example of center point and end point dimensions	10-38
10-31	Example of end point and radius dimensions	10-39
10-32	Spline interpolation	10-40
10-33	ASPLINE	10-41
10-34	CSPLINE	10-42
10-35	Conditions for ASPLINE and CSPLINE	10-43
10-36	BSPLINE, associated control polygon	10-45
10-37	Spline grouping, e.g. with three path axes	10-46
10-38	Block change depending on the size of the exact stop limit	10-48
10-39	Velocity-dependent machining of contour corners with G64	10-49
10-40	Continuous-path mode with rounding clearance: G641 with ADIS/ADISPOS	10-50
10-41	Comparison of velocity response in G60 and G64 with short paths	10-51
10-42	Acceleration characteristic with BRISK / SOFT / DRIVE	10-53
10-43	Working area limitations G25 and G26	10-63
10-44	Three-dimensional effect of tool length compensation	10-67
10-45	Effect of tool offset and zero offset in the G17 plane	10-68
10-46	Example of a program run with double subroutine call	10-81
10-47	Nesting depth	10-81
10-48	Working with ASUBs	10-86
10-49	Structure of motion synchronous actions	10-87
10-50	Execution of a synchronized action	10-103
10-51	Example of periodic and nonperiodic curve tables	10-109
10-52	Example of curve table definition	10-110
11-1	Troubleshooting	11-2
11-2	Status and error displays of the FM 357	11-3

Tabellen

1-1	Components of a positioning controller	1-6
1-2	Interfaces	1-9
1-3	Status and error displays	1-9
4-1	Connecting cables for a multi-axis controller with FM 357	4-5
4-2	Electrical parameters of load power supply	4-6
4-3	Assignment of the screw-type terminal	4-6
4-4	Pin assignment of the X2 connector	4-10
4-5	Electrical parameters of the setpoint signal	4-11
4-6	Electrical parameters of the relay contacts	4-11
4-7	Electrical parameters of the stepper drive signal outputs	4-13
4-8	Assignment of connectors X3 to X6	4-19
4-9	Electrical parameters of encoder power supply	4-21
4-10	Maximum cable length as a function of encoder power supply	4-22
4-11	Maximum cable length as a function of transfer frequency	4-22
4-12	Pin assignment of the X1 connector	4-26
4-13	Electrical parameters of digital inputs	4-27
4-14	Electrical parameters of relay contact NCRDY	4-27
5-1	Machine data (parameters)	5-11
5-2	Menus of "Parameterize FM 357"	5-23
6-1	Standard function blocks for the FM 357	6-2
6-2	FB 1 parameters	6-5
6-3	Signals, status FC 22	6-10
6-4	FC 22 troubleshooting, GF_ERROR	6-11
6-5	FC 24 troubleshooting	6-14
6-6	FB 2 parameters	6-17
6-7	FB 2 troubleshooting	6-18
6-8	FB 3 troubleshooting	6-24
6-9	FB 4 troubleshooting	6-28
6-10	SELECT parameterization	6-29
6-11	Acknowledgement of errors (CANCEL)	6-30
6-12	Diagnostic alarms	6-31
6-13	User DB "NC signals"	6-37
6-14	Control signals for user DB "NC signals"	6-48
6-15	Control signals for user DB "Axis signals"	6-52
6-16	Checkback signals for user DB "NC signals"	6-54
6-17	Checkback signals for user DB "Axis signals"	6-55
6-18	Read data set for user DB "NC signals"	6-57
6-19	Read data set for user DB "Axis signals"	6-57
6-20	Write data set for user DB "NC signals"	6-58
6-21	Write data set for user DB "Axis signals"	6-58
6-22	Auxiliary functions for user DB "NC signals"	6-59
6-23	User handling procedures for controlling axes	6-60
6-24	Configuration	6-66
7-1	Installation and wiring checklist	7-2
7-2	Settings with the start-up switch of the FM 357	7-3
7-3	Parameterization checklist	7-5
9-1	Differences between FM357-L and FM357-LX	9-1
9-2	Parameters for absolute encoders	9-12
9-3	Position monitoring time	9-31
9-4	Features of monitoring systems for static limits	9-37
9-5	Parameters for referencing	9-43

9-6	H function parameters without group assignment	9-50
9-7	Digital I/Os on the FM 357	9-52
9-8	Digital inputs/outputs on local P bus	9-53
9-9	Status of digital inputs	9-55
9-10	Disabling digital outputs	9-55
9-11	Status of digital outputs	9-56
9-12	Cam position parameters	9-57
9-13	Software cam minus/plus	9-61
9-14	Operating modes and their properties	9-62
9-15	Typical program run	9-64
9-16	Program states	9-65
9-17	Gantry parameters	9-72
9-18	Assignment of gantry interface signals for leading and synchronized axes	9-73
9-19	Effect of individual interface signals on leading and synchronized axes	9-74
9-20	Offset and scaling of master and slave axis positions	9-81
9-21	Parameters for travel to fixed stop	9-89
10-1	Operators and arithmetic functions	10-70
10-2	Compare operators	10-71
10-3	System variables	10-73
10-4	Operators in synchronized actions	10-97
10-5	System variables	10-98
10-6	Overview of statements	10-113
11-1	Status and error displays	11-4
11-2	Summary of LED error displays	11-5
11-3	Error list	11-10



Product Overview

1

What can the FM 357 do?

The FM 357 is a microprocessor-based multi-axis module for the control of servo and/or stepper drives.

The module has one channel and can control up to four axes.

It is a high-performance module for servo-controlled positioning or positioning with a stepper drive, and can be used for individual or synchronized axes.

It can operate rotary and linear axes.

The FM 357 has a variety of operating modes.

Two firmware variants are available with product version 2 and later of the FM 357. In addition to the basic version of FM357-L, a functionally expanded variant, FM357-LX, is also available (see Chapter 9, Description of Functions).

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 user data.

- Data backup with backup battery
- Data backup on memory card (option)

Where can the FM 357 be used?

The FM 357 can be used for both simple positioning tasks and complex traversing profiles with exacting accuracy requirements in axis groupings operating by interpolation or in synchronism.

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

Section overview

Section	Title	Page
1.1	The FM 357 in the S7-300 programmable controller	1-3
1.2	Module description	1-8
1.3	Overview of module functions	1-11

1.1 The FM 357 in the S7-300 programmable controller

How is the FM 357 linked into the S7-300?

The FM 357 has been designed as a SIMATIC S7-300 function module.

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 may run up to four racks with as many as eight bus stations each (see Figure 1-1).

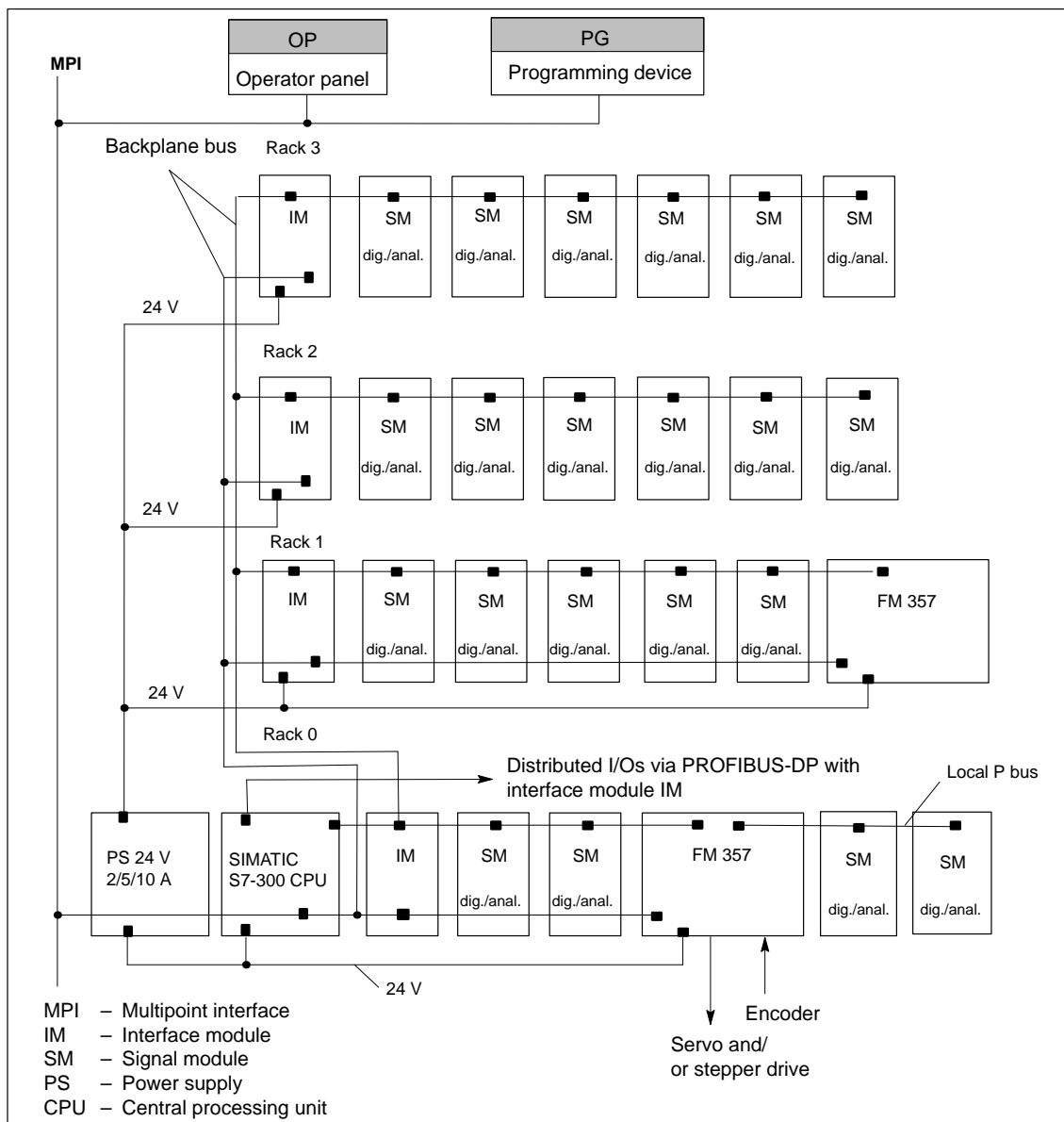


Figure 1-1 Multi-tier configuration of a SIMATIC S7-300 with FM 357 (example)

Single-tier configuration

A single-tier configuration consists of the S7-300 CPU, the FM 357 and a maximum of seven other modules (SM, FM).

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

The FM 357 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 rack 0 to the right of the S7-300 CPU. Eight modules (SMs, FMs and the FM 357) can be installed alongside the interface module.

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

A maximum of three FM 357 modules may be connected to a CPU.

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*

Spatially distributed arrangement

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 and, in a four-tier configuration, at up to 30 m.

Local P bus

The FM 357 is capable of covering a local bus segment. All modules installed to the right of the FM 357 can then only be addressed as high-speed I/Os by the FM 357 when the latter is powered up.

Distributed I/Os via PROFIBUS-DP

The FM 357 can be operated as a distributed module on S7 300/400 systems via PROFIBUS-DP and the ET 200M I/O device.

System overview

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

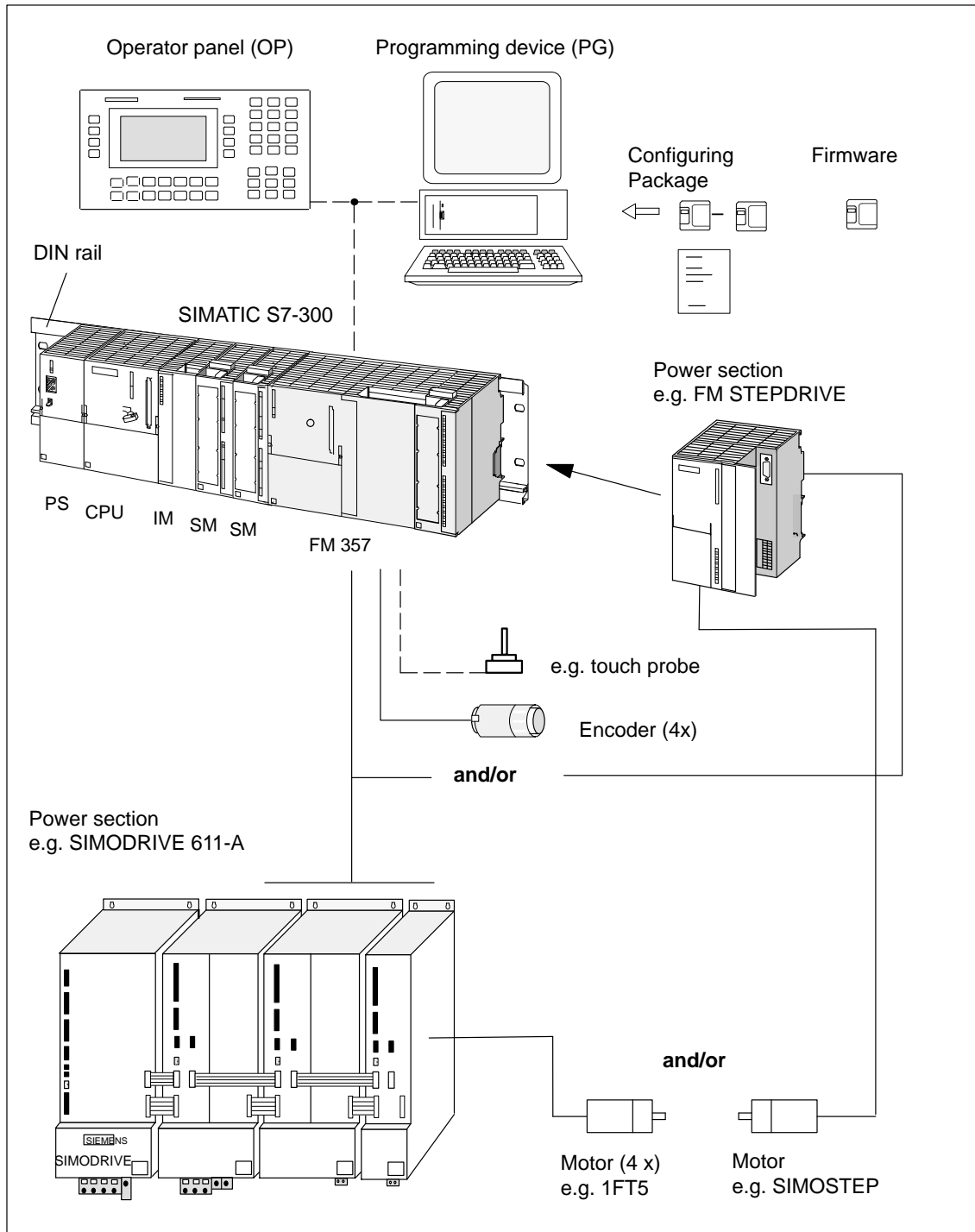


Figure 1-2 System overview (diagrammatic)

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	... is the multi-axis module. It is controlled by the S7-300 CPU.
CPU	... executes the user program; powers the S7-300 backplane bus at 5 V; communicates with the programming device and control panel via the MPI interface and with the FM 357 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.
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.
Power section	... actuates the motor.
Motor	... drives the axis.
Encoder	... the path measurement system that detects the current position of the axis. By comparing the actual position with the applicable setpoint position, the FM 357 immediately detects discrepancies and attempts to compensate for them.
Configuring Package	... includes: <ul style="list-style-type: none"> • A manual • 3 1/2" diskettes with: <ul style="list-style-type: none"> – FM 357 standard function blocks – The "Parameterize FM 357" parameterization tool. – Preconfigured interface for the COROS device OP 17 • 3 1/2" diskette with: NC variable selector
Firmware	3 1/2" diskette with: <ul style="list-style-type: none"> • Installation instructions (file: readme.txt) • Installation routines • FM 357 firmware

System overview, data handling

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

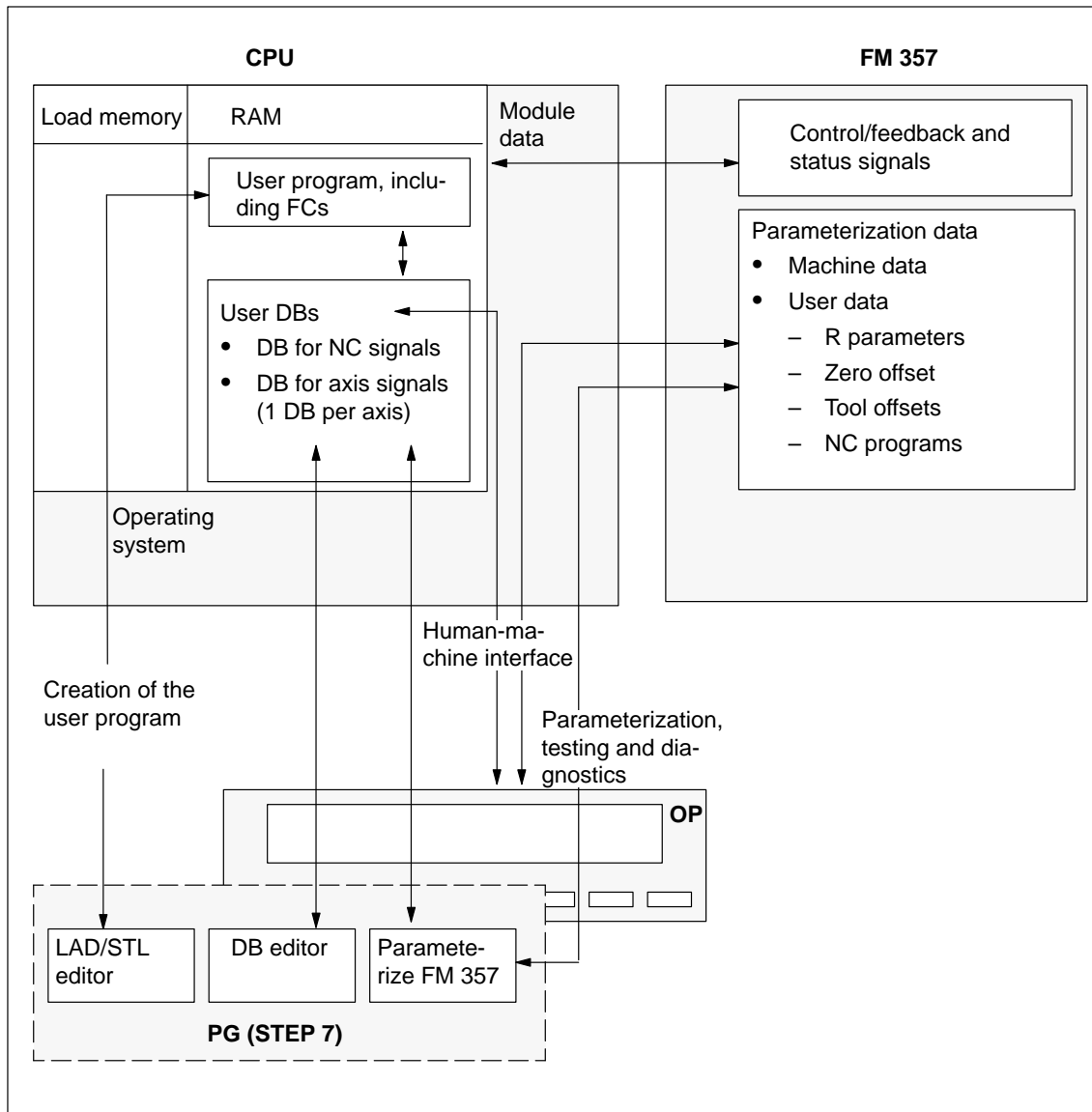


Figure 1-3 Data storage concept

1.2 Module description

View of FM 357

Figure 1-4 shows the FM 357 module with its interfaces and front-panel elements (error and status displays).

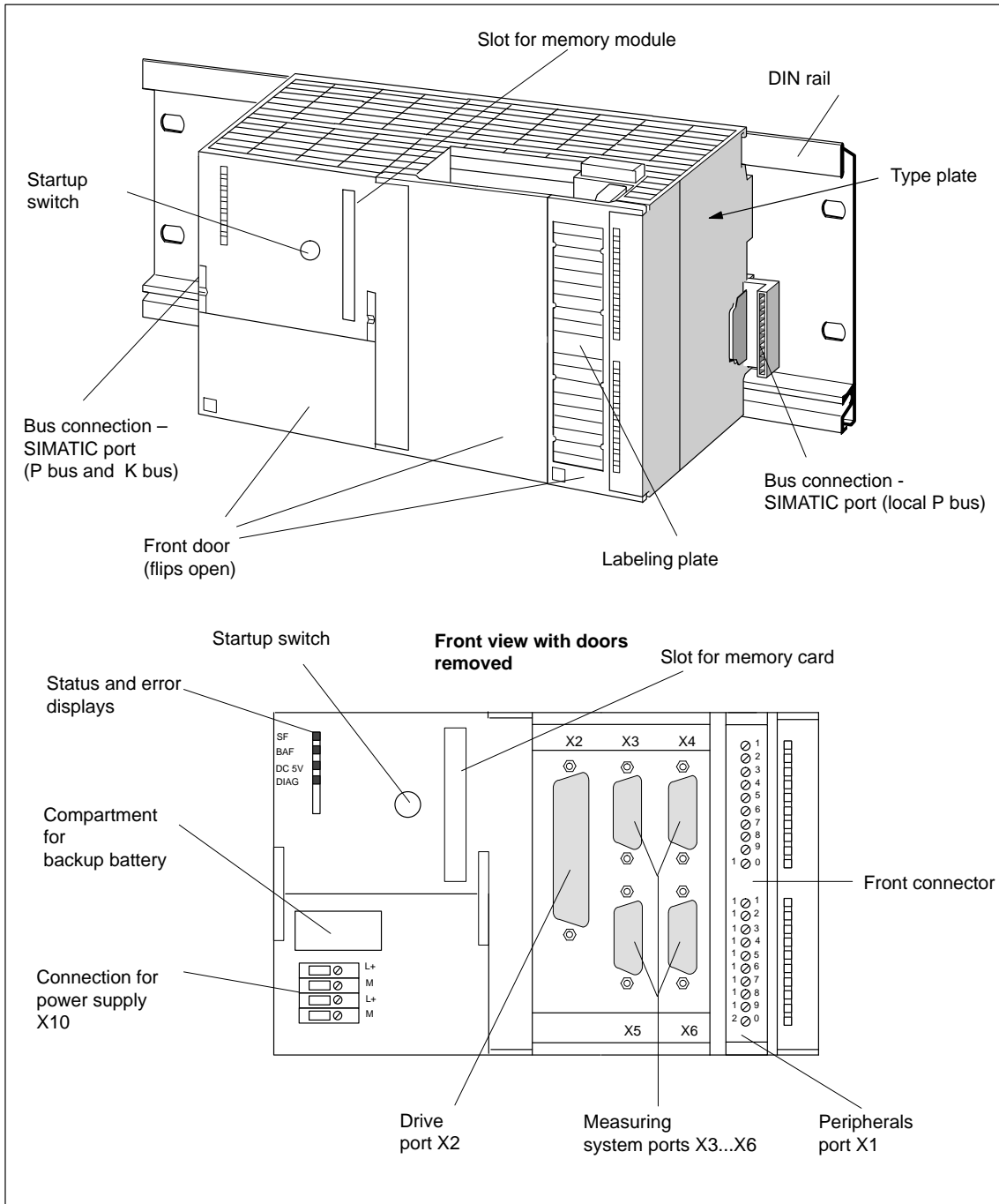


Figure 1-4 Position of interfaces and front-panel elements

Interfaces

Table 1-2 describes the interfaces and their meaning.

Table 1-2 Interfaces

Interfaces	Description
Bus link SIMATIC interface	<ul style="list-style-type: none"> Rear-panel connector (left, see Figure 1-4) for connection of FM 357 to other S7-300 modules via the S7 backplane bus (P and K busses) Rear-panel connector (right, see Figure 1-4) for connecting the FM 357 to further S7-300 modules for extending the functionality of the FM 357 (local P bus)
Drive interface	50-pin male sub D connector (X2) for connecting the power sections for up to four analog and/or stepper drives
Measuring system interface	15-pin female sub-D connector (X3 to X6) for connecting encoders (max. 4)
I/O device interface	20-pin male sub D connector (X1) for connecting the high-speed inputs (including the probe) and for wiring the NC-READY relay
Power supply connection	4-pin screw-type terminal connection (X10) for connecting the 24 V load power supply
Memory submodule interface	68-pin PCMCIA card connector for memory card

LED displays

Four LED displays are arranged on the front panel of the FM 357. 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. (see Troubleshooting, Chapter 11)
DC 5V (green) – logic power supply	This LED indicates that the hardware is ready for operation. (see Troubleshooting, Chapter 11)
DIAG (yellow) – diagnostics	This LED indicates various diagnostic states (flashing). (see Troubleshooting, Chapter 11)
BAF (red) – battery fault	If this LED flashes, you need to change the battery (see Section 4.9 and Chapter 11).

Operator elements

Startup 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 1-5 explains all the information displayed on the type plate.

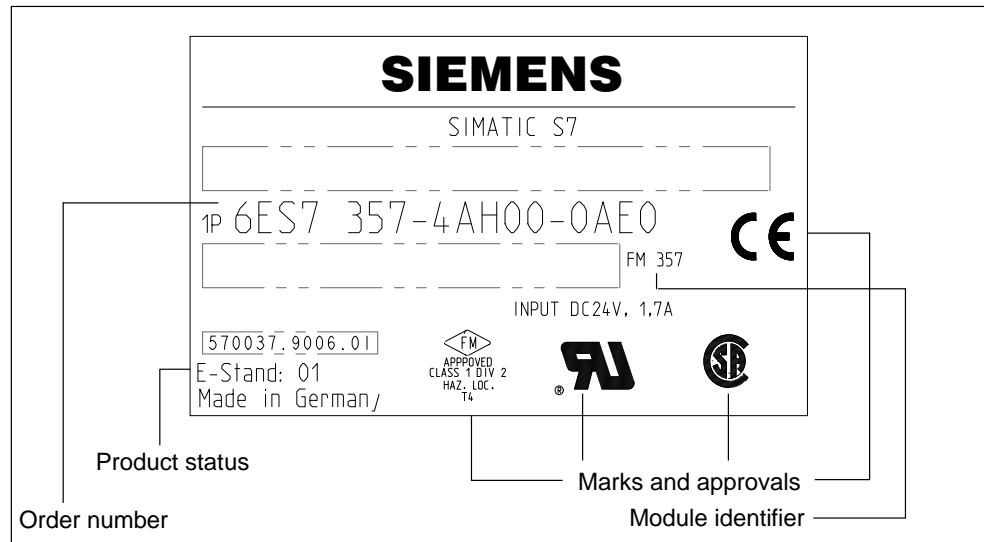


Figure 1-5 Type plate of the FM 357

1.3 Overview of module functions

Overview

The following main functions are implemented in the FM 357 module:

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

Mode control

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

The following modes are available on the FM 357:

- 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 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 inputs

You can use the digital inputs freely to suit your own application.

You might connect:

- Switch for reference point approach
- Touch probes

The switching function is assigned to a given I/O number by way of the machine data.

Software limit switches

The working area is automatically monitored after axis synchronization. This can be done by means of software limit switches for specific axes.

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

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

Data storage on memory card

The memory card is an FM 357 option. You can use this option for:

- Backing up your system software and user data
- Series startup of FM 357 in distributed configuration via PROFIBUS-DP
- Replacing modules without a programming device

Local P bus

The local P bus is used to extend the functionality of the FM 357. Digital I/O modules can be connected.



Fundamental Principles of Motion Control

2

Position-controlled motion control for servo axes

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

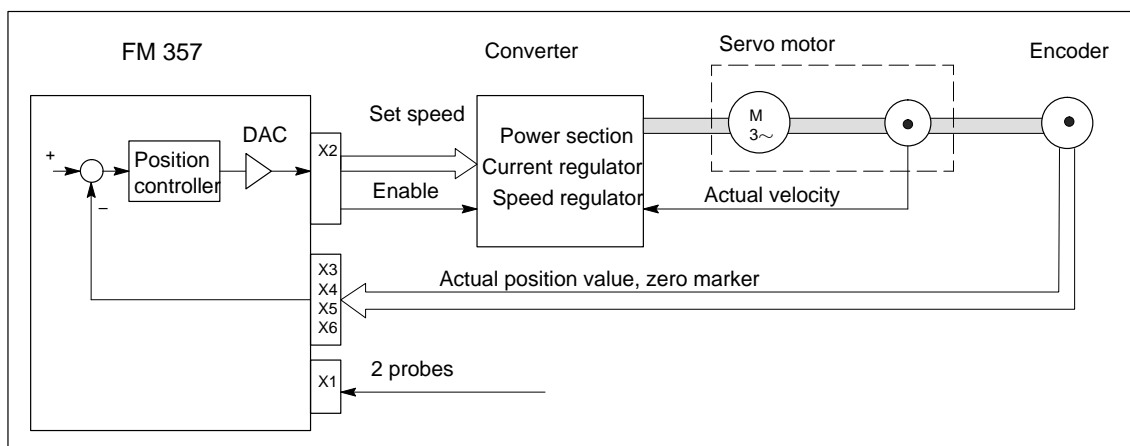


Figure 2-1 Servo system with converter, e.g. SIMODRIVE 611-A

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.

Stepper motor control

In addition to its analog setpoint outputs, the FM 357 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.

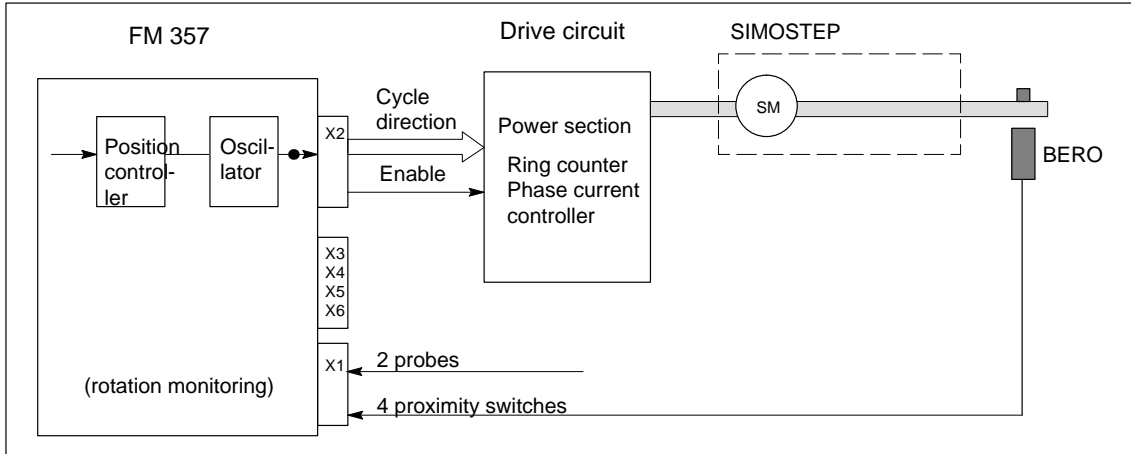


Figure 2-2 Controlled stepper motor system with drive circuit, e.g. FM STEPDRIVE

Stepper motor control as a function of position

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

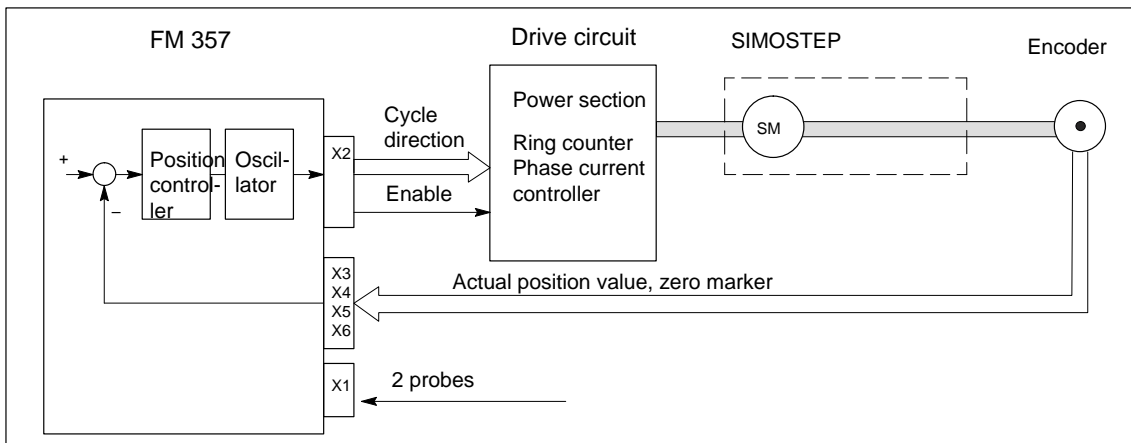


Figure 2-3 Controlled stepper motor system with drive circuit, e.g. FM STEPDRIVE



Installation and Removal of the FM 357

Overview

The FM 357 multi-axis module is integrated as an I/O module into a SIMATIC S7-300 control.

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

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 location of the FM 357 module is dependent on the mechanical configuration of your control. You must follow the rules below:

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

The FM 357 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 requires 100 mA of this amount.

Extension of functionality via local bus

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

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 module in a plant or system is subject to important safety rules which you must observe.

These rules and specifications are described in the manual *S7-300 Programmable Controller, Hardware and Installation*.

Section overview

Section	Title	Page
3.1	Installation of the FM 357	3-3
3.2	Installing firmware/firmware update	3-4
3.3	Installation and removal of the FM 357	3-6

3.1 Installation of the FM 357

Rules

No particular protective measures (ESD guidelines) need be taken for the installation of the FM 357.



Warning

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

Tool required

4.5 mm screwdriver

Procedure

Please proceed as follows to install the FM 357:

1. The FM 357 is supplied with a bus link. Plug this into the bus plug of the module to the left of the FM 357. (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.

If the FM 357 is the last module in the rack, do not connect this bus connector.

2. Engage the FM 357 on the mounting rail and lower it into place.
3. Tighten the screws on the FM 357 (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*.

Note

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 Install firmware/firmware update

Preconditions for central configuration

In order to install or replace (with new software version) the firmware of the FM 357, you will need the following:

- The supplied diskette containing:
 - Installation instructions (file: readme.txt)
 - Installation routines
 - FM 357 firmware
- A PG/PC with an
 - MPI interface and MPI connecting cable with free space (as specified in readme.txt) on the hard disk.
 - "Windows 95" operating system and corresponding STEP 7 program (version 3.1 or later).

Installation

A link must be made between the PG/PC and the S7-300 CPU before the firmware can be installed (see Figure 4-1 or 4-2).

Switch the CPU to the STOP state.

Set the startup switch on the FM 357 to position 2.

Note

The system is ready to receive the update when the red LED "SF" starts to flash cyclically.

Install the software as follows:

1. Insert the diskette containing the firmware in the diskette drive of your PG/PC.
Read file "readme.txt"!
2. Start the file named UPDFM357.EXE
3. Follow the instructions of the installation routine.

Result: The **Transfer firmware** dialog appears.

The remainder of the firmware installation or update procedure is described in file "readme.txt" (installation instructions).

Note

Before you install a new software version, you must back up **all** FM 357 data (e.g. machine data, user data), which do not belong to the firmware, on a data storage medium.

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

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

Distributed configuration

The firmware on FM 357 modules in a distributed configuration cannot be updated via the MPI interface of the PG/PC. To do this, you will need to create a memory card on the PG (for instructions, see file readme.txt)

Firmware update from memory card

Please proceed as follows:

1. With the control switched off, insert the memory card.
2. Set startup switch to position 6
3. Switch control "ON" → The system software and 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. Remove the memory card from the FM 357.
6. Set the startup switch on the FM 357 to position 1
7. Control "ON" → Wait for control to power up with defaults (approx. 3 min.)
8. Switch control "OFF"
9. Set the startup switch on the FM 357 to position 0
10. Control "ON" → The control powers up with the new firmware

3.3 Removal and replacement of the FM 357

Overview

The FM 357 module must be replaced as a complete unit.



Warning

It is only possible to replace the FM 357 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

Note

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

Data backup on memory card

The memory card is an FM 357 option and can be purchased with the control or at a later date.

All user data and the firmware are stored.

Procedure for backing up data:

1. Switch control "OFF"
2. Insert the memory card in the FM 357 module.
3. Set the startup switch on the FM 357 to position 0 or 1
4. Switch control "ON" and wait for it to power up
5. Set the startup switch on the FM 357 to position 3

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

6. LED "DIAG" flashes three times to indicate that all data have been backed up successfully.

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

7. Switch control "OFF"
8. Remove the memory card from the FM 357.

Note:

No message is output to confirm that data have been backed up (memory dump) on the memory card.

A battery error must not be active.

Remove a defective module

Please proceed as follows to remove the FM 357:

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.
5. Undo the mounting 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. 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 backed up user data from the memory card.

Reading in backed up data from the memory card

Proceed as follows:

1. With the control switched off, insert the memory card.
2. Set startup switch to position 6
3. Switch control "ON" → The system software and 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. Remove the memory card from the FM 357.
6. Set startup switch to position 0
7. Control "ON" → The control powers up with the firmware and the backup data from the memory card.



Wiring the FM 357

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 STOP strategy according to applicable engineering standards (e.g. European standards EN 60204, EN 418 and related 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, in order to identify sources of danger affecting the complete system.

Further references

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

- Guidelines for handling of electrostatic sensitive devices (ESDs): Appendix B.
- Configuring the electrical installation: Chapter 4

As a further source of information on the subject of EMC guidelines, we would recommend the description: *Equipment for Machine Tools, EMC Guidelines for WS/ WF Systems*, Order number: 6ZB5 440-0QX01-0BA1.

Standards and regulations

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

Section overview

Section	Title	Page
4.1	Wiring diagram of an FM 357	4-3
4.2	Connecting the power supply	4-6
4.3	Description of the drive port	4-9
4.4	Connecting the drive unit	4-15
4.5	Description of the measurement system port	4-19
4.6	Connecting the encoders	4-23
4.7	Description of the I/O port	4-25
4.8	Wiring up the front connector	4-28
	Installing and changing the backup battery	4-31

4.1 Wiring diagram of an FM 357

FM 357 with servo drive

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

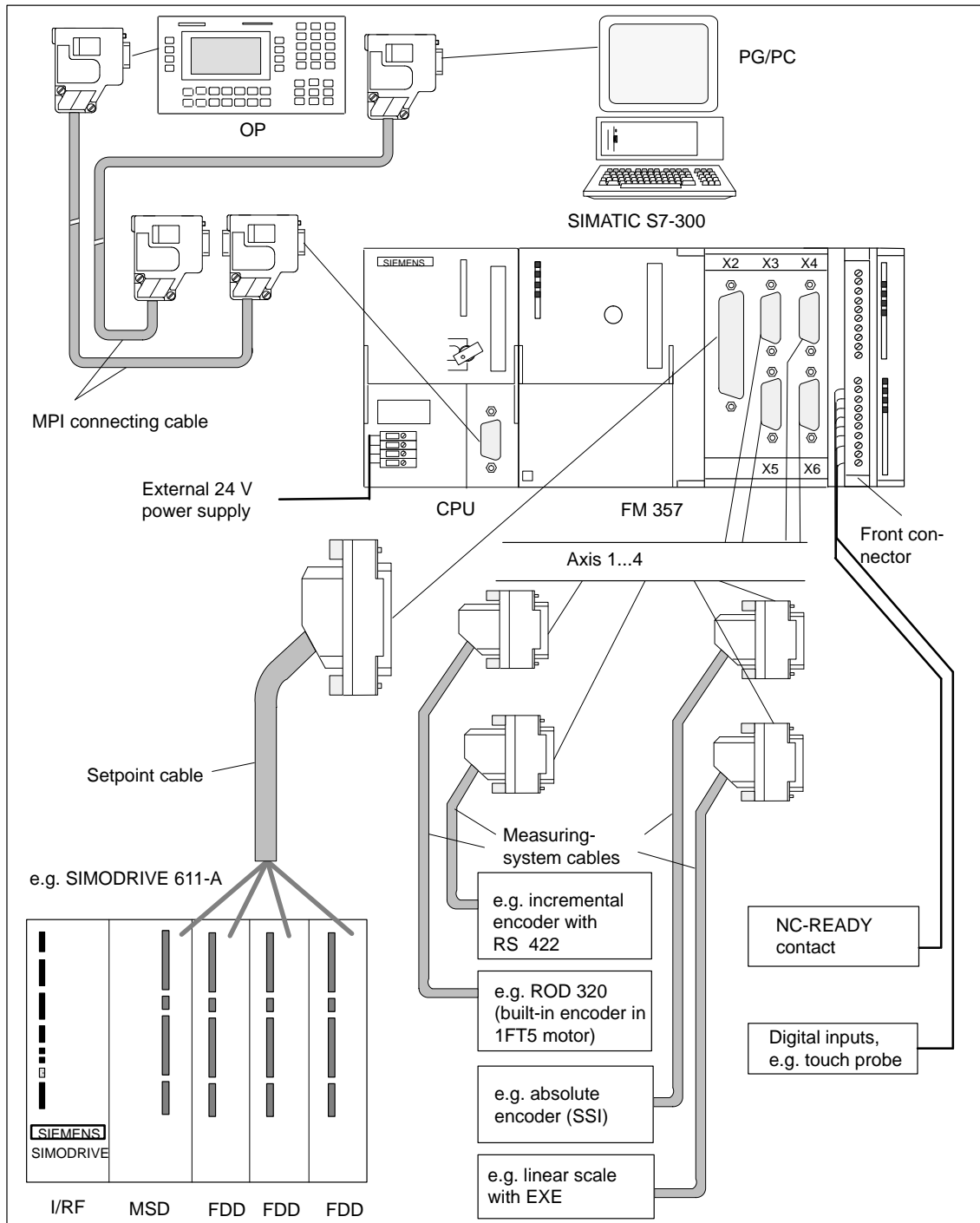


Figure 4-1 Overview of connecting cable between FM 357 and servo drive (example)

FM 357 with stepper drive

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

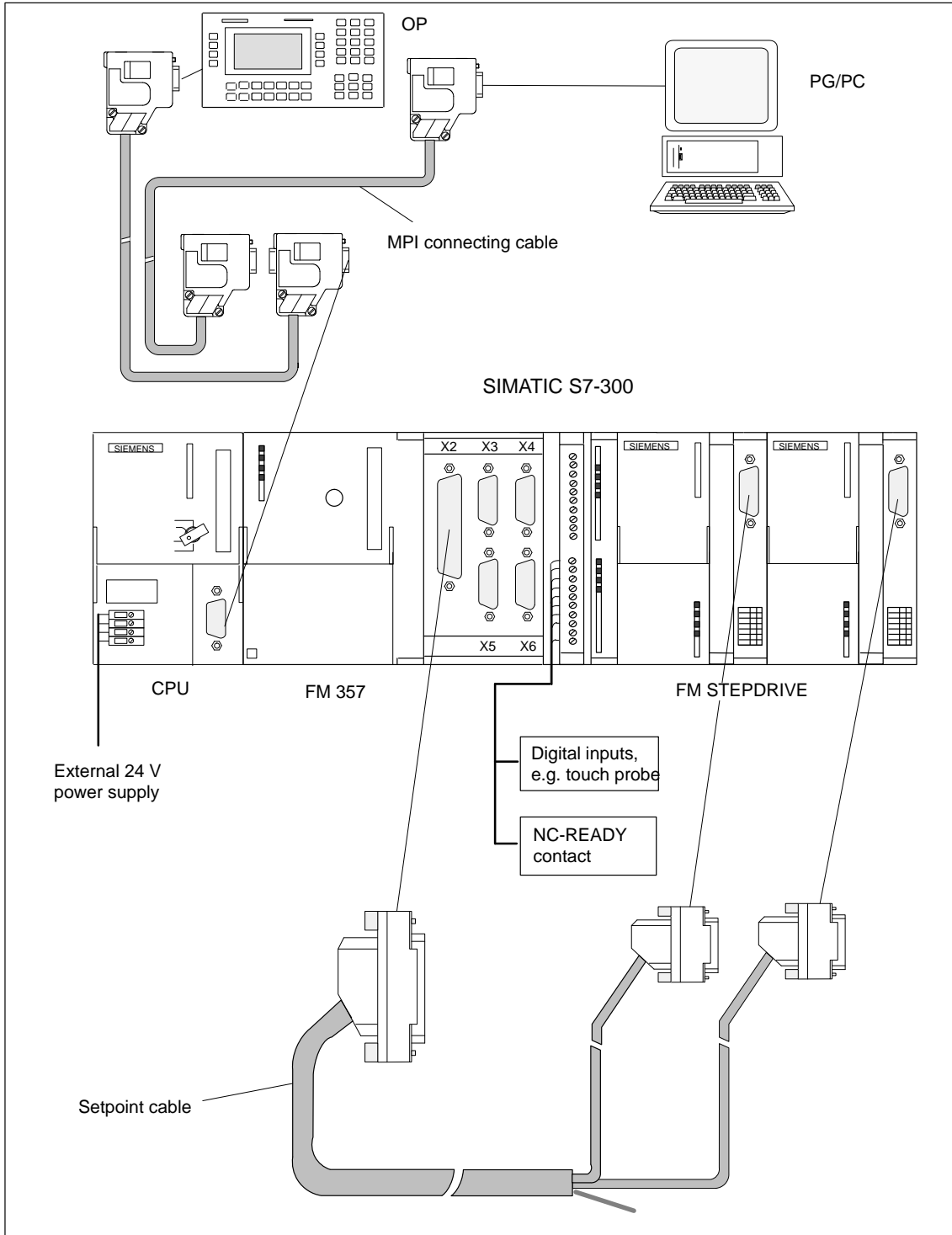


Figure 4-2 Overview of connecting cable between FM 357 and stepper drive (example)

Connecting cable

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

Table 4-1 Connecting cables for a multi-axis controller with FM 357

Type	Order No.	Description
MPI connecting cable	See manual <i>S7-300 Programmable Controller, Hardware and Installation</i>	Connection between OP, PG and S7-400 CPU
Setpoint cable	6FX2 002-3AD01-1□□□ see <i>Catalog NC Z</i> Order No.:E86060-K4490-A001-A5-7600	Connection between FM 357 and SIMODRIVE 611-A ± 10 V
Setpoint cable	6FX2 002-3AD02-□□□□ see <i>Catalog NC Z</i> Order No.:E86060-K4490-A001-A5-7600	Connection between FM 357 and stepper drive
Measuring system cable	6FX2 002-2CD01-1□□□ see <i>Catalog NC Z</i> Order No.:E86060-K4490-A001-A5-7600	Incremental encoder with RS 422 and FM 357 (EXE with linear scale)
Measuring system cable	6FX2 002-2CE01-□□□□ see <i>Catalog NC Z</i> Order No.:E86060-K4490-A001-A5-7600	ROD 320 encoder with 1FT5 motor and FM 357
Measuring system cable	6FX2 002-2CC01-□□□□ see <i>Catalog NC Z</i> Order No. :E86060-K4490-A001-A5-7600	Connection of absolute encoder (SSI) and FM 357

Front connector

To wire up the digital inputs/outputs, you will require a 20-pin, screw-type front connector. It must be ordered separately.

Order No.: 6ES7 392-1AJ00-0AA0

Catalog ST 70, Order No. E86060-K4670-A101-A3

see *Catalog NC 60.1*, Order No. E86060-K4460-A101-A5-7600

4.2 Connection of 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

The 24 V DC voltage must be generated as a functional extra-low voltage with protective separation (according to EN 60204-1, Section 6.4, PELV).

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.

Note

The FM 357 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

Mains buffering of 20 ms is guaranteed when a PS 307 is used.

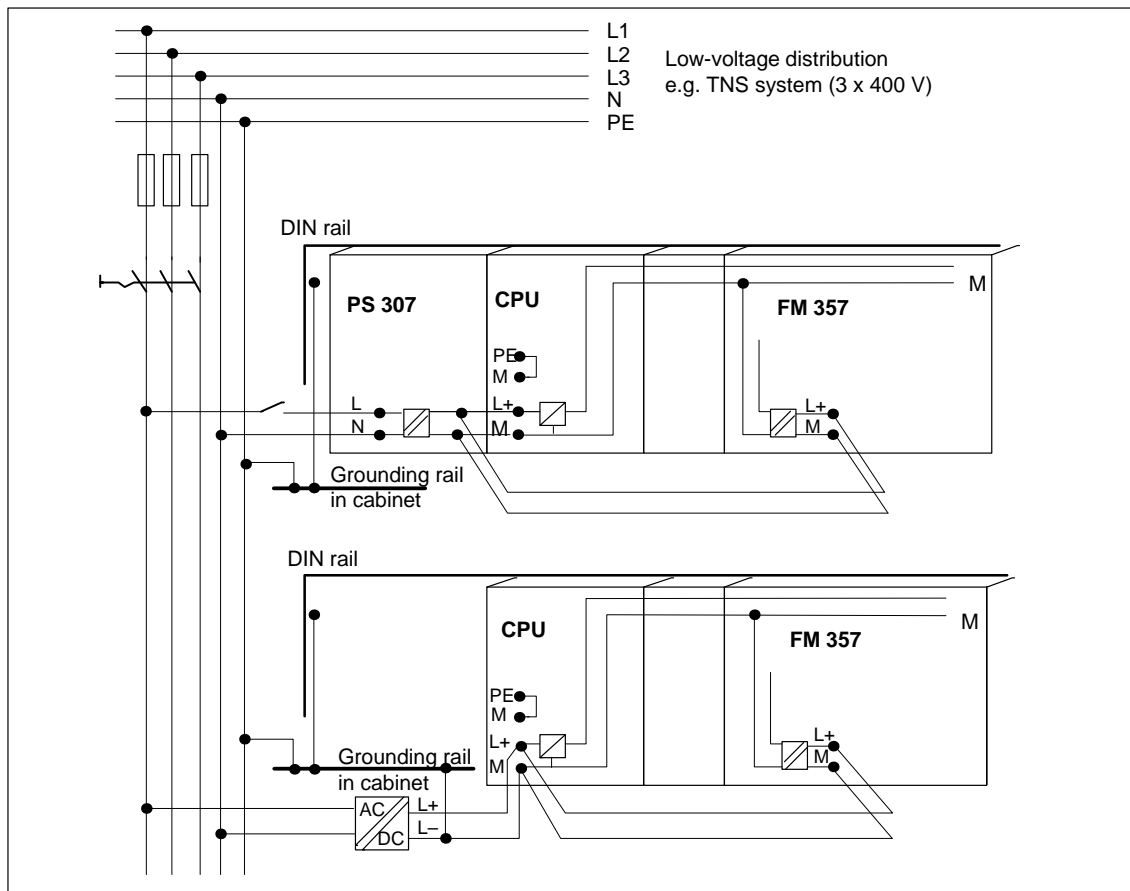


Figure 4-3 Module supply options



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. 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 60 to 80 Ncm.
4. Make the connection to the power supply unit (e.g. PS 307).

Note

You can use the top or bottom pair of terminals. The remaining pair of terminals can be used to supply power to other peripherals or modules.

Polarity reversal protection

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

Note

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 Description of drive interface

Connector to drive unit

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

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

Position of connector

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

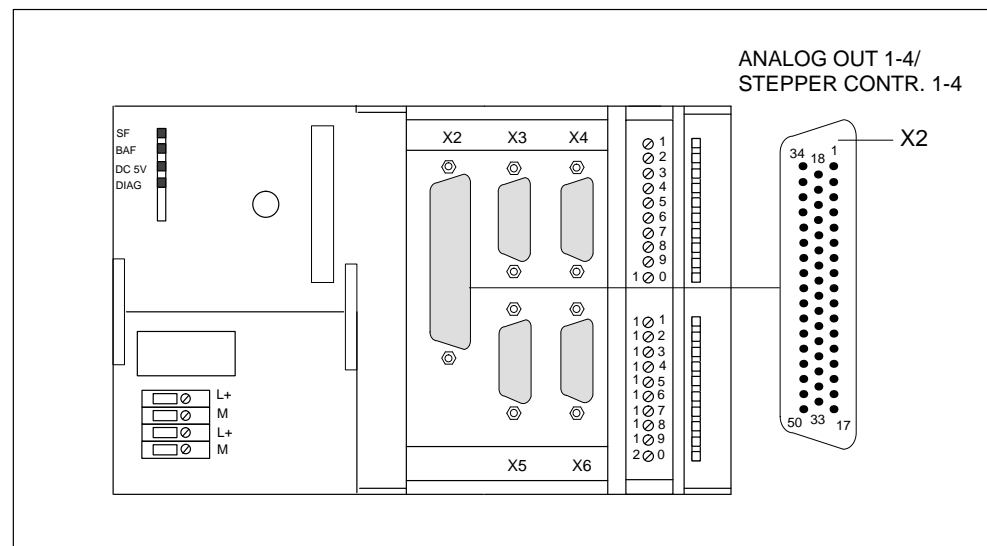


Figure 4-4 Position of X2 connector

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 has powered up, the RF signal to the drive is set as soon as controller enable RFG (user DB, "Axis signals" DBX12.1) is signalled by the user program.

Signal parameters

The setpoint is output as an analog differential signal.

Table 4-5 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 4-6 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”

Note

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 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:

- Disable pulse input
- Switch off power to motor
- Reset ring counter
- Erase error messages

Note

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.

All outputs are electronically protected against shorting and thermal overload.

Figure 4-5 shows various options for connecting signals. Table 4-7 gives a summary of the electrical data of the interface output signals.

Table 4-7 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		625	kHz	

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

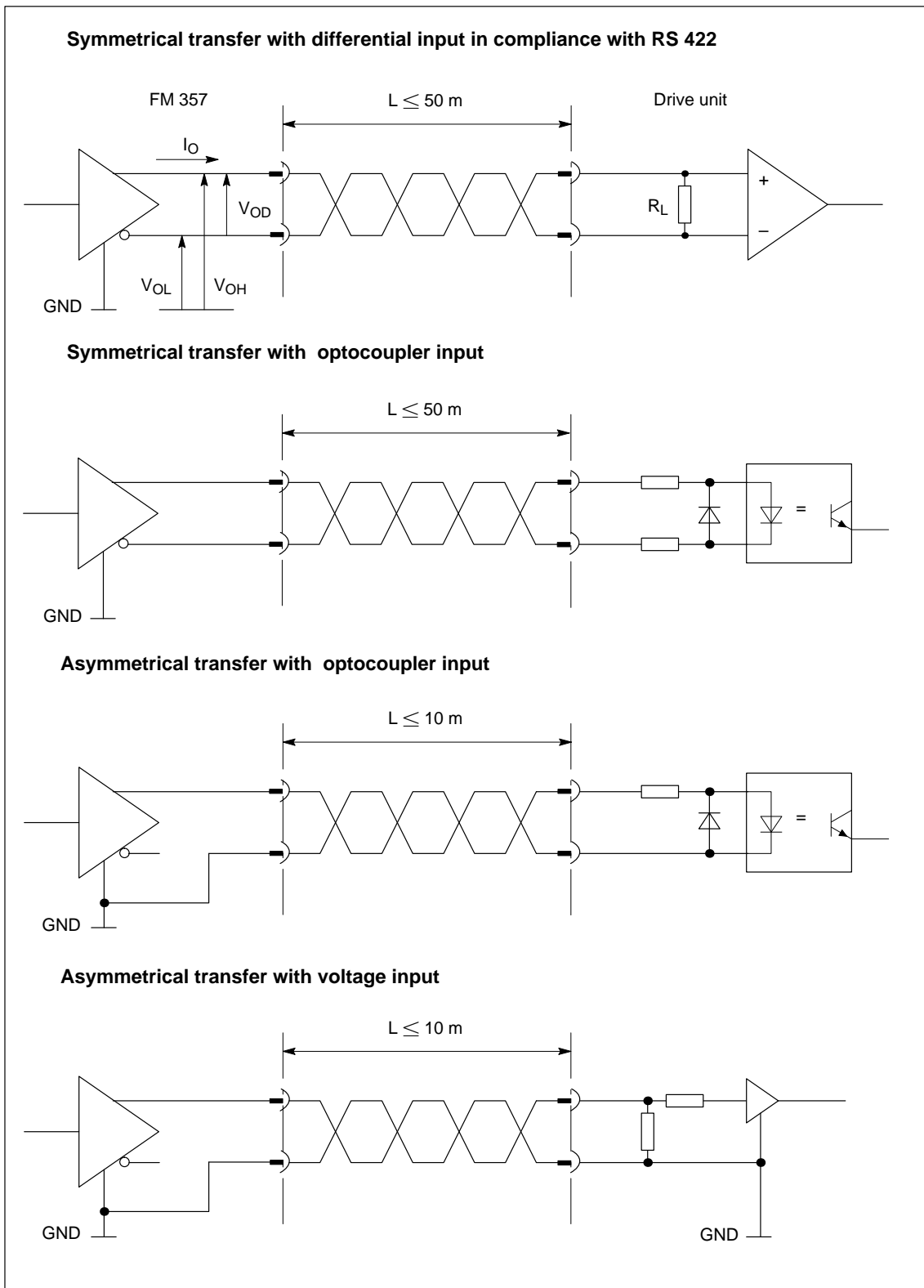


Figure 4-5 Options for connecting signals on stepper motor interface

4.4 Connection of drive unit

Connection of the connecting cable

Please note the following:

Note

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.

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

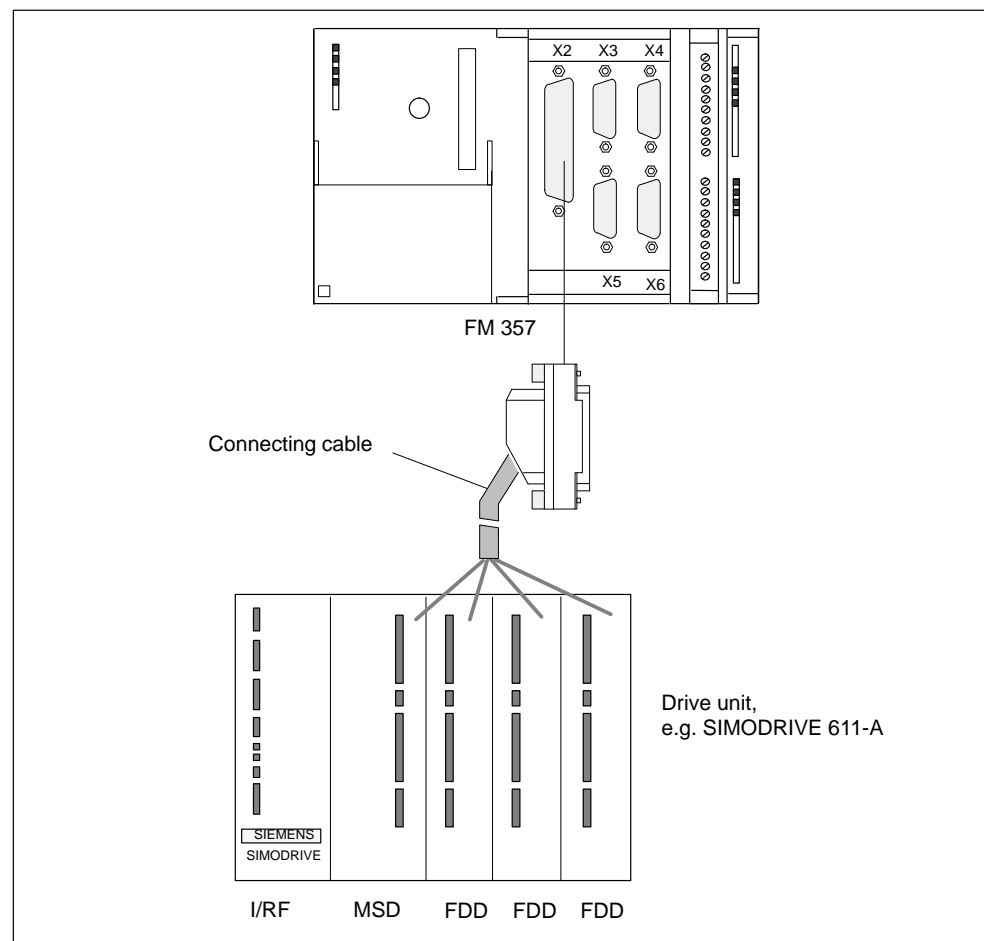


Figure 4-6 Connection of a SIMODRIVE 611-A drive unit

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

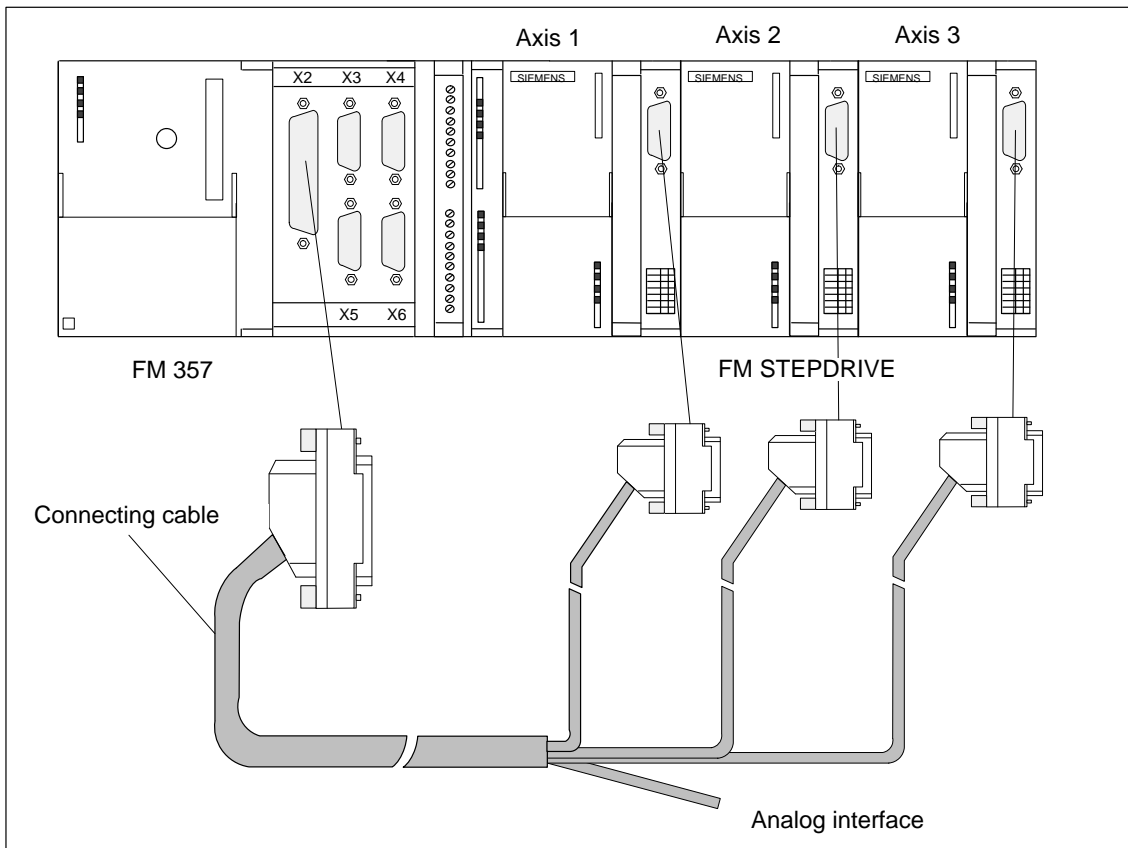


Figure 4-7 Connection of FM STEPDRIVE drive units

Connection of analog drives

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 and insert the sub-D socket connector in the module.
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*

Connection of stepper drives

Please proceed as follows:

1. Connect the free end of the cable to the terminals on a terminal distributor or make your own cable set with connectors in accordance with the specification of the power section manufacturer. Make the connection to the drive unit.
2. Open the front door and insert the sub-D socket connector in the module.
3. Use the thumbwheel screws to lock the connector in place. Close the front door.

Note

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 4.3 of the FM 357 Manual.

Connecting cable

The connecting cable is a prefabricated cable for three stepper motor drive units and a drive with an analog interface.

The connecting cable is available in a variety of lengths.

see *Catalog NC Z*

Note

With this cable, you can operate three stepper motors on axes 1 to 3. Axis 4 is assigned to an analog interface.

Details of further cables, e.g. for four stepper motor axes, are available on request.

Mixed operation of analog and stepper drives

Proceed as described for the connection of 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.

Connecting cable

The connecting cable is a prefabricated cable for three stepper motor axes and an axis with an analog interface.

The connecting cable is available in a variety of lengths.

see *Catalog NC Z*

Note

This cable assigns the analog interface to axis 4. Please remember this when you configure your controller.

Details of cables for other configurations are available on request.

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

4.5 Description of measuring system interface

Socket connectors for encoder

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

Position of socket connectors

Figure 4-8 shows the mounting position and designation of the socket connector on the module.

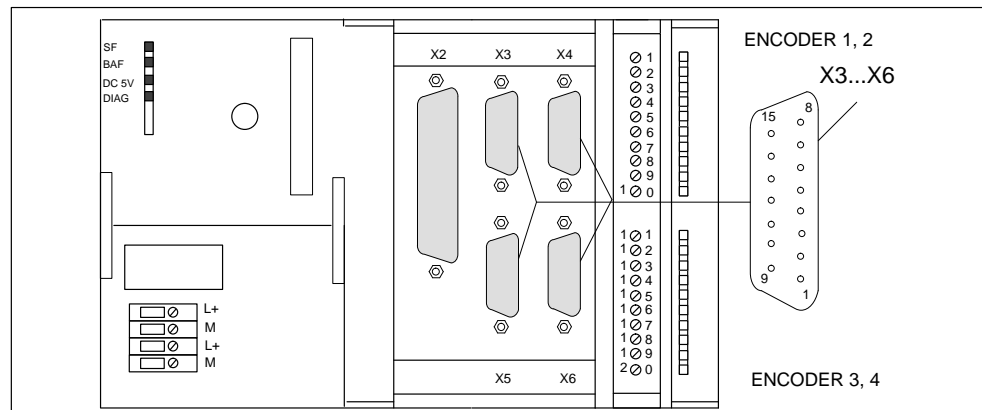


Figure 4-8 Assignment of connectors X3 to X6

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 4-8 Assignment of connectors X3 to X6

Pin	Encoder		Type	Pin	Encoder		Type
	Incremental	Abso-lute			Incremental	Abso-lute	
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 / negated (incremental encoder)
B, B_N	Track B true / negated (incremental encoder)
N, N_N	Zero mark true / negated (incremental encoder)
CLS, CLS_N	SSI sliding pulse true / negated (absolute encoder)
DATA, DATA_N	SSI data true / 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

An incremental encoder or absolute encoder (SSI) can be connected directly (e.g. digital rotary encoders). Encoders are 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.

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} , $\overline{U_{a0}}$)
Maximum output frequency:	1.5 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 Mbit/s
Power consumption:	Not more than 300 mA

5 V encoder 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.

Note

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.

Note

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 4-9 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		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 *Catalog NC Z*, Order No.: E86060-K4490-A001-A5-7600:

Table 4-10 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

Note

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 4-11 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)	1.25 Mbps	10 m
	156 kbps	250 m

4.6 Connecting encoders

Connection of the connecting cable

Please note the following:

Note

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.

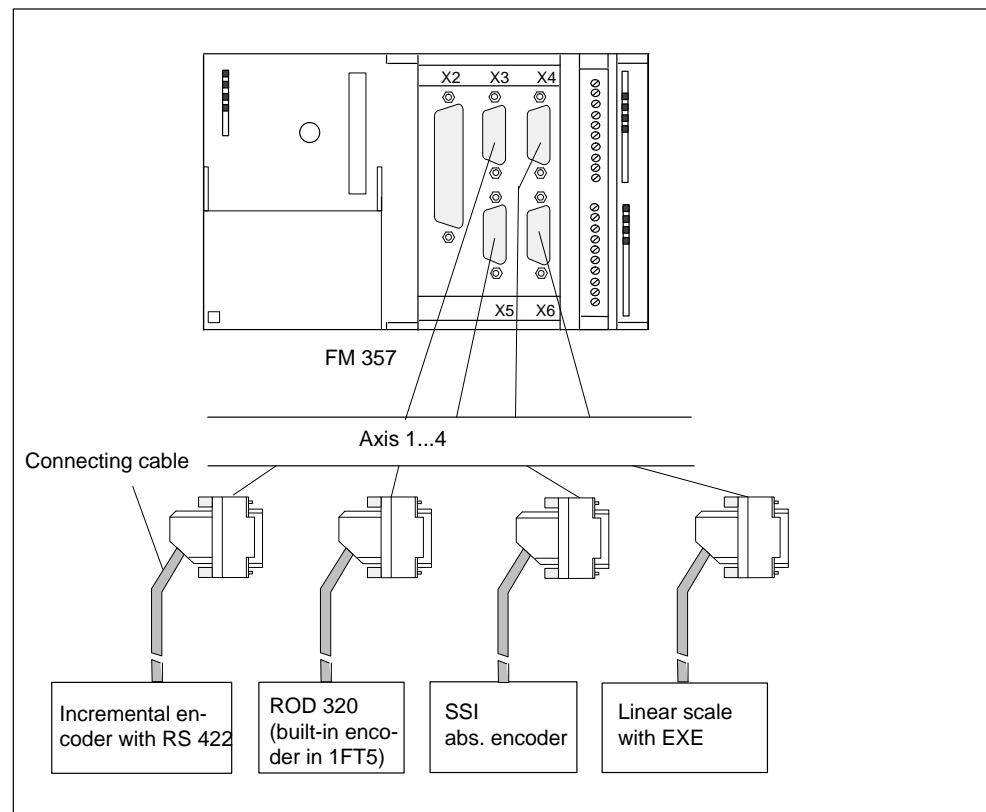


Figure 4-9 Connection of encoders

Procedure for connecting encoders

Please proceed as follows to connect the encoders:

1. Attach the connecting cable to the encoders.

For absolute encoders (SSI) it may be necessary to cut and add connectors to the cable (end of the cable to the encoder) according to the manufacturer's instructions.

2. Open the cover and insert the sub-D connector in the module.
3. Use the thumbwheel screws to lock the connector in place. Close the cover.

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) with a free cable end.

Connecting cables are available in a variety of lengths.

see *Catalog NC Z*

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

4.7 Description of I/O interface

Front connector

Probes, BEROs or other signal encoders can be connected to the 20-pin 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 4-10 shows the position of the front connector.

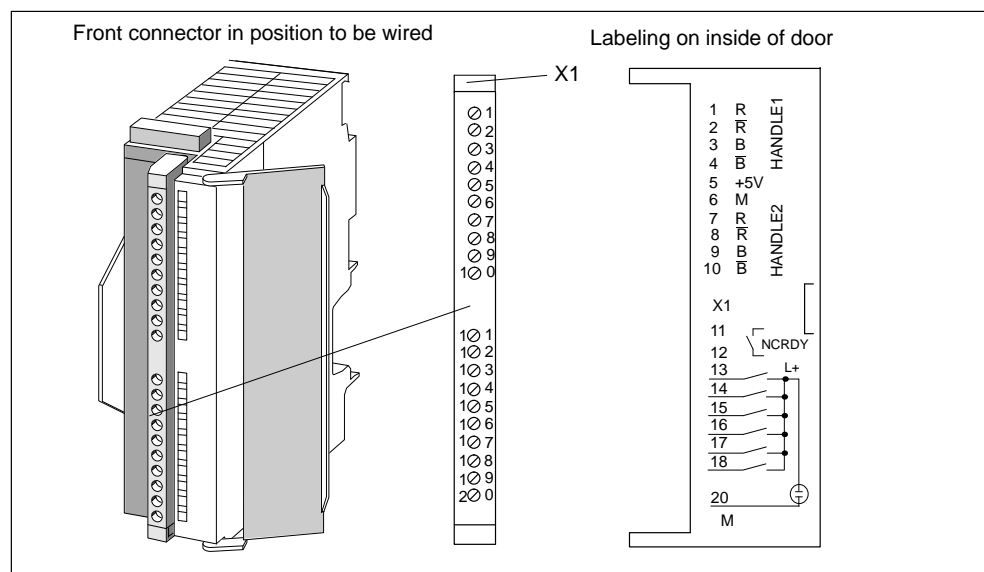


Figure 4-10 Location of X1 connector

Assignment of connector pins

Connector name: **X1**
 Connector type: 20-pin S7 front connector for single-wire terminal

Table 4-12 Pin assignment of the X1 connector

Pin	Name	Type	Pin	Name	Type
1	not assigned		11	NCRDY.1	K
2	not assigned		12	NCRDY.2	K
3	not assigned		13	I0/BERO1	DI
4	not assigned		14	I1/BERO2	DI
5	not assigned		15	I2/BERO3	DI
6	not assigned		16	I3/BERO4	DI
7	not assigned		17	I4/MEPU1	DI
8	not assigned		18	I5/MEPU2	DI
9	not assigned		19	not assigned	
10	not assigned		20	M	

Signal names

NCRDY.1...2	Ready (NC-READY contact 1...2)
BERO1...BERO4 (I0...I3)	BERO input for axes 1...4 or unassigned inputs (not for stepper motor without encoder)
MEPU1, MEPU2 (I4, I5)	Measurement pulse input 1 and 2
M	Reference potential for inputs

Signal type

DI	Digital input (24 V signal)
K	Switching contact

6 digital inputs, 2 of which for probes (I0...I5)

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 (BERO1...BERO4), **not** for stepper motor without encoder

Table 4-13 Electrical parameters of digital inputs

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

NC-READY output (NCRDY)

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

Table 4-14 Electrical parameters of relay contact NCRDY

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

4.8 Wiring of front connector

Wiring of front connector

Figure 4-11 shows you how cables are routed to the front connector and the cable strain relief provided by the shield connecting element.

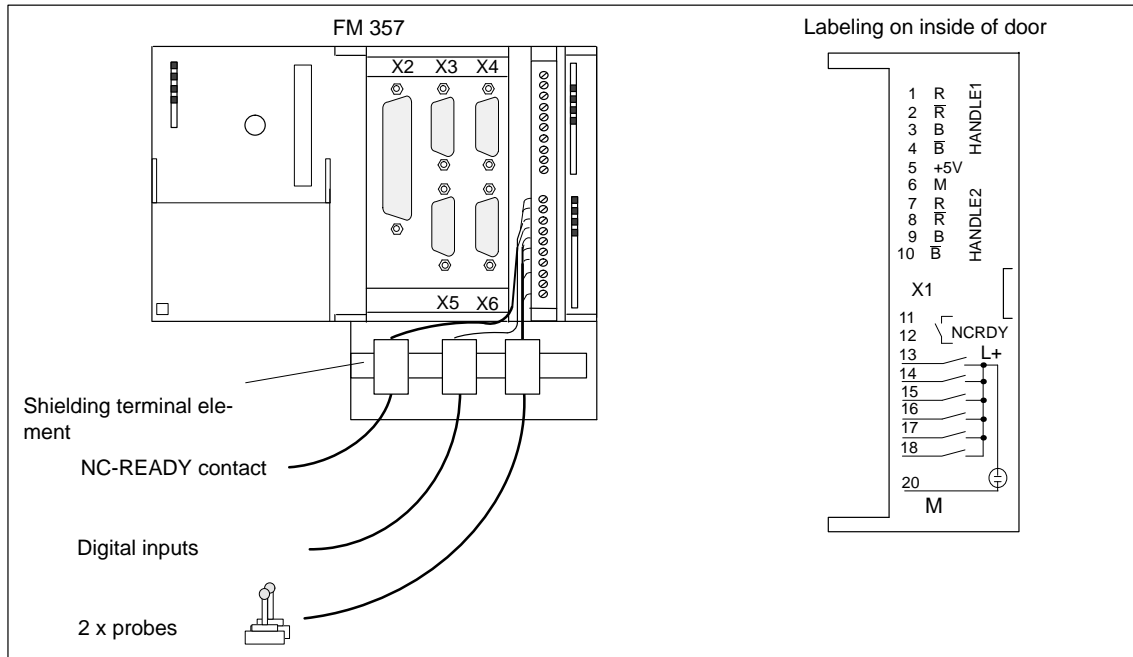


Figure 4-11 Wiring up the front connector

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.

Note

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 (press locking element at the same time).

Lock the connector in place without any electrical contact to the module.

3. Attach the cable strain relief to the connector.
4. 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 60-80 Nm.

5. Tighten the cable strain relief for the cable string.
6. Push the front connector into operating position (pressing the locking element at the same time).
7. 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 shielding terminal 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.

Shielding terminal 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 *Catalog NC Z*

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. You can use the load power supply terminals of the FM 357.
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 shielding terminal and wire up the free cable ends to the front connector.
4. Wire up the signal cable to the front connector.

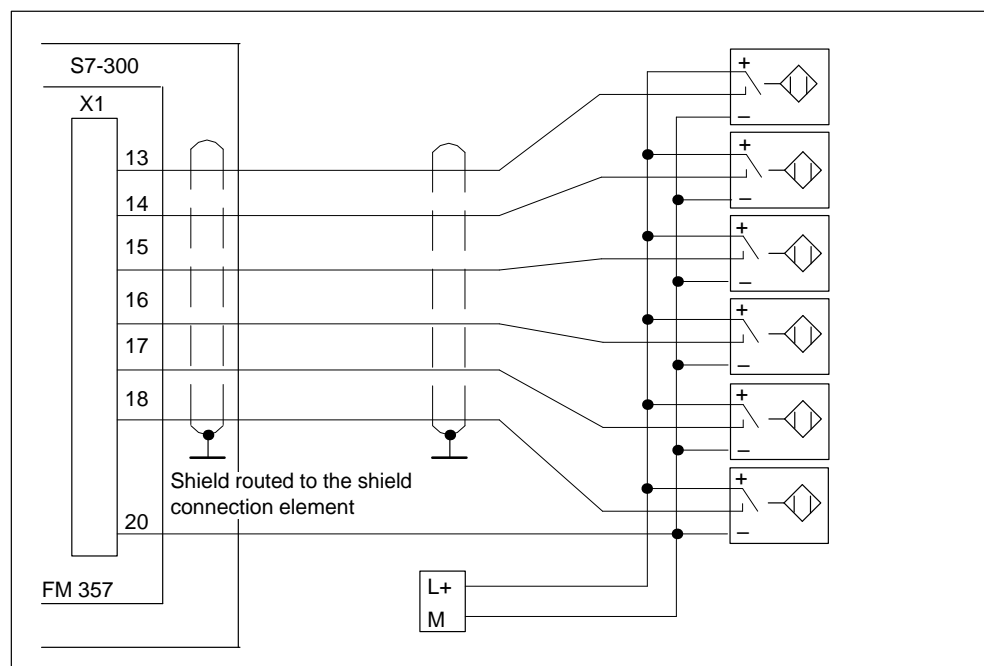


Figure 4-12 Connection overview for probes and proximity switches

Connection of NC-READY contact

When the NC-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 Installation Manual *S7-300 Programmable Controller, Hardware and Installation*.

4.9 Insertion and replacement of backup battery

General

The FM 357 is provided with a backup battery as the power supply for the battery-backed RAM.

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

Inserting the battery

Please proceed as follows:

1. Open the left-hand front door on the FM 357
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, or the lug should be to the left and the positive terminal pointing downward). Then guide the connector into the battery compartment.

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

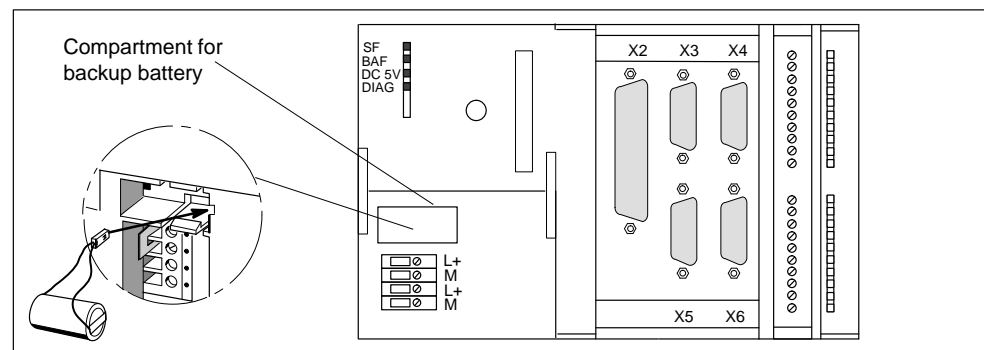


Figure 4-13 Installing the backup battery

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

Note

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” LED also indicates 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 “BAF” flashes

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 forces this status.

Note

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

Inserting a new battery

Please proceed as follows:

1. Lift up the left-hand front door.
2. Remove the battery, pulling the connector out of the socket in the battery compartment.
3. Insert the battery connector into the socket in the compartment (the notch on the connector must point to the right, or the lug should be to the left and the positive terminal pointing downward, then guide the connector into the compartment).
4. Place the battery in the compartment and close the front door (see Figure 4-13).

Battery type

Prefabricated batteries with connectors must be used.

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

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
-

Parameterization of the FM 357

General

This section provides you with an overview of how to parameterize the FM 357 using the “Parameterize FM 357” tool.

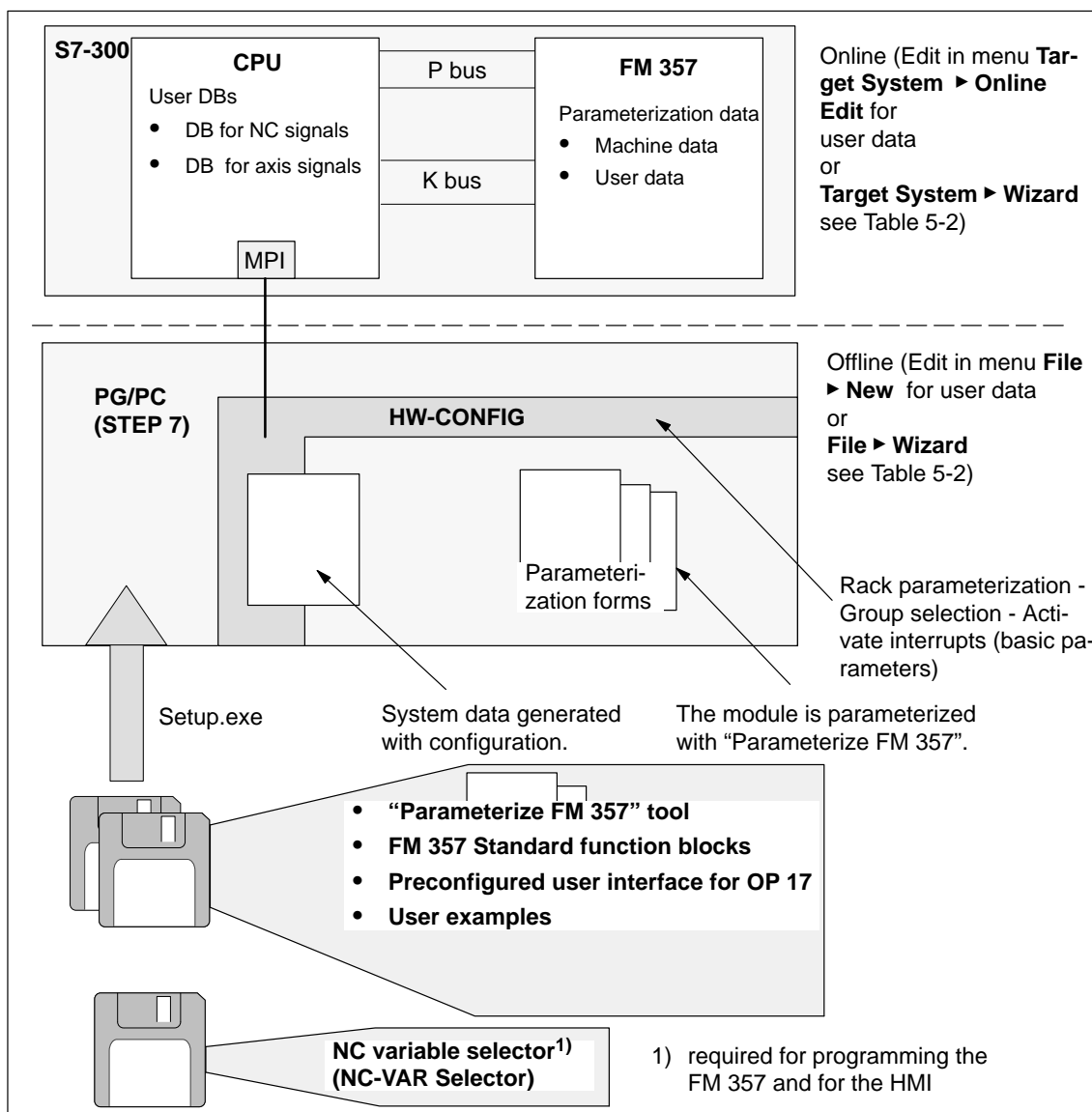


Figure 5-1 Overview of parameterization

Section overview

Section	Title	Page
5.1	Installation of "Parameterize FM 357"	5-3
5.2	Getting started with "Parameterize FM 357"	5-4
5.3	Adaptation to firmware	5-5
5.4	Parameterization data	5-7
5.5	Menus of "Parameterize FM 357"	5-23
5.6	Settings of parameterization interface	5-27

5.1 Installation of “Parameterize FM 357”

Precondition

The “Windows 95” or “Windows NT” (V4.0 or later) operating system and the appropriate STEP 7 program (V3.1 or later) must be installed on the programming device (PG/PC).

For online operation, the link between the PG and the S7-300 CPU must already be set up (see Figures 4-1 and 4-2).

Installation

The entire software (parameterization tool, function blocks, preconfigured interface for OPs) is stored on 3.5 inch diskettes and is installed completely.

Install the software as follows:

1. Insert diskette 1 into the diskette drive of your PG/PC.
2. Start the dialog for installing the software in Windows 95 by double clicking on the symbol “Add/Remove Programs” in “Control Panel”.
3. Select the diskette drive and file **Setup.exe** and start the installation process.
4. Follow the instructions displayed by the installation routine step for step.

Result: The software is installed as standard in the following directories:

- “Parameterize FM 357” tool: **[STEP7 directory]\S7FM357**
- Function blocks: **[STEP7 directory]\S7LIBS\FM357_LI**
- User interface for OP 17: **[STEP7 directory]\EXAMPLES\S7OP_BSP**
- User examples: **[STEP7 directory]\EXAMPLES\FM357_EX**

Note

If you already have an earlier version than 2.0 of “Parameterize FM 357” installed, you must **unload** it. It is absolutely essential that any earlier version is removed before you install the new version.

Please proceed as follows:

1. Start the dialog for removing the software in Windows 95 by double clicking on the symbol “Add/Remove Programs” in “Control Panel”.
 2. Select the entry FM 357 in the list of installed software packages and click on button “Add/Remove”.
 3. In version 1.1/04, you may need to delete write-protected files in directory **...\S7FM357*.***.
-

5.2 Getting started with “Parameterize FM 357”

Precondition

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

Configuring

You can only start to configure if you have already created a project in which you can store the parameter settings. You will find further information on how to configure modules in your user manual *Standard Software for S7 and M7, STEP 7*. The description below outlines only the most important steps.

1. Start the *SIMATIC Manager* and set up a new project.
2. Select menu **Insert ► Station** to insert a **SIMATIC 300 station**.
3. Select the **SIMATIC 300 station**. Call up the S7 hardware configuration from the menu **Edit ► Open Object**.
4. Select a module subrack.
5. Select the CPU and the FM 357 multi-axis module with the corresponding order numbers from the module catalog and insert them in the hardware table to match your configuration.

A communication link to the CPU must be set up to parameterize the FM 357 online.

6. Select the module to be parameterized with a double click.

The **Properties** window appears.

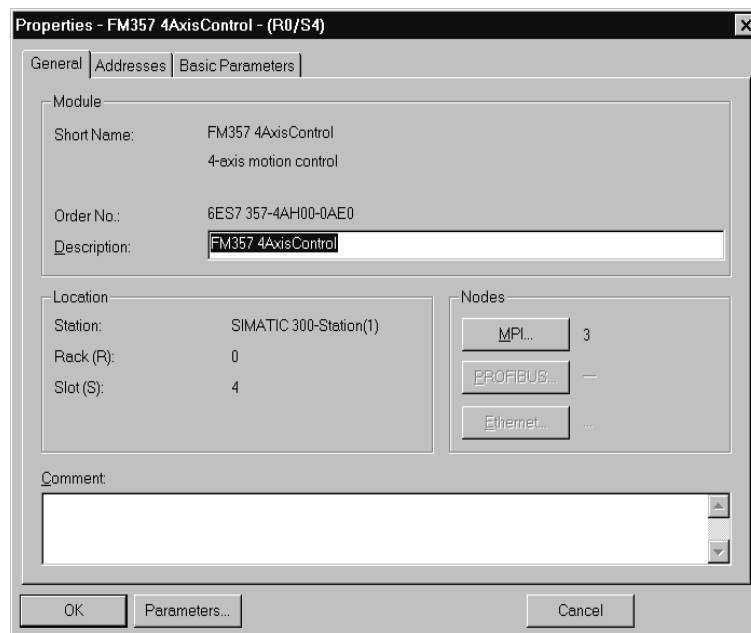


Figure 5-2 Getting started with "Parameterize FM 357"

7. In this display, you can use the index cards (general, addresses and basic parameters) of the FM 357 to
 - Name the FM 357
 - Change the address of the FM 357

Click the **Parameters** button to call up the screen for setting the parameters.

You can now set the parameters of your module. Section 5.4 provides an overview of the data which can be parameterized.

When you have configured your project, you can call up the **Properties** screen in S7 Configuration by selecting the module and activating the menu command **Edit ▶ Object Properties**.

5.3 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 machine data of any firmware version in offline mode and process them in the Parameterization Wizard.

You can set the firmware version in menu **Options ▶ Firmware Version**

Note

If you are using an earlier version than 2.0 of "Parameterize FM 357", then the following functions will not be available:

- Travel to fixed stop
 - Gantry grouping
 - Master-value link/curve tables
 - EMERGENCY STOP
-

Update procedure

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

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

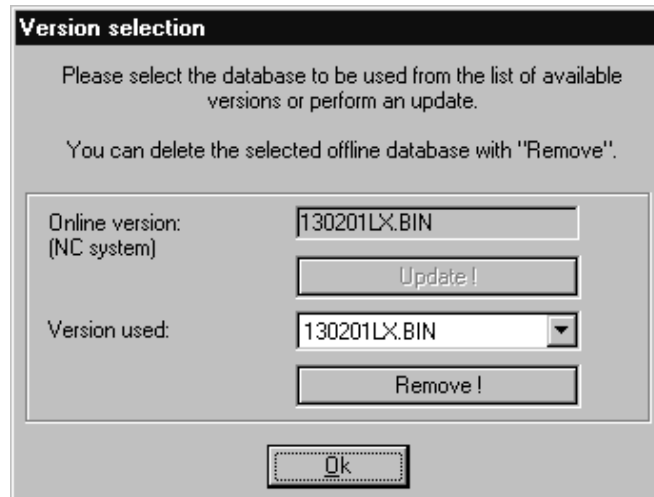


Figure 5-3 Adaptation to firmware

To start the update process, click on the “Update” button.

On completion of the update, another offline database is available.

You can delete a selected offline database by clicking on the “Remove” button.

5.4 Parameterization data

What can be parameterized?

It is possible to parameterize the following data areas:

- Machine data (parameters)
- User data
 - R parameters
 - Zero offset
 - Tool offset values
 - NC programs

Parameterization data can be processed and stored in both online and offline (PG/PC) modes.

Online editing

For online operation, the link between the PG and the S7-300 CPU must already be set up (see Figures 4-1 and 4-2).

You can edit machine data (parameters) in menu **Target System ▶ Wizard**.

You can create and edit user data in menu **Target System ▶ Online Edit** (NC programs, tool offset data).

The following data are stored in the working memory of the FM 357:

- Machine data
- R parameters
- Zero offset
- Tool offset values

The following data are stored in the program memory of the FM 357:

- NC programs
 - Main programs (*.mpf)
 - Subroutines (*.spf)

When you select menu **Target System ► Online Edit** the following selection dialog is displayed:

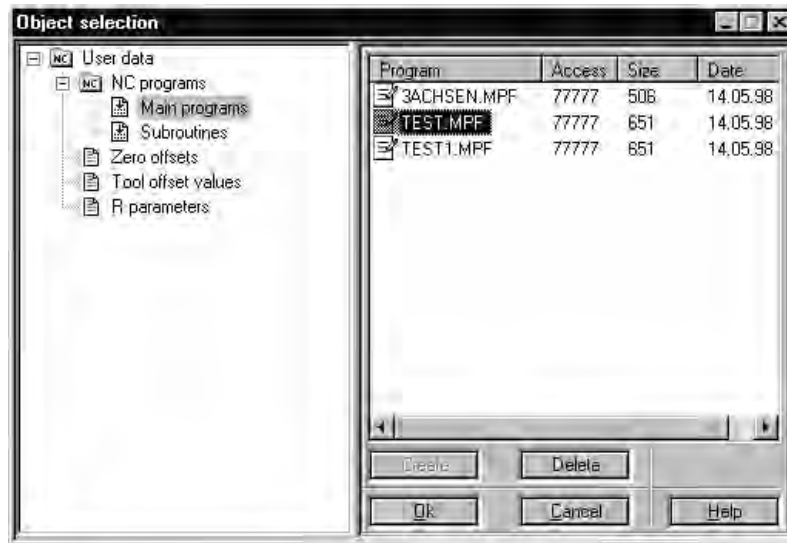


Figure 5-4 Online Edit selection dialog

Offline editing

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

- Machine data (parameters)
in menu **File ► Wizard**
- User data
in menu **File ► New**

The following selection dialog will appear on your screen:



Figure 5-5 Offline editing selection dialog


The data are stored on the hard disk of the PG/PC in menu **File ▶ Save As...as** follows:

- Machine data (*.pda)
- User data
 - R parameters (*.rpa)
 - Zero offsets (*.uif)
 - Tool offset values (*.wzk)
 - NC programs
 - Main programs (*.mpf)
 - Subroutines (*.spf)

You can edit existing files in menu **File ▶ 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.

5.4.1 Machine data (parameters)

General

The purpose of machine data is to adapt the FM 357 to the specific application of the user. Parameterization with machine data is essential in order for the functions of the FM 357 to be activated.

Parameter definition

There are two methods by which the FM 357 can be parameterized:

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

To switch between the Parameterization wizard and list-based parameterization, select **Options ▶ Settings**.

Parameterization 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.

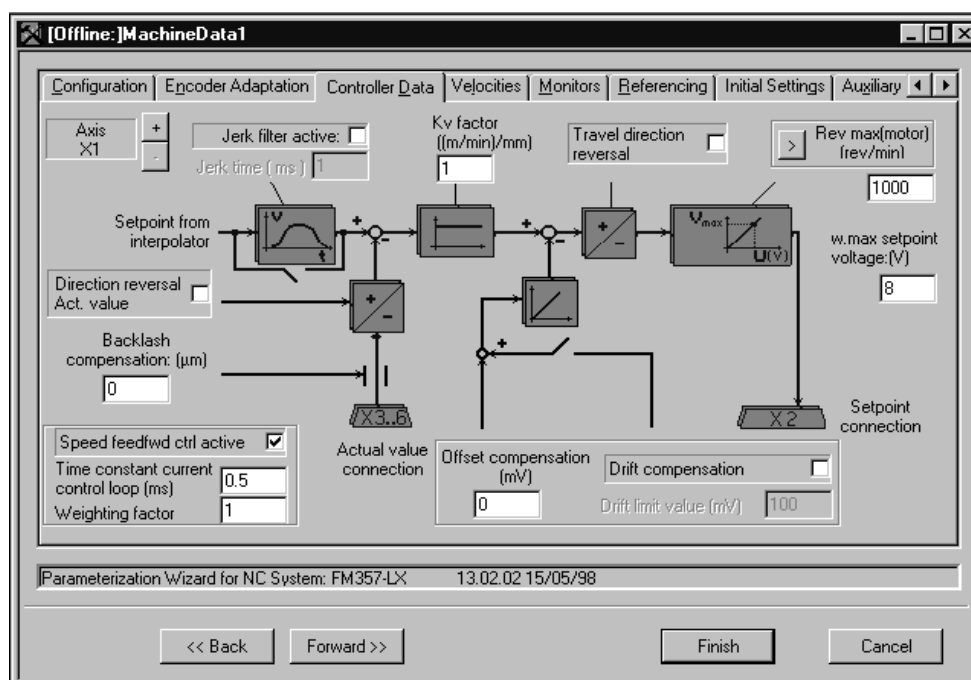


Figure 5-6 Machine data, e.g. controller data

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

Note

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

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.

Machine data list

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

Table 5-1 Machine data (parameters)

Parameters	Default values	Value range/meaning	Unit	See Section
Configuration				
Internal system of measurement	metric	Metric = 10^{-3} Inches = 10^{-4}	[mm] [inches]	9.1
Max. cycle time of user program	40	10 to 200	[ms]	9.1
Override coding	Gray	Gray (default setting) Binary		9.1
Memory configuration				
Number of R parameters	100	0 to 10 000	–	10.17
Number of curve tables	0	0 to 20	–	9.13.3
Number of curve segments	0	0 to 80	–	9.13.3
Number of curve table polynomials	0	0 to 160	–	9.13.3
Axis configuration				
Axis name	X1, Y1, Z1, A1 X, Y, Z A	Machine axis Geometry axis Special axis 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
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
Drive	Simulation	Simulation Servo drive Stepper motor (SM) without encoder Stepper motor (SM) with encoder	–	9.1
External master value	No	No Yes		9.1 9.13.3
VDI output (for simulation)	No	No Yes		9.1

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section	
Encoder adjustment					
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	1 to 10 000	–	9.2
		No. of spindle revolutions	1 to 10 000		
Resolver gearbox	1/1	No. of motor revolutions	1 to 10 000	–	9.2
		No. of encoder revolutions	1 to 10 000		
Increments per encoder revolution	2048	2 to 16 384	–	9.2.1	
Indexing period	0.01	0.001 to 100	[mm]	9.2.1	
Linear measurement system is negative	–	No: Absolute value changes to plus with positive axis movement (positive) Yes: Absolute value changes to minus with positive axis movement (negative)	–	9.2.1	
Baud rate	250	250 kHz 400 kHz 500 kHz 1 MHz	[kHz]	9.2.2	
			[MHz]		
Coding	x	Output code of encoder: Gray code Binary code	–	9.2.2	
Parity test	Yes	Yes No	–	–	
Parity	x	Odd Even	–	9.2.2	
Measurement	x	Not provided Provided	–	9.2.2	
Sensing probe connection	x	Input 4 Input 5	-	9.2.2	
Message frame length	x	25-bit multi-turn 13-bit single-turn 21-bit multi-turn	–	9.2.2	

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section
Increments per encoder revolution	–	8192 only with 25-bit multi-turn and 13-bit single-turn 4096 2048 ... 2 ¹	–	9.2.2
Increments 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	[μm], [10 ^{–3} degr.]	9.3
Position control gain (K _v factor)	1	0.1 to 100	[(10 ³ mm/min)/mm], [(10 ³ degr./min)/degr.]	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 drive 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	100	–3 000 to +3 000	[mV]	9.3
Speed feedforward control active	Yes	No Yes	–	9.3
Time constant for current control loop	0.5	0 to 10	ms	9.3
Weighting factor	1	0 to 10	–	9.3

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section
Velocities				
Positioning velocity	10 000	0 to 999 999	[mm/min], [rev/min]	9.4
Axis velocity	2 000	0 to 999 999	[mm/min], [rev/min]	9.4
Rapid traverse override	10 000	0 to 999 999	[mm/min], [rev/min]	9.4
Acceleration pattern Initial setting	Brisk acce- leration	Brisk acceleration Soft acceleration Drive acceleration √ Brisk acceleration	–	9.4
Acceleration	1	0 to 10 000	[m/s ²], [rev/s ²]	9.4
Jerk	1 000	0 to 100 000	[m/s ³], [rev/s ³]	9.4
Creep velocity	10 000	0 to 999 999	[mm/min], [rev/min]	9.4
Creep acceleration	1	0 to 10 000	[m/s ²], [rev/s ²]	9.4
Path acceleration	10	0 to 1 000	[m/s ²]	9.4
Path jerk	100	0 to 100 000	[m/s ³]	9.4
Braking time EMER- GENCY STOP	0.05	0.02 to 1 000	[s]	9.16
Cutout delay controller enable EMERGENCY STOP	0.1	0.02 to 1 000	[s]	9.16
Monitoring				
Monitoring time (mo- ving into position)	1	0 to 100	[s]	9.5.1
Target range coarse	0.04	0 to 1 000	[mm], [degrees]	9.5.1
Target range fine	0.01	0 to 1 000	[mm], [degrees]	9.5.1
Following error monitoring system (movement of axis)	1	0 to 1 000	[mm], [degrees]	9.5.1
Zero speed control delay	0.4	0 to 100	[s]	9.5.1
Zero speed range	0.2	0 to 1 000	[mm], [degrees]	9.5.1
Threshold velocity for stationary axis	5	0 to 10 000	[mm/min], [rev/min]	9.5.1

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section
Clamping tolerance	0.5	0 to 1 000	[mm], [degrees]	9.5.1
Set speed	100	0 to 200	[%]	9.5.1
Monitoring time (set speed)	0	0 to 100	[s]	9.5.1
Actual speed	11 500	0 to 9 999 999	[mm/min], [rev/min]	9.5.1
Encoder limit frequency	300 000	0 to 1 500 000	[Hz]	9.5.2
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	–	9.5.2
Number of increments (rotation monitoring)	2 000	10 to 1 000 000	–	9.5.2
Increment tolerance (rotation monitoring)	50	10 to number of increments	–	9.5.2
1st software limit switch plus	10 ⁸	–100 000 000 to +100 000 000	[mm], [degrees]	9.5.3
1st software limit switch minus	–10 ⁸	–100 000 000 to +100 000 000	[mm], [degrees]	9.5.3
2nd software limit switch plus	10 ⁸	–100 000 000 to +100 000 000	[mm], [degrees]	9.5.3
2nd software limit switch minus	–10 ⁸	–100 000 000 to +100 000 000	[mm], [degrees]	9.5.3
Referencing				
NC start without reference point approach	No	No Yes	–	9.6
Reference point approach necessary	Yes	Yes No	–	9.6
Axis with reference point switch	Yes	Yes No	–	9.6.1
Reference point approach direction	Plus	Plus Minus	–	9.6.1
Zero marker/BERO	In front of RPS	In front of reference point switch (RPS) Behind/on RPS	–	9.6.1
Reference point coordinate	0	–100 000 to +100 000	[mm], [degrees]	9.6.1
Reference point offset	–2	–100 000 to +100 000	[mm], [degrees]	9.6.1
Maximum distance to RPS	10 000	0 to 100 000	[mm], [degrees]	9.6.1

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section
Max. distance to zero marker/BERO	20	0 to 10 000	[mm], [degrees]	9.6.1
Referencing speed	5 000	0 to 999 999	[mm/min], [rev/min]	9.6.1
Creep velocity	300	0 to 999 999	[mm/min], [rev/min]	9.6.1
Approach velocity	1 000	0 to 999 999	[mm/min], [rev/min]	9.6.1
BERO edge evaluation	1	1-edge evaluation 2-edge evaluation	–	9.6.2
Traversing direction key	Minus	Minus direction Plus direction	–	9.6.3
Encoder alignment status	Not aligned	Not aligned Enabled Aligned	–	9.6.3
Actual value (alignment value)	0	–100 000 to +100 000	[mm], [degrees]	9.6.3
Initial setting				
Initial setting on program start				
Movement	x	Linear interpolation with rapid traverse G0	–	10.5.3
		Linear interpolation with feed G1		10.5.4
Exact stop Continuous-path mode	x	Exact stop G60	–	10.7.1
		Continuous-path mode G64		10.7.2
		Continuous-path mode with programmed rounding clearance G641		
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	–	10.7.3
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

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section
Speed feedforward control	x	Feedforward control off FFWOF Feedforward control on FFWON	–	10.25
Tool number	0	0 to 29	–	10.16
Behaviour after program end and NC 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
Auxiliary functions				
Output options for M functions	x	Output before movement Output during movement Output after movement	–	9.7
Output options for H functions	x	Output before movement Output during movement Output after movement	–	9.7
Digital I/Os				
Slots used	None	None Slot 1 Slot 1 + 1	–	9.8
Module size	1 byte	1 byte 2 bytes	–	9.8
Byte 1	–	Inputs Outputs	–	9.8
Byte 2	–	Inputs Outputs	–	9.8
9 to 16 17 to 24	9 to 16	9 to 16; 17 to 24 Assignment of bit numbers	–	9.8
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) ...	–	9.9.1
Cam position Minus cam	0	–100 000 000 to +100 000 000	[mm], [degrees]	9.9.1
Cam position Plus cam	0	–100 000 000 to +100 000 000	[mm], [degrees]	9.9.1
Lead time/delay time minus cam	0	–100 to +100	[s]	9.9.1

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section
Lead time/delay time plus cam	0	-100 to +100	[s]	9.9.1
Signal level for minus cam	0 → 1	0 → 1 1 → 0 (inverted)	-	9.9.1
Signal level for plus cam	0 → 1	0 → 1 1 → 0 (inverted)	-	9.9.1
Assignment to digital outputs, minus cam	None	No assignment Digital outputs 9 to 16 Digital outputs 17 to 24	-	9.9.1
Assignment to digital outputs, plus cam	None	No assignment Digital outputs 9 to 16 Digital outputs 17 to 24	-	9.9.1
Motion coupling				
Overlaid motion with synchronized actions				
Calculation of compensation value	Absolute	Absolute Integral	-	9.13.4
Upper limit of compensation value	10 ⁸	0 to 100 000 000	[mm], [degrees]	9.13.4
Velocity of compensation value	10 ³	0 to axis velocity	[mm/min], [rev/min]	9.13.4
Master value link				
Type of master value link	Setpoint value	Actual value Setpoint Simulated master value	-	9.13.3
Threshold value for coarse synchronism	1	0 to 10 000	[mm], [degrees]	9.13.3
Threshold value for fine synchronism	0.5	0 to 10 000	[mm], [degrees]	9.13.3
Parameterize curve tables				
Offset to master axis position	0	-100 000 000 to +100 000 000	[mm], [degrees]	9.13.3
Scaling of master axis position	1	-1 000 000 to +1 000 000	-	9.13.3
Offset to slave axis position	0	-100 000 000 to +100 000 000	[mm], [degrees]	9.13.3
Scaling of slave axis position	1	-1 000 000 to +1 000 000	-	9.13.3
Behaviour after program end and NC Reset				
Active coupled axis groupings remain valid	No	No Yes	-	9.13.1
Master value link remains active	No	No Yes	-	9.13.3

Table 5-1 Machine data (parameters), continued

Parameters	Default values	Value range/meaning	Unit	See Section
Gantry grouping				
Master axis	–	Machine axis name of master axis	–	9.13.2
Synchronized axis	–	Machine axis name of synchronized axis	–	9.13.2
Separate gantry grouping	No	No Yes	–	9.13.2
Limit value for warning	0	0 to 100	[mm], [degrees]	9.13.2
Trip limit	0	0 to 100	[mm], [degrees]	9.13.2
Trip limit for referencing	0	0 to 100	[mm], [degrees]	9.13.2
Fixed stop				
Permit travel to fixed stop	No	No Yes	–	9.15
Fixed stop detection	Following error	Following error Sensor Following error or sensor	–	9.15
Following error for fixed stop detection	2	0 to 1 000	[mm], [degrees]	9.15
Monitoring window	1	0 to 1 000	[mm], [degrees]	9.15
Clamping torque	5	0 to 100	%	9.15
Torque limit on approach to fixed stop	5	0 to 100	%	9.15
Error messages: • Axis has not reached the fixed stop • Approach to fixed stop aborted	Yes	Yes No	–	9.15

List-based parameterization

The user is provided with a list of all the machine data of the FM 357.

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

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

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

The following screen shot shows you the list parameterization window.

Number	Name	Value	Unit	Activation	Comment
32040.A1	JOG_REV_VELO_RAPID	2.5000	mm/U	NC Reset	
32040.A2	JOG_REV_VELO_RAPID	2.5000	mm/U	NC Reset	
32040.A3	JOG_REV_VELO_RAPID	2.5000	mm/U	NC Reset	
32040.A4	JOG_REV_VELO_RAPID	2.5000	mm/U	NC Reset	
32050.A1	JOG_REV_VELO	0.5000	mm/U	NC Reset	
32050.A2	JOG_REV_VELO	0.5000	mm/U	NC Reset	
32050.A3	JOG_REV_VELO	0.5000	mm/U	NC Reset	
32050.A4	JOG_REV_VELO	0.5000	mm/U	NC Reset	
32060.A1	POS_AX_VELO	10000.0000	mm/min	NC Reset	
32060.A2	POS_AX_VELO	10000.0000	mm/min	NC Reset	
32060.A3	POS_AX_VELO	10000.0000	mm/min	NC Reset	
32060.A4	POS_AX_VELO	10000.0000	mm/min	NC Reset	
32070.A1	CORR_VELO	50.0000	%	NC Reset	
32070.A2	CORR_VELO	50.0000	%	NC Reset	
32070.A3	CORR_VELO	50.0000	%	NC Reset	
32070.A4	CORR_VELO	50.0000	%	NC Reset	
32074.A1	FRAME_OR_CORRPOS_NOTALLOWED	0	[hex]	Power On	
32074.A2	FRAME_OR_CORRPOS_NOTALLOWED	0	[hex]	Power On	
32074.A3	FRAME_OR_CORRPOS_NOTALLOWED	0	[hex]	Power On	
32074.A4	FRAME_OR_CORRPOS_NOTALLOWED	0	[hex]	Power On	
32080.A1	HANDWH_MAX_INCR_SIZE	0.0	mm	NC Reset	

Figure 5-7 Machine data

Note

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.

5.4.2 User data

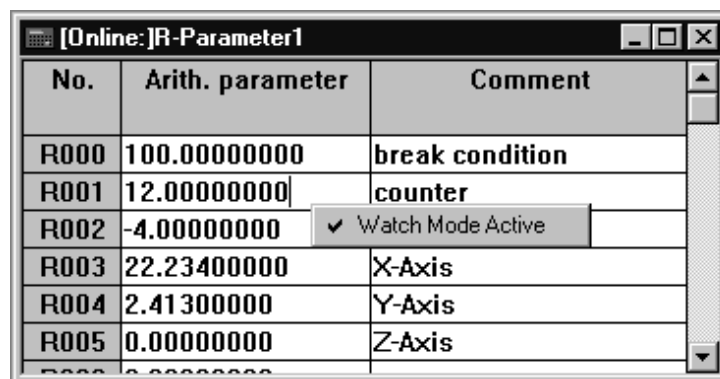
General

The following data can be parameterized specifically by the user:

- R parameters

Values are input in the menu for R parameters.

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 key.

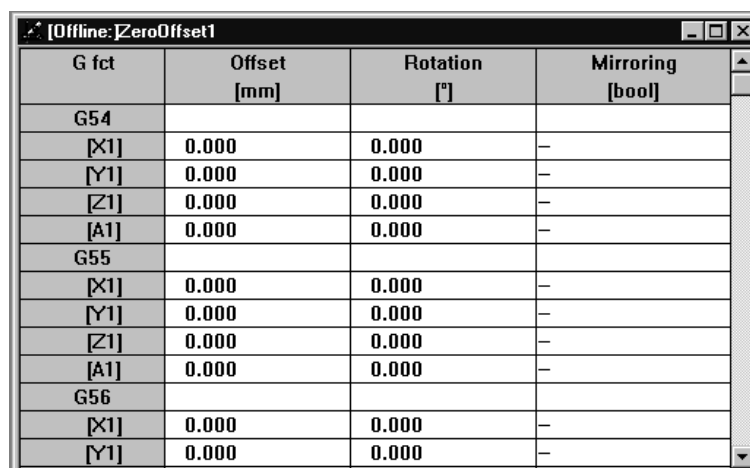


No.	Arith. parameter	Comment
R000	100.00000000	break condition
R001	12.00000000	counter
R002	-4.00000000	<input checked="" type="checkbox"/> Watch Mode Active
R003	22.23400000	X-Axis
R004	2.41300000	Y-Axis
R005	0.00000000	Z-Axis

Figure 5-8 Entering values for R parameters

- Zero offset

Values are input in the menu for zero offsets.



G fct	Offset [mm]	Rotation [°]	Mirroring [bool]
G54			
[X1]	0.000	0.000	-
[Y1]	0.000	0.000	-
[Z1]	0.000	0.000	-
[A1]	0.000	0.000	-
G55			
[X1]	0.000	0.000	-
[Y1]	0.000	0.000	-
[Z1]	0.000	0.000	-
[A1]	0.000	0.000	-
G56			
[X1]	0.000	0.000	-
[Y1]	0.000	0.000	-

Figure 5-9 Entering values for zero offsets

- Tool offset values
Values are input in the menu for tool offsets.

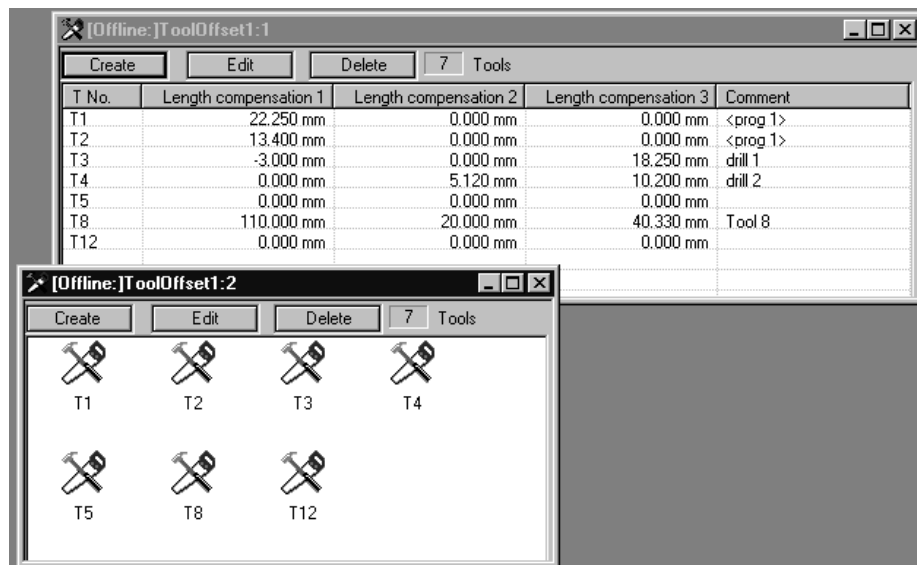


Figure 5-10 Entering values for tool offsets

- NC programs
Values are input in the menu for NC programs.

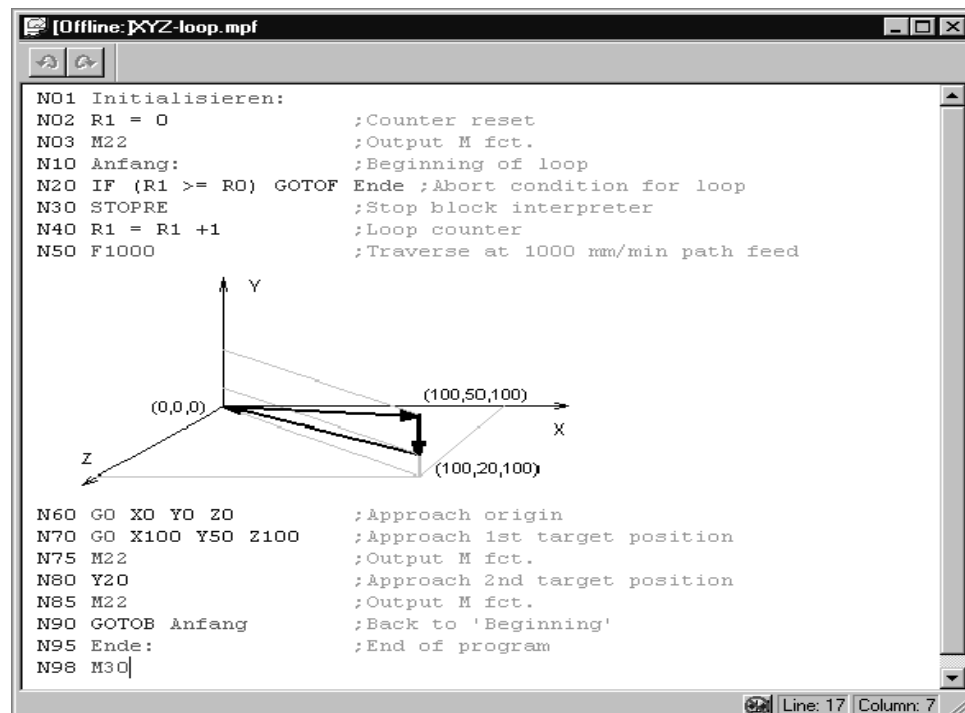


Figure 5-11 Input of values for NC programs

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

5.5 Menus of “Parameterize FM 357”

Menus

The following table provides an overview of all “Parameterize FM 357” menus.

Table 5-2 Menus of “Parameterize FM 357”

Menu title or entry (with single command)	Shortcut	Significance
File	Alt + F	Create, open, save, print and generate files. Actions such as New..., Open..., refer to offline data
<u>N</u> ew >	Ctrl + N	Creates a new <main object> (offline object)
<u>O</u> pen...	Ctrl + O	Opens a < main object > that has already been saved
<u>W</u> izard	–	Starts the offline Parameterization Wizard
<u>C</u> lose	Ctrl + F4	Closes all windows contained in the < main object > (offline and online objects)
<u>S</u> ave	Ctrl + S	Saves the active < main object > (offline)
Save As ASC <u>I</u> I File	Ctrl + S	Saves the NC program in the old format (ASCII)
Save <u>A</u> s...	–	Saves the active < main object > with a new name on the permanent data storage medium (offline)
<u>P</u> rint...	Ctrl + P	Prints the active < main object > or part of it
Print <u>P</u> review	–	Shows the object exactly as it will be printed – you cannot edit the object
Print <u>S</u> etup...	–	Sets up the printer and sets print options
<u>1</u> < Name of last file opened>	–	Opens the last file to have been opened
<u>2</u> <Name of next to last file opened>	–	Opens the next to last file to have been opened
<u>3</u> <Name of third from last file opened>	–	Opens the third from last file to have been opened
<u>4</u> <Name of fourth from last file opened>	–	Opens the fourth from last file to have been opened
<u>E</u> xit	Alt + F4	Closes all windows of the application and exits the application
Edit	Alt + E	Undo the last action, cut, copy, paste or deletion of selected objects
<u>U</u> ndo	Ctrl + Z	Undoes the last action
<u>R</u> edo	–	Redoes the last action that was undone (reverses Undo)
<u>C</u> ut	Ctrl + X	Deletes the selected objects and inserts them in the Clipboard
<u>C</u> opy	Ctrl + C	Copies the selected objects and inserts them in the Clipboard
<u>P</u> aste	Ctrl + V	Inserts the clipboard contents at the cursor position

Table 5-2 Menus of "Parameterize FM 357", continued

Menu title or entry (with single command)	Shortcut	Significance
<u>D</u> elete	–	Deletes the selected data
S <u>e</u> lect All	Ctrl + A	Selects the entire document
<u>S</u> earch...	Ctrl + F	Searches text
R <u>e</u> place...	–	Replaces one particular text with another
<u>F</u> ont...	–	Opens a dialog for the selection of a new font
I <u>n</u> sert New Object...	–	Inserts a new OLE object
L <u>i</u> nk <u>s</u>	–	Opens a linked OLE object
O <u>bj</u> ect Properties	–	Displays the properties of the OLE object
<O <u>bj</u> ect>	–	Context menu of OLE object
T arget System	Alt + Z	Loads and controls blocks and programs, controls and monitors modules. All actions refer to online objects
✓ <u>C</u> ommunication	–	Establishes or breaks the online connection to the target system
<u>W</u> izard	–	Starts the Parameterization Wizard
<u>O</u> nline Edit...	–	Opens an online data object
<u>T</u> ransfer/Activate Data	–	Activates online data or transfers data created in offline mode to the FM
FM R <u>e</u> start >	–	Module power-up
Normal Powerup	–	FM power-up, modified machine data can be activated
Powerup with Defaults	–	FM power-up with default values, the entire program memory is erased
FM P <u>r</u> operties...	–	Displays properties of the module (e.g.: SW versions, system settings, memory utilization, user DBs)
T est	Alt + T	<Controls or monitors> the program running on the module
✓ S <u>t</u> art <u>p</u>	–	Opens the startup window
✓ T <u>r</u> oubleshooting	–	Opens the troubleshooting window Displays faults in the module
✓ <u>S</u> ervice	–	Opens the window to look at servicing data
✓ <u>T</u> race	–	Opens the trace window

Table 5-2 Menus of "Parameterize FM 357", continued

Menu title or entry (with single command)	Shortcut	Significance
View	Alt + V	Selection of various view sizes, zoom factors, views and representations
Axis <u>S</u> tatus >	–	Opens or closes the axis status window
✓ Active	–	
Display of >	–	} Switches the data display in the axis status window
✓ Actual Values	–	
Distances to go	–	
Setpoints	–	
Coordinate System >	–	} Switches between the machine coordinate system/ workpiece coordinate system
✓ MCS	–	
WCS	–	
<u>A</u> xis Selection >	–	Selects another axis as the active axis
✓ <u>1</u> st Axis	–	Selects the 1st axis as the active axis
<u>2</u> nd Axis	–	Selects the 2nd axis as the active axis
<u>3</u> rd Axis	–	Selects the 3rd axis as the active axis
<u>4</u> th Axis	–	Selects the 4th axis as the active axis
Contents <u>5</u> th Column >	–	Determines what is displayed in the 5th column
✓ Default Values	–	Displays default values
Upper Limit Values	–	Displays the upper limit value
Lower Limit Values	–	Displays the lower limit value
Both Limit Values	–	Displays upper and lower limits
Activation Criterion	–	Displays the time of activation for each machine data
Data Type	–	Displays the data type
✓ Watch Mode Active	–	Activates/deactivates watch mode for online R parameters (R parameters can be updated cyclically).
✓ Parameter List	–	Switches between the overview display of all tools for the tool offsets and display of individual values (parameters) for the tool offsets.
✓ <u>T</u> ool Bar	–	Displays the symbol bar (on/off)
✓ <u>S</u> tatus <u>B</u> ar	–	Displays the status line (on/off)
Options	Alt + O	Generation of the application
Customize...	–	Changes various individual settings of this application (e.g. switches between Parameterization Wizard and list parameterization)
<u>F</u> irmware Version	–	If the user has several different FMs, he can select the appropriate firmware here

Table 5-2 Menus of "Parameterize FM 357", continued

Menu title or entry (with single command)	Shortcut	Significance
Window	Alt + W	Arranges all application windows. Changes to a specified window
A <u>rrange</u> >	–	Arranges all windows
C <u>ascade</u>	–	Stacks all windows, one behind the other
H <u>orizontally</u>	–	Spaces all windows evenly top to bottom
V <u>ertically</u>	–	Spaces all windows evenly left to right
A <u>rrange Icons</u> ...	–	Rearranges all windows which are displayed as icons
N <u>ew Window</u>	–	Displays the data of the active object in a separate view. Exception: It is not possible to open a 2nd Parameterization Wizard. However, one or several table views may be opened for one Wizard
C <u>lose All</u>	–	Closes all the windows open in the application except for the overview window
1 < opened window >	–	Changes to window <window name>
n < opened window n >	–	Changes to window <window name>
?	Alt + ?	Searches and displays help information
C <u>ontents</u> ...	F1	Offers a variety of ways to access help information
A <u>bout</u> ...	–	Displays information on the current version of the application (About Box)

5.6 Settings of parameterization interface

Changing settings

You can make specific program settings for the “Parameterize FM 357” tool in the **Change settings** dialog.

You can activate this dialog by selecting menu command **Options ▶ Settings**.

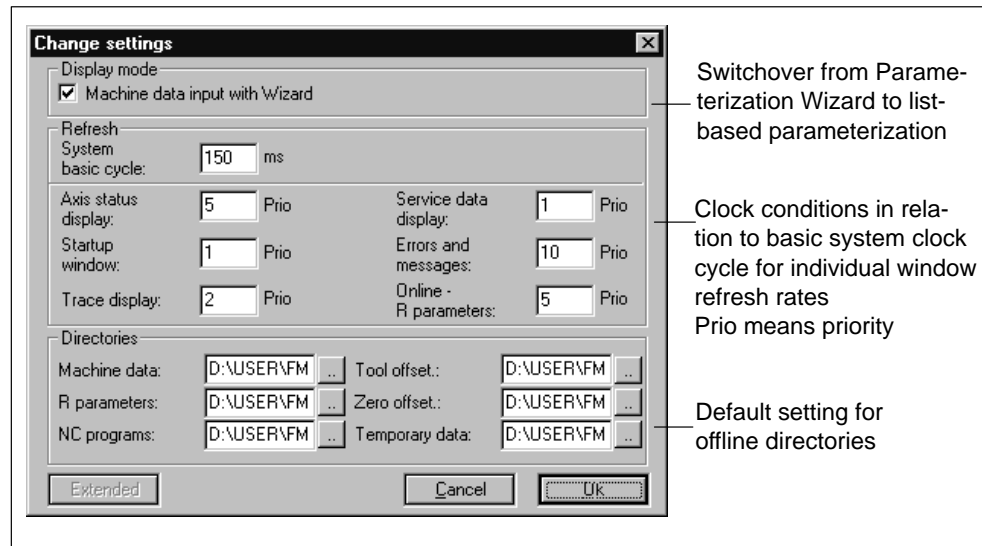


Figure 5-12 Settings of parameterization interface

The system clock cycle indicates the ms cycle in which system data (e.g. in startup window or axis configuration) are read from the FM 357.

The priority specifies the ratio to the basic cycle.

Example

Errors and messages: 10 prio, system clock cycle: 100 ms

The data in this window are read every $10 * 100$ ms.

If data are displayed too slowly (e.g. too much computer capacity is taken up with the data transfer), you must increase the system cycle time.



Programming the FM 357

General

The purpose of this function description of the modules listed in Table 6-1 is to explain the process of communication between the CPU and FM 357 in the SIMATIC S7-300 programmable controller. The blocks, which you parameterize, and the user data blocks (see Section 6.9) enable you to develop the user program for your application.

Note

This description is valid for one to three FM 357 modules.

Preconditions

You must ensure that the following preconditions are fulfilled if you wish to control the FM 357 via your user program:

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

The block library with the basic functions it contains is stored as standard in directory **[STEP7 directory]\S7LIBS\FM357_LI**.

- The link between the PG and the S7-300 CPU must already be set up (see Figures 4-1 and 4-2).

Blocks

To start working with the FM357, please proceed as follows:

1. Module configuration, see Section 5.2 under points 1. to 5.
2. Save and translate the created hardware project by selecting menu command **Station ► Save and Translate**.
3. The configured CPU and FM357 are now included in your project in the *SIMATIC Manager*. Select **SIMATIC 300 Station – CPUxxx – S7 program**. Open the installed S7 library (FM357_LI) and copy the following from this to your project:
 - Symbols
 - STL source FM357OBNx (x = configured no. of FM357)
 - Blocks

4. Go into your project and open the copied STL source (starts the LAD-STL editor). Enter the FM 357 module address at the correct place in OB 100 (see block FB1). Enter your user program at the correct place (USER program) in OB 1. Select menu commands **File ▶ Save** and **File ▶ Translate** to generate organization blocks (OB 1, OB 82, OB 100) from the STL source.
5. Select in the *SIMATIC Manager* under **SIMATIC 300 Station – CPUxxx – S7 Program – Blocks**, load all the S7 blocks (including system data) to your CPU and start up your system again.

If the startup between the CPU and FM357 has been successful (approximately 1 minute), the bit in the relevant user data block “NC signals”, DBX7.2, ANLAUF is set to FALSE.

The following table shows an overview of FBs/FCs and DBs on the CPU which need to be supplied with parameters and/or signals and data to allow communication with the FM 357.

Table 6-1 Standard function blocks for the FM 357

Function Block No.	Function Block Name	Significance	DB assignment
FB 1	RUN_UP	Initialization	DB 7
FC 22	GFKT	Startup, basic functions and operating modes	1. FM 357 – user DB 21, 31...34 2. FM 357 – user DB 22, 36...39 3. FM 357 – user DB 23, 41...44
FC 24	POS_AX	Positioning of linear and rotary axes	–
FB 2	GET	Read NC variable	DB for NC-VAR selector (default DB 120) DB for variables
FB 3	PUT	Write NC variable	
FB 4	PI	Select program, acknowledge error	DB 16, DB for program no.
FC 5 ¹⁾	GF_DIAG	Standard function, diagnostic alarm	–
FC 9	ASUP	Start an asynchronous subroutine	–

1) You do not need to parameterize this block

User DBs

Depending on the FM 357 configuration (e.g. number of axes, one to three FM 357), the user data blocks are set up internally during power-up.

- User DB “NC signals”
- User DBs “Axis signals”

Linking in the user program

The following diagram shows you how the FM 357, the user data blocks (DB for "NC signals" and DB for "Axis signals") and the standard function blocks communicate.

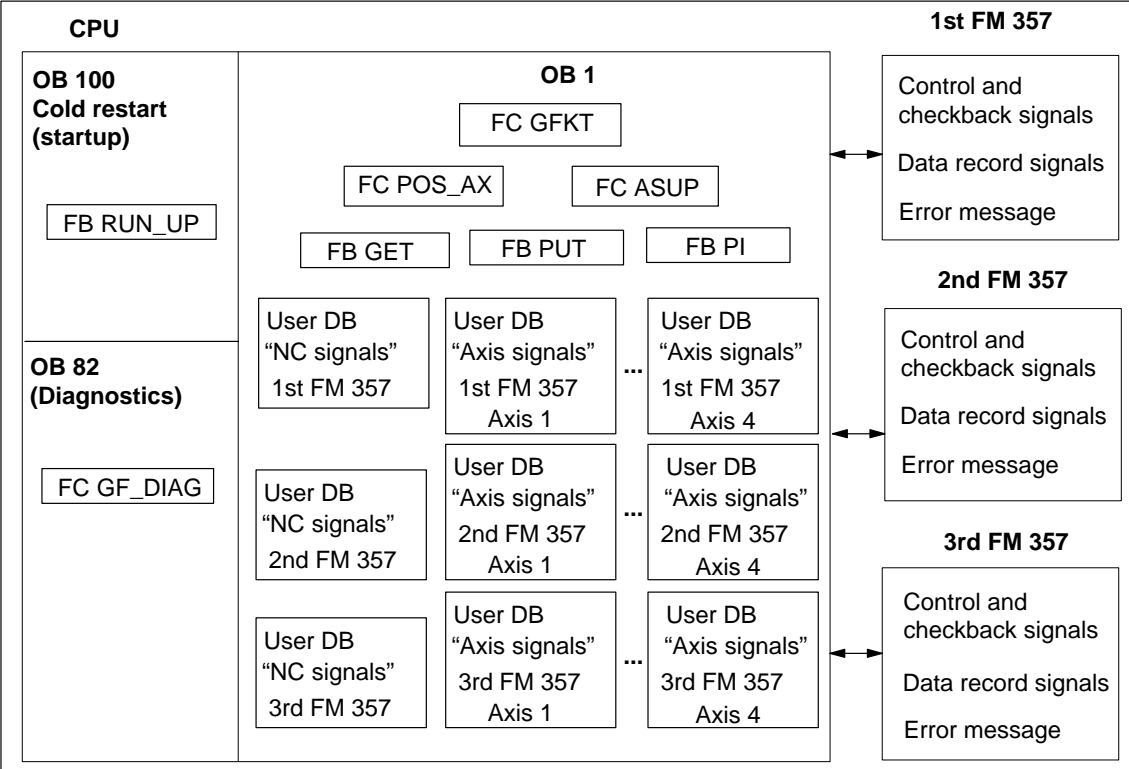


Figure 6-1 Block diagram of communication between CPU and three FM 357

Note

Further FCs, FBs and DBs are required internally to operate one to three FM 357 modules.

- FC 1, 2, 12, 23, 28
- FB 6, 18
- DB 1, 5, 15

Each transmission of data block signals or data with "read/write data block" takes about 4 ms (1 CPU cycle) for central configurations and several CPU cycles for distributed configurations. Data block transmissions should be activated only when required. Data blocks are transmitted in response to an FC 22 call by DATEN_L and DATEN_S. In addition, data blocks are automatically transferred when FB 2, FB 3, FB 4 and FC 24 are called.

The CPU cycle time must be set to > approx. 8 ms.

NC-VAR selector

You will need the NC-VAR selector to read and write variables (e.g. machine data, actual position, R parameters, velocities, etc.) of the FM 357 module (FB 2 and FB 3).

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

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

- The “NC-VAR selector” tool
 - This tool is designed for use on a control family. Use only the variables which are relevant for your application.
- Description of Functions. *Basic Machine (Part 1), Basic PLC Program (P3)*
Order No.: 6FC5 297-4AC20-0BP1

Installation:

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

Section overview

Section	Title	Page
6.1	FB 1: RUN_UP – Standard function, run-up section	6-5
6.2	FC 22: GFKT – Standard functions and operating modes	6-7
6.3	FC 24: POS_AX – Positioning of linear and rotary axes	6-12
6.4	FB 2: GET – Read NC variable	6-16
6.5	FB 3: PUT – Write NC variable	6-22
6.6	FB 4: PI – Select program, acknowledge error	6-27
6.7	FC 5: GF_DIAG – Standard function, diagnostic alarm	6-31
6.8	FC 9: ASUP – Start asynchronous subprograms	6-33
6.9	User data blocks	6-36
6.10	User handling procedures for controlling axes	6-60
6.11	Application examples	6-62
6.12	Technical specifications	6-66

6.1 FB 1: RUN_UP – Basic function, startup section

Task

FB 1 must be called once in OB 100 with the relevant parameters. The corresponding user DBs “NC signals” (DB 21...DB 23) are initialized and generated.

Call options

DB 7 belongs to FB 1 as an instance DB.

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
	<pre>CALL FB 1, DB 7 (NCLaddr1 :=, NCLaddr2 :=, NCLaddr3 :=, NCCyclTimeout :=, NCRunupTimeout :=, NCKomm :=, User_Version :=, User_Date :=, User_Time :=);</pre>

Description of parameters

The following table shows all formal parameters of the RUN_UP function for the FM 357.

Table 6-2 FB 1 parameters

Name	Data type	I/O type	Value range	Meaning
NCLaddr1	INT	I	256...752 ¹⁾ 320 (default is slot 8)	I/O address of the 1st FM 357
NCLaddr2	INT	I	0 ²⁾	I/O address of the 2nd FM 357
NCLaddr3	INT	I	0 ²⁾	I/O address of the 3rd FM 357
NCCyclTimeout	S5time	I	Recommended: 200 ms	Cyclical sign-of-life monitoring of the FM 357
NCRunupTimeout	S5time	I	Recommended: 3 min	FM 357 power-up monitoring time

Parameter types: I = input parameter

1) See Manual *S7-300 Programmable Controller, Hardware and Installation*

2) If 2nd or 3rd FM 357 is not installed

Example: See supplied STL sources (FM357 OB n1...n3)

Table 6-2 FB 1 parameters, continued

Name	Data type	I/O type	Value range	Meaning
NCKomm	BOOL	I	–	CPU-FM communication services (FB 2/3/4: GET/PUT/PI) active
User version	DWORD	I	For structure, see example of call	User program version
User data	DWORD	I		User program date
User time	DWORD	I		User program time

Parameter types: I = input parameter

1) See Manual *S7-300 Programmable Controller, Hardware and Installation*

2) If 2nd or 3rd FM 357 is not installed

Example: See supplied STL sources (FM357 OB n1...n3)

Example of call

An example of how to call FB 1 in OB 100 is given below.

```

AWL
-----
ORGANIZATION_BLOCK OB 100

VAR_TEMP
  OB100_EV_CLASS      :BYTE;
  OB100_STARTUP      :BYTE;
  OB100_PRIORITY      :BYTE;
  OB100_OB_NUMBR     :BYTE;
  OB100_RESERVED_1   :BYTE;
  OB100_RESERVED_2   :BYTE;
  OB100_STOP          :WORD;
  OB100_STRT_INFO    :DWORD;
  OB100_DATE_TIME    :DATE_AND_TIME;
END_VAR

BEGIN
  Call FB 1, DB 7(

      NCLaddr1      :=320,           //position 8
      NCLaddr2      :=0,
      NCLaddr3      :=0,
      NCCyclTimeout :=S5T#200MS,
      NCRunupTimeout :=S5T#3M,
      NCKomm        :=TRUE,
      User_Version  :=DW#16#20030000, //user-specific
      User_Date     :=DW#16#980224002), //year, month, day
      User_Time     :=DW#16#123000002)); //hour, minute, second

  // INSERT USER PROGRAM1)
  // HERE
END_ORGANIZATION_BLOCK

```

1) At this point, you can insert startup defaults for your own special applications.

2) The last two zeros are filler bytes (double word format).

6.2 FC 22: GFKT – Basic functions and operating modes

Task

This function comprises the following:

- Startup and synchronization with the FM 357
- Generation of user DBs “Axis signals” according to parameterized axes
- Standard function operation between CPU and FM 357
- Setting the operating modes
- Operation of the axes in the various operating modes
- Startup and testing
- Writing of general and specific axis signals and data (acc. to user DBs “NC signals” and “Axis signals”)
- Reading of general and specific axis signals and data (acc. to user DBs “NC signals” and “Axis signals”)

The block must be called in the cyclical program (OB 1).

The selected operating mode is active for all axes of an FM 357. Block FC 22 is executed once per FM module in the CPU cycle. It must always be called before the other FBs and FCs! This ensures proper execution of the standard functions and cyclical exchange of the control and checkback signals.

The user has absolute access to data and control and feedback signals. These signals/data are an integral component of the assigned user DBs “NC signals” and user DBs “Axis signals” and are transferred from FC 22 to the FM 357 and vice versa by way of peripheral inputs/outputs or with “read/write data block”.

Call options

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
	<pre>CALL FC 22(FM357No :=);</pre>

Description of parameter

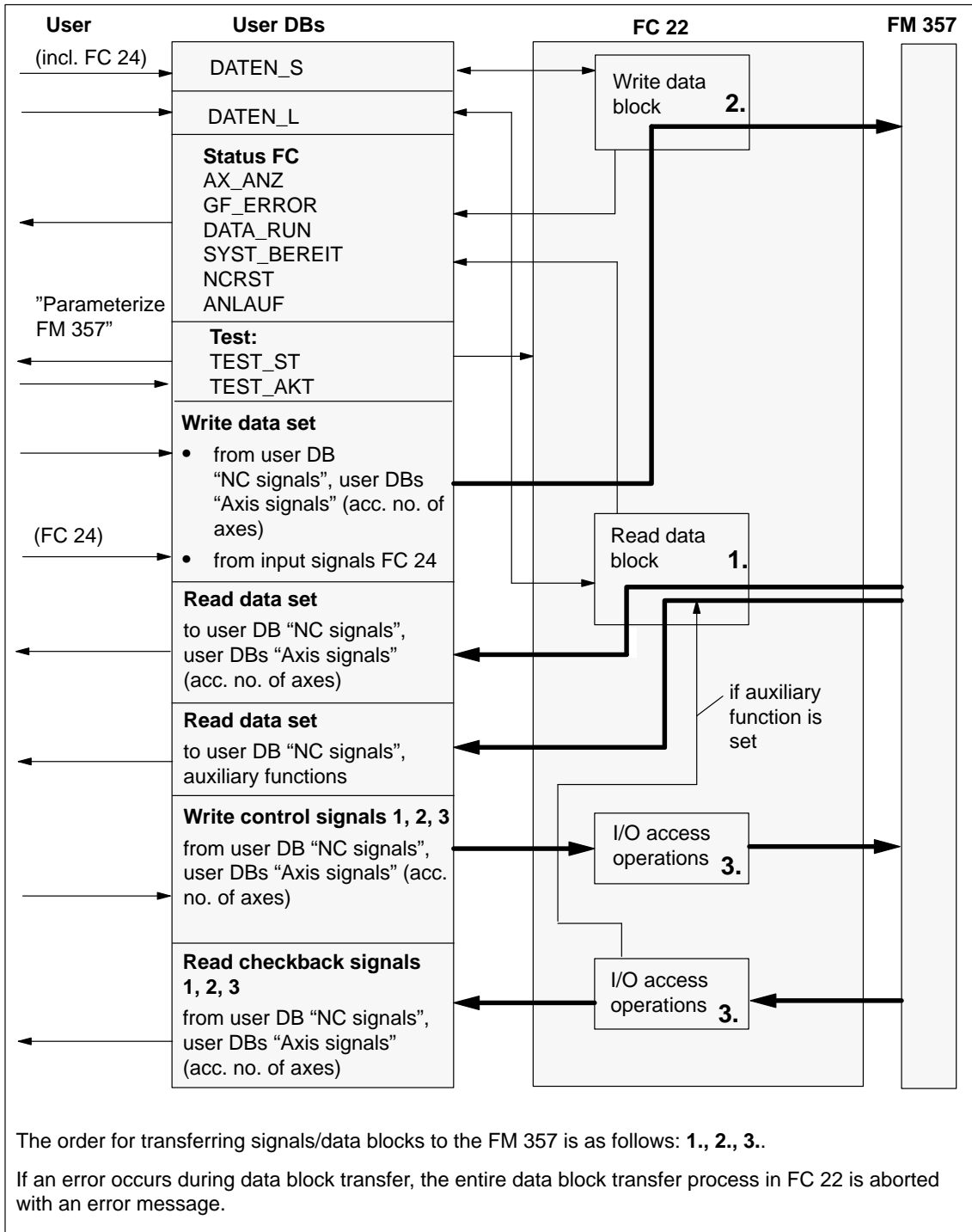
The following table shows the parameter associated with FC 22.

Name	Data type	I/O type	Significance	is ... by the user	is ... by the block
FM357No	INT	I	0, 1= 1st FM 357 2 = 2nd FM 357 3 = 3rd FM 357	entered	scanned

Parameter type: I = input parameter

Mode of operation

The function operates in conjunction with user DB "NC signals" and user DBs "Axis signals", whose DB numbers are defined in parameter FM357No when the function is called.



Every time the FC is called, the control signals are read from the relevant user DB and written to the FM 357. The checkback signals from the FM 357 are also read and stored in the relevant user DB.

If data block signals must be read from the FM 357 to the user DBs, then signal "Read data" (user DB "NC signals", DBX6.1) must be set.

If data block signals of user DBs must be written to the FM 357, then the appropriate signals in the user DBs and the signal "Write data" (user DB "NC signals", DBX6.0) must be set.

Other functions are as follows:

- Resetting of all signals/data in user DB "NC signal" and in user DB "Axis signals" on NC Restart (local variable OB 82_MDL_STOP). A new startup process begins.
- Transfer of data/parameters that are activated with FB 2, 3, 4.

Signals, status FC 22

The following signals must be set or interrogated by the user for the purpose of controlling FC 22.

Table 6-3 Signals, status FC 22

Signal user DB "NC signals"	is ... by the user	is ... by the block	Significance
DATEN_S DBX6.0	set	cancelled	Activation of a data block transfer to FM 357. The signal is cancelled after successful transmission or in the event of a transmission error.
DATEN_L DBX6.1	set	cancelled	Activation of a data block transfer from FM 357. The signal is cancelled after successful transmission or in the event of a transmission error.
TEST_ST DBX6.3	scanned	–	Selection of test mode with "Parameterize FM 357". The signal is set and cancelled in the "Startup window".
TEST_AKT DBX6.4	set/cancelled	–	Activation of test mode
DATA_RUN_W DBX6.2	scanned	set/cancelled	Data block transfer to FM 357 is active
DATA_RUN_R DBX6.6	scanned	set/cancelled	Data block transfer from FM 357 is active
AX_ANZ DBB2	scanned	set	Number of configured axes
SYST_BEREIT DBX7.0	scanned	set/cancelled	Communication link between CPU and FM 357 is ready
GF_ERROR DBW4	scanned	set/cancelled	Error code, communication error
NSRST DBX7.1	scanned	set/cancelled	An NC Restart is initiated manually.
ANLAUF ¹⁾ DBX7.2	scanned	set/cancelled	Startup is not yet finished.

1) **Important:** As long as ANLAUF (startup) is active (not cancelled), no USER program may be started for the FM 357.

Troubleshooting

If signal SYST_BEREIT has been reset, the communication link between the FM 357 and CPU is not ready (after power-up) or a communication error has occurred.

Table 6-4 FC 22 troubleshooting, GF_ERROR

Error code	Significance
Checkback code of SFC 58/59 in RET_VAL	See Reference Manual <i>System Software for S7-300/400; System and Standard Functions</i>
W#16#0100	Incorrect FM 357 firmware version
W#16#0101	Powerup timeout (3 min)
W#16#0102	Cyclical sign-of-life timeout (200 ms)

Note

If the FM 357 fails to start, the “Parameterize FM 357” tool can be used after about 6 minutes to create an online link to the FM 357, where, for example, the firmware version can be displayed.

Startup and parameterization

If the FM 357 is being started up or tested with the “Parameterize FM 357” tool (preselected by TEST_ST in the “Startup window” and activated by means of TEST_AKT from the user program), the signals/data are written to the user DB via the “Startup window”.

Once TEST_AKT has been set, the user program may not be used to influence the signals/data in the user DB.

Note

The STL source FM357_LI/OBFM357Nx supplied with the package already contains a structure which predefines the reactions to TEST_ST and TEST_AKT.

If startup mode is deselected and TEST_ST reset, the signals set from the “Startup window” are not cancelled.

Exception:

- The Feed Stop signal (user DB, “Axis signals”, DBX11.3) is set if desired by the “Parameterize FM 357” user (prompt window) on deselection of Test/Startup.
- The signals “Rapid traverse override active”, “Feed override active” (user DB, “NC signals”, DBX12.5/12.6) and “Activate override” (user DB, “Axis signals”, DBX12.7) are set to the valid status for TEST_ST selection.

Example of call

See library FM357_LI/OBFM357Nx

6.3 FC 24: POS_AX – Positioning of linear and rotary axes (CPU axis)

Task

An axis can be traversed from the CPU with FC POS_AX.

For the axis to be traversed from the CPU, it may not already be activated by the FM 357, e.g. checkback signals FR– (user DB, “Axis signals”, DBX15.6), FR+ (user DB, “Axis signals”, DBX15.7) are not set. When FC 24 is called and the “Start” parameter and input parameters activated, the CPU requests control of the axis from the FM 357 (axis exchange). The request is valid if checkback signal POS_AX (user DB, “Axis signals”, DBX15.5) is set.

When the positioning operation is complete (InPos is set), the “Start” parameter must be reset by the user. The axis request is then also reset, i.e. the axis is switched to a neutral status (POS_AX is reset) and can be programmed by the NC or requested again by the CPU.

A new positioning operation cannot commence until “InPos”, or in the case of an error, “Error”, has been reset.

FC 24 may only be called once in the CPU cycle for each axis.

The signals/parameters are conditioned in FC 24. Signals/parameters are transferred by means of FC 22.

Note

To achieve a high-speed succession of positioning operations, it is also possible to set a continuous axis request by the CPU using signal POS_ANFO (user DB, “Axis signals”, DBX1.0). This signal must be set by the user. Requests are acknowledged by POS_AX.

If FC 24 is subsequently called with “Start”, i.e. when an axis request has been made, the axis request and return are suppressed in FC 24. As a consequence, the user cycles needed for an axis exchange are omitted between successive positioning operations.

The axis is switched back to the neutral status when signal POS_ANFO is reset.

Interruption of motion:

- with feed stop, V_STOP (user DB, “Axis signals”, DBX11.3)
- NC Stop, STP (user DB, “NC signals”, DBX11.1) – can be continued with NC Start, ST (user DB, “NC signals”, DBX11.0)

Motion abort:

- with feed stop, V_STP (user DB, “Axis signals”, DBX11.3) and
- with deletion of distance to go, DEL_DIST (user DB, “NC signals”, DBX11.4) and
- with reset “Start” parameter if “InPos” or “Error” is set

Call options

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
<pre> graph LR EN --- FC24[FC 24] FM357No --- FC24 Start --- FC24 AxisNo --- FC24 IC --- FC24 Inch --- FC24 Pos --- FC24 FRate --- FC24 FC24 --- ENO FC24 --- InPos FC24 --- Active FC24 --- StartErr FC24 --- Error </pre>	<pre> CALL FC 24 (FM357No :=, Start :=, AxisNo :=, IC :=, Inch :=, Pos :=, FRate :=, InPos :=, Active :=, StartErr :=, Error :=); </pre>

Description of parameters

The following table describes the parameters associated with FC 24.

Name	Data type	I/O type	Value range	Significance
FM357No	INT	I	0, 1, 2, 3	0 or 1 = 1st FM 357 2 = 2nd FM 357 3 = 3rd FM 357
Start	BOOL	I	–	Request
AxisNo	BYTE	I	1 to 4	No. of axis to be traversed
IC	BOOL	I	–	FALSE = absolute TRUE = incremental
Inch	BOOL	I	–	FALSE = mm TRUE = inch
Pos	REAL	I	± 0.01 to $\pm 10^8$	Position of Linear axis: mm, inch Rotary axis: degrees
FRate ¹⁾	REAL	I	± 0.001 to $\pm 10^6$	Feedrate of Linear axis: mm/min, inch/min Rotary axis: degrees/min
InPos	BOOL	O	–	Position reached or function executed
Activ	BOOL	O	–	Active
StartErr	BOOL	O	–	Axis cannot be started
Error	BOOL	O	–	Traversing error

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

1) If the value = 0, the parameterized value "Positioning velocity" is activated in the FM

When absolute positioning is used for rotary axes, a negative feed value can be programmed in order to move the axis across the shortest distance. In incremental mode (parameter "IC" := TRUE), the leading sign of the "Pos" parameter can be used to determine the direction of movement. A positive leading sign causes movement in the plus direction. A negative sign causes movement in the minus direction.

Example of call

An example of how to call FC 24 is given below.

```

AWL
CALL FC 24(
    Start      :=M 36.0,      //job request
    FM357No    :=1,          //FM number
    AxisNo     :=2,          //number of axis to be traversed
    IC         :=TRUE,       //incremental traversing mode
    Inch       :=FALSE,      //dimensions in mm
    Pos        :=100.0,      //position
    FRate      :=1000.0,     //feed
    InPos      :=M 36.1,     //position reached
    Activ      ;=M 36.2,     //positioning active
    StartErr   :=M 36.3,     //start error
    Error      :=M 36.4);    //error
    
```

For another application example, see also supplied example FM357_EX\EXAMPLE3.

Troubleshooting

If parameter "Error" = TRUE, then an error number is entered in user DB "Axis signals", DBB33 (POS_FENR).

The error code is deleted when the start signal is reset after an error message.

Table 6-5 FC 24 troubleshooting

State	Significance
2	Axis is not parameterized
30	The axis was transferred to the FM before the motion was finished (e.g. NC-Reset)
115	The programmed position was not reached
125	DC (shortest distance) not possible
126	Minus absolute value not possible
127	Plus absolute value not possible
130	Software limit switch plus
131	Software limit switch minus
132	Working area limitation plus
133	Working area limitation minus

Timing diagram, FC 24 with axis exchange

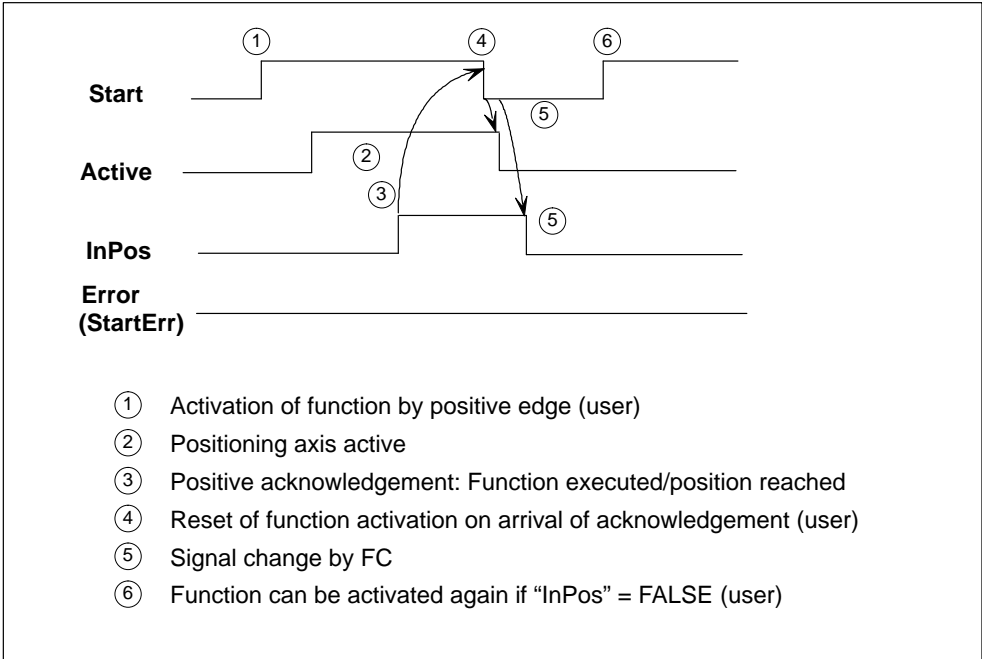


Figure 6-2 Timing diagram for FC 24

Timing diagram (error condition)

The following diagram shows the timing diagram for FC 24 under error conditions

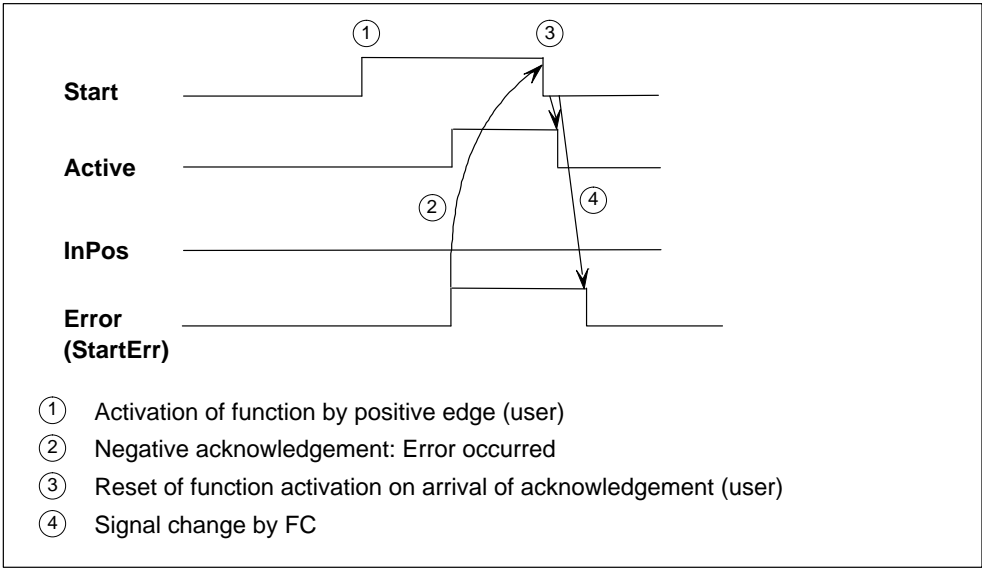


Figure 6-3 Timing diagram for the FC 24 (error case)

6.4 FB 2: GET – Read NC variable

Task

FB GET can be used to read variables from the FM 357.

FB 2 includes a DB from the user area.

Calling FB 2 on a positive edge change on control input Req starts a request to read the variables referenced by Addr1...Addr8, and, after the read operation, to copy them into the CPU operand areas referenced by RD1...RD8. Successful completion of the read operation is indicated by TRUE on status parameter NDR (new data received).

The read operation may last for several CPU cycles (generally 1...2 cycles for central configuration). The block should be called cyclically (OB 1).

Any errors which occur are indicated by Error and State.

In order to reference the variables, all the required variables are first selected with the "NC-VAR selector" tool, and generated in a data block as statement list source code. A name must be assigned to this DB in the symbol list. "DB name.variable name" is transferred as the FM variable address parameter (addr1...addr8) when FB 2 is called.

Call options

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
FB 2	CALL FB 2(
— 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	FM357No :=,
— FM357No	Error :=,
— RD1	NDR :=,
— RD2	State :=,
—	RD1 :=,
— RD8	RD2 :=,
	...
	RD8 :=);

Addressing a variable

For a number of variables it is necessary to select area no. and/or line or column in the NC-VAR selector. It is possible to select a basic type for such variables, i.e. area/column/line are preset to "Zero" (see DB 120).

The contents of the area no., line and column specified by the NC-VAR selector are checked for "zero" in the FB. If they contain "zero", the value of the input parameter is accepted. Before calling FB Get, the user must pass the desired parameters (UnitX/ColumnX/LineX).

Unit corresponds here to the area no., column and line.

Note

FB 2 can read variables only if parameter NCKomm has been set to TRUE (in OB 100: FB 1, DB 7).

If the communication link between the CPU and FM 357 (FB 2, 3, 4) is aborted by POWER OFF/EMERGENCY STOP/Acknowledgement/NC-Reset, the start jobs in the first OB 1 run after restart or NC-Reset must be deleted (Signal Req = FALSE).

When reading variables, only variables from one FM 357 can be addressed in a given request (FB 2 call) by reference to Addr1...Addr8.

Description of parameters

The following table describes the parameters associated with the GET function.

Table 6-6 FB 2 parameters

Name	Data type	I/O type	Value range	Meaning
Req	BOOL	I	–	Start request on positive edge
NumVar	INT	I	1...8 (corresponds to addr1...addr8)	Number of variables to be read
Addr1...Addr8	ANY	I	[DBName].[Var-name]	Variable name from NC-VAR selector
Unit1...Unit8	BYTE	I	–	Area address, optional for addressing variables
Column1...Column8	WORD	I	–	Column address, optional for addressing variables
Line1...Line8	WORD	I	–	Line address, optional for addressing variables
FM357No	INT	I	0, 1, 2, 3	0 or 1 = 1st FM 357 2 = 2nd FM 357, 3 = 3rd FM 357
Error	BOOL	O	–	Request was given negative acknowledgement or could not be executed

Table 6-6 FB 2 parameters, continued

Name	Data type	I/O type	Value range	Meaning
NDR	BOOL	O	–	Request was successfully executed. Data are available
State	WORD	O	–	see “Troubleshooting”
RD1...RD8	ANY	I/O	P#Mn.n BYTE x... P#DBnr.dbxm.n BYTE x	Destination area for read data

Parameter types: I = input parameter, O = output parameter,
I/O = in/out parameter (initiation parameter)

Troubleshooting

Status parameter Error is set to TRUE when it has not been possible to execute a job. The cause of the error is encoded at the State block output. The error code is deleted when the start signal is reset after an error message.

Table 6-7 FB 2 troubleshooting

State		Significance	Note
High byte	Low byte		
1 to 8	1	Access error	The high byte contains the number of the variable in which the error has occurred.
0	2	Error in request	Incorrect combination of variables in a job
0	3	Negative acknowledgement, request not executable	Internal error, remedy: NC Reset
1 to 8	4	Not enough local user memory available	Read variable is longer than specified in RD1...RD8; high byte contains the number of the variable in which the error has occurred
0	5	Format conversion error	Error in converting variable type double: Variable is not in the S7-REAL range
0	6	Serial data buffer is full	Request must be repeated, because the queue is full
0	7	Variable does not exist	Parameter “NCKomm” is not enabled
1 to 8	8	Invalid destination area (RD)	RD1...RD8 cannot be local data
0	9	Communication system busy	Request must be repeated
1 to 8	10	Error in addressing a variable	Unit or Column/Line contains value 0
0	11	Invalid variable address	Check Addr (or variable name), Area, Unit

Configuring sequence

The following configuring sequence must be followed to select variables:

1. Set up directory ...\`nc_var\ablage`.
2. Select the NC-VAR selector.
3. Select menu **Complete list ▶ Select** to go to dialog window **Select complete list**.
4. Then select list sw2.357 in directory ...\`nc_var\data`.
Open file `ncvar357.mdb`. This contains all the NC variables of the FM 357.
5. Select the variables you need for your project (use Help).
You may need to assign an axis number and other relevant parameters to the variables.
Confirm your inputs with OK and the selected variables will be transferred to your project.
6. Store this project under `[name].var` in directory ...\`nc_var\ablage`.
7. Select menu **Code ▶ Generate** to generate a STEP 7 source file `[name].awl` which contains a DB in ASCII format.
8. Close the NC-VAR selector.
9. Open your STEP 7 project.
10. Store the source file `[name].awl` in menu **Insert ▶ External source** in directory **Sources**.
Open the source and translate it.
11. Create the DB (default: DB 120) with the relevant address data.
12. Enter the name of the DB you have created in the symbol list so that the address parameters can be accessed symbolically in the user program.
13. Set the parameters for FB 2.

Variables, examples from complete list

Variable	Enter additional parameters in "Line"	Significance
<code>C_SMA_actToolBasePos</code>	Axis number	Actual position
<code>C_SMA_cmdToolBasePos</code>	Axis number	Set position
<code>C_SMA_ToolBaseDistTogGo</code>	Axis number	Distance to go
<code>C_SEMA_actFeedRate</code>	Axis number	Actual speed
<code>C_SMA_name</code>	Axis number	Axis name
<code>N_SALAL_textIndex</code>	–	Error number
<code>C_RP_rpa</code>	R parameter no. + 1	R parameters

Timing diagram

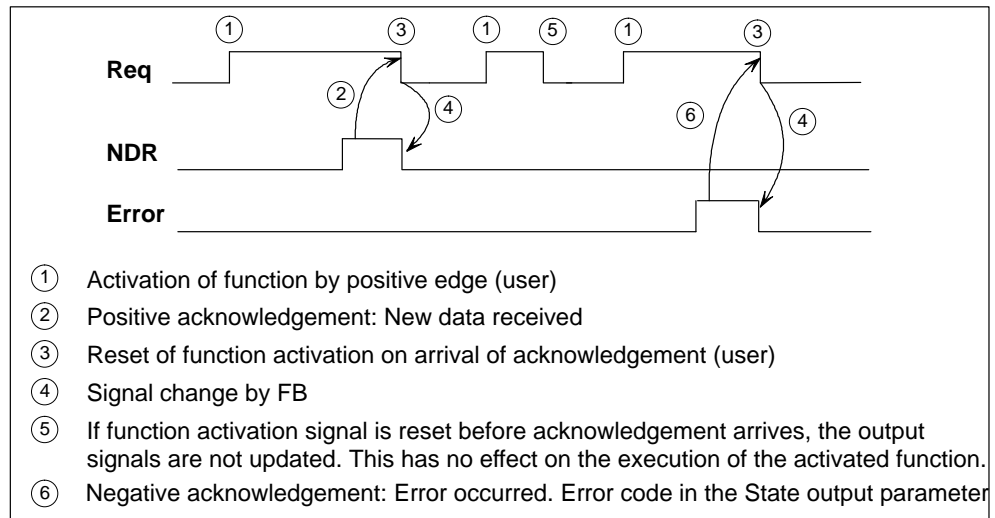


Figure 6-4 Timing diagram for FB 2

1st example of call

Indirect reading of two R parameters whose addresses are stored in DB 120. For direct addressing, see examples FM357_EX/EXAMPLE1.

- **Selection of data with NC Var selector** and storage in file DB120.var; followed by generation of file DB120.awl

Area	Function Block	Name	S7 type	S7 name
C[1]	RP	rpa[0]	Real	C1_RP_rpa0_1

The S7 (ALIAS) name from the NC-VAR selector has been selected to assign the variable name as an S7 name and make it symbolically accessible.

File DB120.awl must be compiled and the block transferred to the CPU.

- **Entry of name in the S7 symbol list** (e.g. NCVAR for DB 120):

Symbol	Address	Data type	Comments
NCVAR	DB 120	DB 120	Variable selection from NC

The R parameter number is specified in the LineX parameter.

In the call example, DB 110 is an unassigned data block and generated as an instance DB of FB 2.

```

AWL
CALL FB 2, DB 110 (
    Req      :=M 37.0,           //Request
    NumVar   :=2,               //number of variables to be read
    Addr1    :=NCVAR.C1_RP_rpa0_1, //declared variables from
                                           //DB 120 "NCVAR"
    Line1    :=W#16#1,         //line number
    Addr2    :=NCVAR.C1_RP_rpa0_1,
    Line2    :=W#16#2,
    FM357No  :=1,               //FM number
    Error    :=M 37.1,         //error
    NDR      :=M 37.2,         //error message
    State    :=MW 38,          //error status
    RD1      :=P#M 40.0 REAL 1, //destination area
    RD2      :=P#M44.0 REAL 1); // (S7 type from NC-VAR selector)
    
```

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

2nd example of call

For indirect addressing, see supplied example FM357_EX\EXAMPLE2.

Addressing a variable

For a number of variables it is necessary to select area no. and/or line or column in the NC-VAR selector. It is possible to select a basic type for such variables, i.e. area/column/line are preset to "Zero" (see DB 120).

The contents of the area no., line and column specified by the NC-VAR selector are checked for "zero" in the FB. If they contain "zero", the value of the input parameter is accepted. Before calling FB PUT, the user must pass the desired parameters (UnitX/ColumnX/LineX).

Unit corresponds here to the area no., column and line.

Note

FB 3 can write variables only if parameter NCKomm has been set to TRUE (in OB 100: FB 1, DB 7).

If the communication link between the CPU and FM 357 (FB 2, 3, 4) is aborted by POWER OFF/EMERGENCY STOP/Acknowledgement/NC-Reset, the start jobs in the first OB 1 run after restart or NC-Reset must be deleted (Signal Req = FALSE).

When writing variables, only variables from one FM 357 can be addressed in a given request (FB 3 call) by reference to Addr1...Addr8.

Description of parameters

The following table describes the parameters associated with the PUT function.

Name	Data type	I/O type	Value range	Meaning
Req	BOOL	I	–	Start request on positive edge
NumVar	INT	I	1...8 (corresponds to addr1...addr8)	Number of variables to be written
Addr1...Addr8	ANY	I	[DBName].[Var-name]	Variable name from NC-VAR selector
Unit1...Unit8	BYTE	I	–	Area address, optional for addressing variables
Column1...Column8	WORD	I	–	Column address, optional for addressing variables
Line1...Line8	WORD	I	–	Line address, optional for addressing variables
FM357No	INT	I	0, 1, 2, 3	0 or 1 = 1st FM 357 2 = 2nd FM 357, 3 = 3rd FM 357
Error	BOOL	O	–	Request was given negative acknowledgement or could not be executed
Done	BOOL	O	–	Request was successfully executed.
State	WORD	O	–	see "Error evaluation"
SD1...SD8	ANY	I/O	P#Mn.n BYTE x... P#DBnr.dbxm.n BYTE x	Data to be written

Parameter types: I = input parameter, O = output parameter,
I/O = in/out parameter (initiation parameter)

Troubleshooting

Status parameter Error is set to TRUE when it has not been possible to execute a job. The cause of the error is encoded at the State block output: The error code is deleted when the start signal is reset after an error message.

Table 6-8 FB 3 troubleshooting

State		Significance	Note
High byte	Low byte		
1 to 8	1	Access error	The high byte contains the number of the variable in which the error has occurred.
0	2	Error in request	Incorrect combination of variables in a job

Example of call

Writing of three R parameters:

- **Selection of data with NC Var selector** and storage in file DB120.var; followed by generation of file DB120.awl

Area	Function Block	Name	S7 type	S7 name
C[1]	RP	rpa[5]	Real	C1_RP_rpa5_1
C[1]	RP	rpa[11]	Real	C1_RP_rpa11_1
C[1]	RP	rpa[14]	Real	C1_RP_rpa14_1

The S7 (ALIAS) name from the NC-VAR selector has been selected to assign the variable name as an S7 name and make it symbolically accessible.

File DB120.awl must be compiled and the block transferred to the CPU.

- **Entry of name in the S7 symbol list** (e.g. NCVAR for DB 120):

Symbol	Address	Data type	Comments
NCVAR	DB 120	DB 120	Variable selection from NC

In the call example, DB 111 is an unassigned data block and generated as an instance DB of FB 3.

AWL

```
CALL FB 3, DB 111(

Req          :=M 100.0,           //Request
NumVar       :=3,                //write 3 variables
Addr1        :=NCVAR.C1_RP_rpa5_1, //declared variables from
                                           //DB 120 "NCVAR"

Addr2        :=NCVAR.C1_RP_rpa11_1,
Addr3        :=NCVAR.C1_RP_rpa14_1,
FM357No      :=1,                //FM number
Error        :=M 102.0,           //error
Done         :=M 100.1,           //finished message
State        :=MW 104,            //error status
SD1          :=P#DB99.DBX0.0 REAL 1, //data to be written
SD2          :=P#DB99.DBX4.0 REAL 1, //data to be written
SD3          :=P#M110.0 REAL 1);
```

6.6 FB 4: PI – Select program, acknowledge error

Task

FB PI can be used to select an NC program and acknowledge an error in the FM 357. A separate DB from the user area must be assigned to the FB 4 call.

A request is started by calling FB 4 on a positive edge change at control input Req. Successful completion is indicated by TRUE on status parameter Done.

Any errors which occur are indicated by Error and State.

Data block "PI" (DB 16) contains the internal descriptions of the PI service. A name must be assigned to this DB in the symbol list ("PI" in the example).

Execution of the PI service lasts for several CPU cycles (generally 1...2 cycles for central configurations). The block can only be called up during cyclical execution.

Note

FB 4 can start PI services only if parameter NCKomm has been set to TRUE (in OB 100: FB 1, DB 7).

Call options

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
	<pre>CALL FB 4(Req :=, PIService :=, Unit :=, Addr1 :=, Addr2 :=, FM357No :=, Error :=, Done :=, State :=);</pre>

Description of parameters

The following table describes the parameters associated with the PI function.

Name	Data type	I/O type	Value range	Meaning
Req	BOOL	I	–	Request
PIService	ANY	I	[DBName].[Var-name]	PI service: SELECT to select a program or CANCEL to acknowledge an error
Unit	INT	I	1	Area number
Addr1...Addr2	ANY	I	[DBName].[Var-name]	Reference to string specification according to selected PI service
FM357No	INT	I	0, 1, 2, 3	0 or 1 = 1st FM 357 2 = 2nd FM 357 3 = 3rd FM 357
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,

Troubleshooting

Status parameter Error is set to TRUE when it has not been possible to execute a job. The cause of the error is encoded at the State block output: The error code is deleted when the start signal is reset after an error message.

Table 6-9 FB 4 troubleshooting

State	Significance	Note
3	Negative acknowledgement, request not executable	Internal error, remedy: NC Reset
6	Serial data buffer is full	Request must be repeated, because the queue is full
7	Option not enabled	Parameter "NCKomm" is not enabled
9	Communication system busy	Request must be repeated

Timing diagram

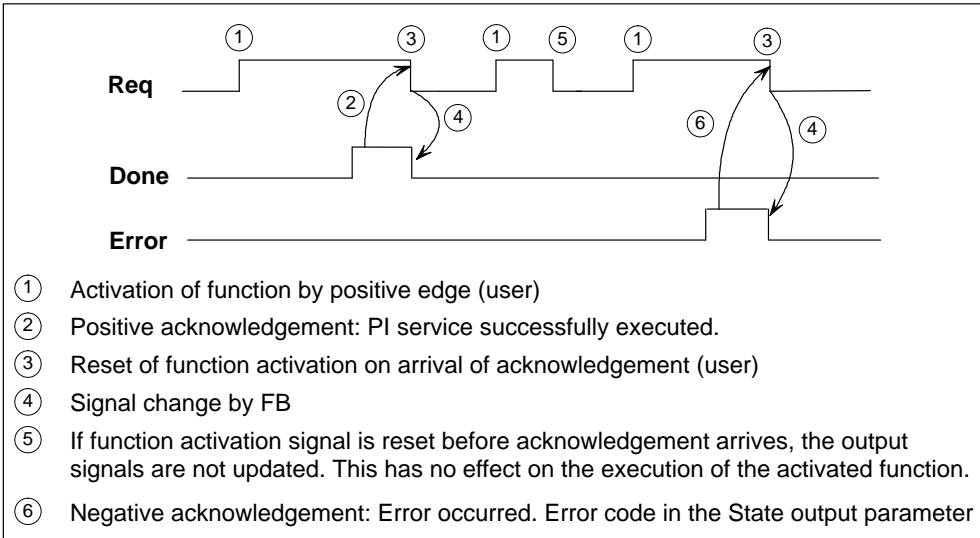


Figure 6-6 Timing diagram for FB 4

Mode of operation

Select a program for execution (SELECT)

- **Function**
 A program stored on the FM 357 is selected for execution. This is only possible if the file is suitable for execution. The path names and program names must be entered as described in the Programming Guide, Section 10.
- **Possible block types**
 - Main program MPF
 - Subprogram SPF
- **Parameterization**

Table 6-10 SELECT parameterization

Name	Data type	Value range	Significance
Req	BOOL	–	Request
PIService	ANY	SELECT	Select program
Unit	INT	1	Area number
Addr1	STRING		Path name ¹⁾
Addr2	STRING		Program name ²⁾
FM357No	INT	0, 1, 2, 3	0 or 1 = 1st FM 357 2 = 2nd FM 357, 3 = 3rd FM 357

1) Main program: '/_N_MPF_DIR/'; subroutine: '/_N_SPF_DIR/'
 2) Main program: '_N_<Name>_MPF'; subroutine: '_N_<Name>_SPF'

Acknowledgement of errors (CANCEL)

- **Function**

This signal is sent directly to the FM 357 via an OP or the “Parameterize FM 357” tool.

- **Parameterization**

Table 6-11 Acknowledgement of errors (CANCEL)

Name	Data type	Value range	Significance
Req	BOOL	–	Request
PIService	ANY	CANCEL	Acknowledge error
Unit	INT	1	Area number
FM357No	INT	0, 1, 2, 3	0 or 1 = 1st FM 357 2 = 2nd FM 357, 3 = 3rd FM 357

Example of call (program selection)

Entry PI for DB 16 and PROG for DB 124 in the symbol list

Symbol	Address	Data type	Comments
PI	DB 16	DB 16	PI service description
PROG	DB 124	DB 124	PI service program

In the call example, DB 112 is an unassigned data block and generated as an instance DB of FB 4.

```

AWL
-----
DATA_BLOCK DB 124
STRUCT
  PName:string[32] :='_N_TEST_MPF';
  Path:string[32] :='/_N_MPF_DIR/';
END_STRUCT
BEGIN
END_DATA_BLOCK

FUNCTION FC "PICall" :VOID

CALL FB 4, DB 112(
  Req           :=M 0.0,           //request
  PIService     :=PI.SELECT,       //program selection
  Unit          :=1,               //area number
  Addr1         :=PROG.Path,       //path, main program
  Addr2         :=PROG.PName,      //program name
  FM357No       :=1,               //FM number
  Error         :=M 1.0,           //error
  Done          :=M 1.1,           //finished message
  State        :=MW 2);           //error status
    
```

6.7 FC 5: GF_DIAG – Basic function, diagnostic alarm

Task

FC 5 acquires the diagnostic signals output by an FM 357 (see Table 6-12), incoming and outgoing.

The associated FM 357 address that has signalled the diagnostic alarm can be found under local variable OB82_MDL_ADDR.

If an NC-Restart is initiated in the FM 357 (via OP or “Parameterize FM 357” tool, e.g. after modification of machine data), a corresponding diagnostic alarm is output (OB82_MDL_STOP), acquired by FC 5 and evaluated in basic function program FC 22. After a reaction program has been started, an outgoing diagnostic alarm signals the reaction in the FM 357. During this period, signal “NC_BEREIT” (NC READY) (user DB “NC signals”, DBX24.4) is reset and “NCRST” (user DB “NC signals”, DBX7.1) set.

FC 5 must be called only once in OB 82, even for a number of FM 357 modules.

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

Table 6-12 Diagnostic alarms

Error code	Significance
W#16#0010	Diagnostic alarm “NC-Restart” (NCRST)
W#16#0011	Diagnostic alarm “Hardware fault FM 357” (INT_FAULT)
W#16#0012	External fault “Local P bus segment” (EXT_FAULT)
W#16#0013	Diagnostic alarm “Time watchdog response” (WTCH_DOG_FLT)
W#16#0014	Diagnostic alarm “Internal FM supply voltage failure” (INT_PS_FLT)

The error code is stored in GF_ERROR (user DB “NC signals”, DBW4).

Example of call

The following example contains the relevant defaults settings for OB 82 and the basic function call in FC 5.

```

AWL
VAR_TEMP
  OB82_EV_CLASS      :BYTE;           //16#39, Event class 3, Entering
                                     //event state, Internal fault event
  OB82_FLT_ID        :BYTE;           //16#XX, Fault identification code
  OB82_PRIORITY      :BYTE;           //26/28 (Priority of 1 is lowest)
  OB82_OB_NUMBR      :BYTE;           //82 (Organization block 82, OB82)
  OB82_RESERVED_1    :BYTE;           //Reserved for system
  OB82_IO_FLAG        :BYTE;           //Input (01010100), Output
                                     //(01010101)
  OB82_MDL_ADDR      :INT;            //Base address of module with fault
  OB82_MDL_DEFECT     :BOOL;           //Module defective
  OB82_INT_FAULT      :BOOL;           //Internal fault
  OB82_EXT_FAULT      :BOOL;           //External fault
  OB82_PNT_INFO       :BOOL;           //Point information
  OB82_EXT_VOLTAGE    :BOOL;           //External voltage low
  OB82_FLD_CONNCTR    :BOOL;           //Field wiring connector missing
  OB82_NO_CONFIG      :BOOL;           //Module has no configuration data
  OB82_CONFIG_ERR     :BOOL;           //Module has configuration error
  OB82_MDL_TYPE       :BYTE;           //Type of module
  OB82_SUB_NDL_ERR    :BOOL;           //Sub-Module is missing or has error
  OB82_COMM_FAULT     :BOOL;           //Communication fault
  OB82_MDL_STOP       :BOOL;           //Module is stopped
  OB82_WTCH_DOG_FLT   :BOOL;           //Watch dog timer stopped module
  OB82_INT_PS_FLT     :BOOL;           //Internal power supply fault
  OB82_PRIM_BATT_FLT  :BOOL;           //Primary battery is faulty
  OB82_BCKUP_BATT_FLT :BOOL;           //Backup battery is faulty
  OB82_RESERVED_2     :BOOL;           //Reserved for system
  OB82_RACK_FLT       :BOOL;           //Rack fault, only for bus interface
                                     //module
  OB82_PROC_FLT       :BOOL;           //Processor fault
  OB82_EPROM_FLT      :BOOL;           //EPROM fault
  OB82_RAM_FLT        :BOOL;           //RAM fault
  OB82_ADU_FLT        :BOOL;           //ADC fault
  OB82_FUSE_FLT       :BOOL;           //Fuse fault
  OB82_HW_INTR_FLT    :BOOL;           //Hardware interrupt input faulty
  OB82_RESERVED_3     :BOOL;           //Reserved for system
  OB82_DATE_TIME      :DATE_AND_TIME; //Date and time OB82 started

END_VAR
BEGIN
  CALL FC 5;

// INSERT USER PRO-
// GRAM1) HERE

END_ORGANIZATION_BLOCK

```

- 1) If other diagnostics-triggering modules are connected to the CPU in addition to the FM 357, you can insert the reaction program here.

6.8 FC 9: ASUP – Start of asynchronous subroutines

Task

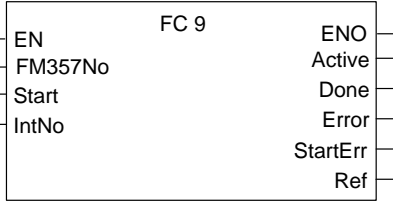
FC ASUP can be used to start subroutines in the FM. Subroutines can be started only if an NC program for this purpose has been set up (see Section 10.21, NC Programming and Section 9.12, Function). “Interrupt 8” must be declared for this function in the NC program. Once an asynchronous subroutine has been prepared in this way, it can be started by the CPU at any time. The NC program in progress is interrupted by the asynchronous subroutine. Only one asynchronous subroutine can be started.

The start parameter must be set to FALSE by the user once the routine has run (Done) or if an error has occurred.

To process the job, FC ASUP needs its own WORD parameter (**Ref**) from the global user memory area. This parameter is used internally and must not be changed by the user. Parameter **Ref** is initialized in the first OB 1 cycle; for this reason, **a call must be programmed for every FC ASUP.**

Alternatively, the user can initialize parameter **Ref** with FALSE during starting, thereby allowing conditional calls as well. When a conditional call is activated by parameter **Start**, it must remain TRUE until parameter **Done** has executed a status change from TRUE to FALSE.

Call options

Call in LAD notation (ladder diagram)	Call in STL notation (statement list)
 <pre> FC 9 EN --- FM357No --- Start --- IntNo --- ENO --- Active --- Done --- Error --- StartErr --- Ref --- </pre>	<pre> CALL FC 9(FM357No :=, Start :=, IntNo :=, Active :=, Done :=, Error :=, StartErr :=, Ref :=); </pre>

Description of parameters

The following table shows all formal parameters associated with the ASUP function.

Name	Data type	I/O type	Value range	Meaning
FM357No	INT	I	0, 1, 2, 3	0 or 1 = 1st FM 357, 2 = 2nd FM 357, 3 = 3rd FM 357
Start	BOOL	I	–	Request
IntNo	INT	I	8	Interrupt No.
Active	BOOL	O	–	Active
Done	BOOL	O	–	Asynchronous subroutine done
Error	BOOL	O	–	Error
StartErr	BOOL	O	–	Interrupt number not assigned
Ref	WORD	I/O	Global variable (MW, DBW, ...)	One word (for internal use)

Parameter types: I = input parameter, O = output parameter,
I/O = in/out parameter (initiation parameter)

Timing diagram

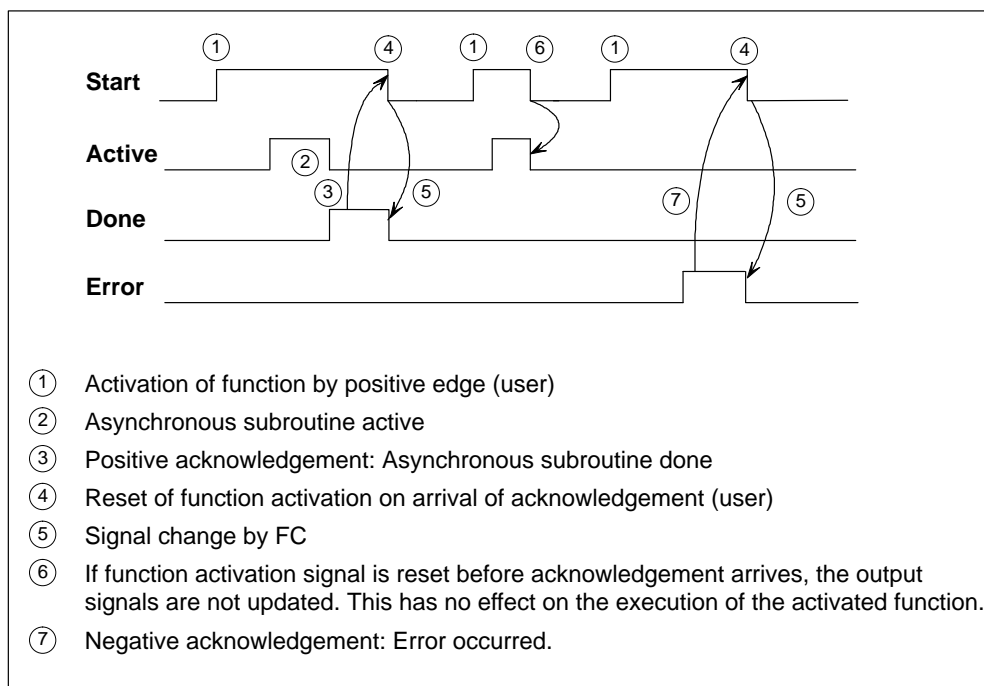


Figure 6-7 Timing diagram for FC 9

Example of call

An example of how to call FC 9 is given below.

```
AWL
CALLFC 9(
    FM357No    :=1           //FM number
    Start      :=M 202.0,    //start of an asynchronous subroutine
    IntNo      :=8,         //interrupt no. 8
    Activ      :=M 204.0,    //asynchronous subroutine active
    Done       :=M 204.1,    //asynchronous subroutine finished
    Error      :=M 204.4     //error
    StartErr   :=M 204.5     //starting error
    Ref        :=MW 200);
```


6.9 User data blocks (user DB)

General

The following user DBs are set up depending on the configuration (maximum of 3 FM 357s):

- User DB “NC signals”
 - User DB “NC signals” for the 1st FM 357 (assigned: DB 21)
 - User DB “NC signals” for the 2nd FM 357 (assigned: DB 22)
 - User DB “NC signals” for the 3rd FM 357 (assigned: DB 23)
- User DB “Axis signals”
 - User DB “Axis signals” for the 1st FM 357 (assigned: DB 31...34 for axes 1...4)
 - User DB “Axis signals” for the 2nd FM 357 (assigned: DB 36...39 for axes 1...4)
 - User DB “Axis signals” for the 3rd FM 357 (assigned: DB 41...44 for axes 1...4)

OL control and checkback signals

The control/checkback signals can be found in the following areas:

- User DB “NC signals”, DBD10 to DBD24 and DBD110 to DBD114
- User DB “Axis signals”, DBD10 to DBD14 and DBD110 to DBD114

Read/write data block

The signals for read/write data block can be found in the following areas:

- User DB “NC signals”, DBB30 through DBB79
- User DB “Axis signals”, DBB20 through DBB50

Auxiliary functions

The signals for auxiliary functions can be found in the following areas:

User DB “NC signals”, DBB80 through DBB105

6.9.1 User data block “NC signals”

General

The following table describes the structure of the user DB.

This description is valid for one or three FM 357 modules.

Table 6-13 User DB “NC signals”

Address		Variable	Data type	Comments
Absolute	Relative			
Signals, FC 22 status				
0.0	+0.0		BYTE	Reserved
1.0	+1.0		BYTE	Reserved
2.0	+2.0	AX_ANZ	BYTE	No. of axes (configuration)
3.0	+3.0		BYTE	Reserved
4.0	+4.0	GF_ERROR	WORD	Error in basic function
6.0	+6.0	DATEN_S	BOOL	Write data
6.1	+6.1	DATEN_L	BOOL	Read data
6.2	+6.2	DATA_RUN_W	BOOL	Data block transfer to FM is active
6.3	+6.3	TEST_ST	BOOL	Preselect test mode from parameterization tool
6.4	+6.4	TEST_AKT	BOOL	Activate test mode from user program
6.5	+6.5		BOOL	Reserved
6.6	+6.6	DATA_RUN_R	BOOL	Data block transfer from FM is active
6.7	+6.7		BOOL	Reserved
7.0	+7.0	SYST_BEREIT	BOOL	System is ready
7.1	+7.1	NCRST	BOOL	NC Restart
7.2	+7.2	ANLAUF	BOOL	Startup
7.3 to 7.7	+7.3 to +7.7		BOOL	Reserved
8.0	+8.0		WORD	Reserved
Control signals 1				
	10.0		STRUCT	Control signals 1
10.0	+0.0		BYTE	Reserved
11.0	+1.0	ST	BOOL	NC Start
11.1	+1.1	STP	BOOL	NC Stop
11.2	+1.2		BOOL	Reserved
11.3	+1.3	ESP	BOOL	Read-in disable
11.4	+1.4	DEL_DIST	BOOL	Delete the distance to go

Table 6-13 User DB “NC signals”, continued

Address		Variable	Data type	Comments
Absolute	Relative			
11.5	+1.5	SA	BOOL	Skip block
11.6	+1.6	QHF	BOOL	Acknowledge auxiliary function
11.7	+1.7		BOOL	Reserved
12.0	+2.0	AUTOMATIK	BOOL	”Automatic” mode
12.1	+2.1	MDI	BOOL	”MDI” mode
12.2	+2.2	TIPPEN	BOOL	”Jog” mode
12.3	+2.3	REFPKT	BOOL	”Reference point approach” mode
12.4	+2.4	AUTO_E	BOOL	”Automatic single block” mode
12.5	+2.5	EILG_KOR_WIR	BOOL	Rapid traverse override active
12.6	+2.6	VOR_KOR_WIR	BOOL	Feedrate override active
12.7	+2.7	RES	BOOL	NC Reset
13.0	+3.0	1 INC	BOOL	Increment 1
13.1	+3.1	10 INC	BOOL	Increment 10
13.2	+3.2	100 INC	BOOL	Increment 100
13.3	+3.3	1000 INC	BOOL	Increment 1 000
13.4	+3.4	10000 INC	BOOL	Increment 10 000
13.5	+3.5		BOOL	Reserved
13.6	+3.6	KONTIN	BOOL	Continuous traversing mode
13.7	+3.7		BOOL	Reserved
	=4.0		END_STRUCT	
Checkback signals 1				
	14.0		STRUCT	Checkback signals 1
14.0	+0.0		BYTE	Reserved
15.0	+1.0	PROGL	BOOL	Program running
15.1	+1.1	PROGW	BOOL	Program waiting
15.2	+1.2	PROG_ANGEH	BOOL	Program stopped
15.3	+1.3	PROG_UNTB	BOOL	Program interrupted
15.4	+1.4	PROG_ABGB	BOOL	Program aborted
15.5	+1.5	AHF	BOOL	Change auxiliary function
15.6	+1.6		BOOL	Reserved
15.7	+1.7	RES_Q	BOOL	Acknowledge NC Reset
16.0	+2.0	AUTOMATIK_A	BOOL	”Automatic” mode active
16.1	+2.1	MDI_A	BOOL	”MDI” mode active
16.2	+2.2	TIPPEN_A	BOOL	”Jog” mode active
16.3	+2.3	REFPKT_A	BOOL	”Reference point approach” mode active

Table 6-13 User DB “NC signals”, continued

Address		Variable	Data type	Comments
Absolute	Relative			
16.4 to 16.7	+2.4 to +2.7		BOOL	Reserved
17.0	+3.0	MNR	BYTE	M function number (17.0 to 17.6)
	=4.0		END_STRUCT	
Control signals 2				
	20.0		STRUCT	Control signals 2
20.0	+0.0		BYTE	Reserved
21.0	+1.0	B_OVERR	BYTE	Path override
22.0	+2.0		BYTE	Reserved
23.0	+3.0		BOOL	Reserved
23.1	+3.1	NOT_AUS	BOOL	EMERGENCY STOP
23.2	+3.2	NOT_AUS_Q	BOOL	Acknowledge EMERGENCY STOP
23.3 to 23.7	+3.3 to +3.7		BOOL	Reserved
	=4.0		END_STRUCT	
Checkback signals 2				
	24.0		STRUCT	Checkback signals 2
24.0	+0.0		BYTE	Reserved
25.0	+1.0	NOT_AUS_A	BOOL	EMERGENCY STOP active
25.1 to 25.3	+1.1 to +1.3		BOOL	Reserved
25.4	+1.4	NC_BEREIT	BOOL	NC ready signal
25.5	+1.5	NC_FE	BOOL	NC error active
25.6	+1.6	NC_BATFE	BOOL	NC battery error active
25.7 to 26.1	+1.7 to +2.1		BOOL	Reserved
26.2	+2.2	AX_REF	BOOL	All axes referenced
26.3	+2.3	AX_STEHEN	BOOL	All axes stationary
26.4 to 26.4	+2.4 to +2.5		BOOL	Reserved
26.6	+2.6	NC_FEOB	BOOL	Error without interruption of machining
26.7	+2.7	NC_FEMB	BOOL	Error with interruption of machining
27.0	+3.0		BYTE	Reserved
28.0	+4.0		WORD	Reserved
	=4.0		END_STRUCT	
Control signals 3				
	110.0		STRUCT	Control signals 3
110.0	+0.0		BYTE	Reserved
111.0	+1.0	SYNA_L1	BOOL	Disable synchronized action ID1
111.1	+1.1	SYNA_L2	BOOL	Disable synchronized action ID2

Table 6-13 User DB “NC signals”, continued

Address		Variable	Data type	Comments
Absolute	Relative			
111.2	+1.2	SYNA_L3	BOOL	Disable synchronized action ID3
111.3	+1.3	SYNA_L4	BOOL	Disable synchronized action ID4
111.4	+1.4	SYNA_L5	BOOL	Disable synchronized action ID5
111.5	+1.5	SYNA_L6	BOOL	Disable synchronized action ID6
111.6	+1.6	SYNA_L7	BOOL	Disable synchronized action ID7
111.7	+1.7	SYNA_L8	BOOL	Disable synchronized action ID8
112.0	+2.0	SYS_DBW_S	WORD	Write system variable
	=4.0		END_STRUCT	
Checkback signals 3				
	114.0		STRUCT	Checkback signals 3
114.0	+0.0		BYTE	Reserved
115.0	+1.0	SYNA_LA1	BOOL	Synchronized action ID1 disabled
115.1	+1.1	SYNA_LA2	BOOL	Synchronized action ID2 disabled
115.2	+1.2	SYNA_LA3	BOOL	Synchronized action ID3 disabled
115.3	+1.3	SYNA_LA4	BOOL	Synchronized action ID4 disabled
115.4	+1.4	SYNA_LA5	BOOL	Synchronized action ID5 disabled
115.5	+1.5	SYNA_LA6	BOOL	Synchronized action ID6 disabled
115.6	+1.6	SYNA_LA7	BOOL	Synchronized action ID7 disabled
115.7	+1.7	SYNA_LA8	BOOL	Synchronized action ID8 disabled
116.0	+2.0	SYS_DBW_L	WORD	Read system variable
118.0	+4.0		WORD	Reserved
	=4.0		END_STRUCT	
Read data set				
	30.0		STRUCT	Read data set
30.0	+0.0		BOOL	Reserved
30.1	+0.1	TASTER_1	BOOL	Probe 1 activated
30.2	+0.2	TASTER_2	BOOL	Probe 2 activated
30.3 to 30.7	+0.3 to +0.7		BOOL	Reserved
31.0	+1.0		BYTE	Reserved
32.0	+2.0	SW_NO0_MINUS	BOOL	Software cam minus 0
32.1	+2.1	SW_NO1_MINUS	BOOL	Software cam minus 1
...
32.7	+2.7	SW_NO7_MINUS	BOOL	Software cam minus 7
33.0	+3.0	SW_NO0_PLUS	BOOL	Software cam plus 0
33.1	+3.1	SW_NO1_PLUS	BOOL	Software cam plus 1

Table 6-13 User DB “NC signals”, continued

Address		Variable	Data type	Comments
Absolute	Relative			
...
33.7	+3.7	SW_NO7_PLUS	BOOL	Software cam plus 7
34.0	+4.0		WORD	Reserved
36.0	+6.0	DIG_EIN9	BOOL	Status of digital input 9 on local P bus
36.1	+6.1	DIG_EIN10	BOOL	Status of digital input 10 on local P bus
...
36.7	+6.7	DIG_EIN16	BOOL	Status of digital input 16 on local P bus
37.0	+7.0	DIG_EIN17	BOOL	Status of digital input 17 on local P bus
37.1	+7.1	DIG_EIN18	BOOL	Status of digital input 18 on local P bus
...
37.7	+7.7	DIG_EIN24	BOOL	Status of digital input 24 on local P bus
38.0	+8.0	DIG_AUS9	BOOL	Status of digital output 9 on local P bus
38.1	+8.1	DIG_AUS10	BOOL	Status of digital output 10 on local P bus
...
38.7	+8.7	DIG_AUS16	BOOL	Status of digital output 16 on local P bus
39.0	+9.0	DIG_AUS17	BOOL	Status of digital output 17 on local P bus
39.1	+9.1	DIG_AUS18	BOOL	Status of digital output 18 on local P bus
...
39.7	+9.7	DIG_AUS24	BOOL	Status of digital output 24 on local P bus
40.0	+10.0		WORD	Reserved
42.0	+12.0	M00/M01_A	BOOL	M00/M01 active
42.1	+12.1	M02/M30_A	BOOL	M02/M30 active
42.2 to 42.7	+12.2 to +12.7		BOOL	Reserved
43.0	+13.0		BYTE	Reserved
44.0	+14.0		BYTE	Reserved
45.0	+15.0		BYTE	Reserved
	=7.0		ENDE_DS	

Table 6-13 User DB “NC signals”, continued

Address		Variable	Data type	Comments
Absolute	Relative			
Write data set				
	54.0		STRUCT	Write data set
54.0	+0.0		BYTE	Reserved
55.0	+1.0		BYTE	Reserved
56.0	+2.0	SP_DIG_AUS9	BOOL	Disable digital output 9 on local P bus
56.1	+2.1	SP_DIG_AUS10	BOOL	Disable digital output 10 on local P bus
...
56.7	+2.7	SP_DIG_AUS16	BOOL	Disable digital output 16 on local P bus
57.0	+3.0		BYTE	Reserved
58.0	+4.0		WORD	Reserved
60.0	+6.0	SP_DIG_AUS17	BOOL	Disable digital output 17 on local P bus
60.1	+6.1	SP_DIG_AUS18	BOOL	Disable digital output 18 on local P bus
...
60.7	+6.7	SP_DIG_AUS24	BOOL	Disable digital output 24 on local P bus
61.0	+7.0		BYTE	Reserved
62.0	+8.0		WORD	Reserved
64.0 to 64.4	+10.0 to +10.4		BOOL	Reserved
64.5	+10.5	M01	BOOL	Activate M01
64.6 to 64.7	+0.6 to +0.7		BOOL	Reserved
65.0	+11.0		BYTE	Reserved
66.0	+12.0		WORD	Reserved
68.0	+14.0	VSP	BOOL	Activate feed disable
68.1 to 68.7	+14.1 to +14.7		BOOL	Reserved
69.0	+15.0	NC_STSP	BOOL	Activate NC start disable
69.1	+15.1		BOOL	Reserved
69.2	+15.2	STP_SG	BOOL	Activate NC stop at block boundary
69.3 to 69.7	+15.3 to +15.7		BOOL	Reserved
70.0	+16.0		DWORD	Reserved
74.0	+20.0		DWORD	Reserved
78.0	+24.0		BYTE	Reserved
79.0	+25.0		BYTE	Reserved
	=26		ENDE_DS	

Table 6-13 User DB "NC signals", continued

Address		Variable	Data type	Comments
Absolute	Relative			
Auxiliary functions				
	80.0		STRUCT	Auxiliary functions
80.0	+0.0	MNR_1	BYTE	M function number 1
81.0	+1.0	MNR_2	BYTE	M function number 2
82.0	+2.0	MNR_3	BYTE	M function number 3
83.0	+3.0	MNR_4	BYTE	M function number 4
84.0	+4.0	MNR_5	BYTE	M function number 5
85.0	+5.0		BYTE	Reserved
86.0	+6.0	HNR_1	WORD	H function number 1
88.0	+8.0	HWERT_1	DWORD	H function value 1 (REAL)
92.0	+12.0	HNR_2	WORD	H function number 2
94.0	+14.0	HWERT_2	DWORD	H function value 2 (REAL)
98.0	+18.0	HNR_3	WORD	H function number 3
100.0	+20.0	HWERT_3	DWORD	H function value 3 (REAL)
104.0	+24.0	TNR	WORD	T function number
	=26		ENDE_HILFS- FUNKTION	
120 to 145				Reserved

6.9.2 User data block “Axis signals”

General

The following table describes the structure of the user DB.

This description is valid for axis 1 to 4.

Table 6-13 User DB “Axis signals”

Address		Variable	Data type	Comments
Absolute	Relative			
0.0	0.0		BYTE	Reserved
1.0	+1.0	POS_ANFO	BOOL	Request positioning axis
1.1 to 1.7	+1.1 to +1.7		BOOL	Reserved
2.0	+2.0		DWORD	Reserved
6.0	+6.0		DWORD	Reserved
Control signals 1				
	10.0		STRUCT	Control signals 1
10.0	+0.0		BYTE	Reserved
11.0	+1.0		BOOL	Reserved
11.1	+1.1	VER_RPS	BOOL	Referencing delay
11.2	+1.2	DEL_DISTA	BOOL	Axial deletion of distance to go
11.3	+1.3	V_STP	BOOL	Feed STOP
11.4 to 11.5	+1.4 to 1.5		BOOL	Reserved
11.6	+1.6	R-	BOOL	Direction minus
11.7	+1.7	R+	BOOL	Direction plus
12.0	+2.0	SWN_AKT	BOOL	Activate software cam
12.1	+2.1	RFG	BOOL	Enable CL controller
12.2	+2.2	DRUE	BOOL	Stepper motor rotation monitoring
12.3	+2.3		BOOL	Reserved
12.4	+2.4	NFB	BOOL	Follow-up mode
12.5	+2.5			Reserved
12.6	+2.6	EILG_UEBERL	BOOL	Activate rapid traverse override
12.7	+2.7	OVERR_AKT	BOOL	Activate override
13.0	+3.0	OVERR	BYTE	Override
	=4.0		END_STRUCT	
Checkback signals 1				
	14.0		STRUCT	Checkback signals 1
14.0	+0.0		BYTE	Reserved

Table 6-13 User DB "Axis signals", continued

Address		Variable	Data type	Comments
Absolute	Relative			
15.0	+1.0	SYN	BOOL	Synchronized, referenced
15.1	+1.1	PEHG	BOOL	Position reached, stop (target range coarse)
15.2	+1.2	PEHF	BOOL	Position reached, stop (target range fine)
15.3	+1.3	SWN_A	BOOL	Software cams active
15.4	+1.4		BOOL	Reserved
15.5	+1.6	POS_AX	BOOL	Axis is a positioning axis of the CPU
15.6	+1.6	FR-	BOOL	Go_minus
15.7	+1.7	FR+	BOOL	Go_plus
16.0 to 16.2	+2.0 to +2.2		BOOL	Reserved
16.3	+2.3	NFB_A	BOOL	Follow-up mode active
16.4	+2.4	STEHT	BOOL	Axis stationary
16.5	+2.5	LR_A	BOOL	Position controller active
16.6	+2.6	DR_A	BOOL	Speed controller active
16.7	+2.7		BOOL	Reserved
17.0 to 17.1	+3.0 to +3.1		BOOL	Reserved
17.2	+3.2	DRUE_FE	BOOL	Error in stepper motor rotation monitoring
17.3 to 17.7	+3.3 to +3.7		BOOL	Reserved
	=4.0		END_STRUCT	
Control signals 2				
	110.0		STRUCT	Control signals 2
110.0	+0.0		BYTE	Reserved
111.0 to 111.3	+1.0 to +1.3		BOOL	Reserved
111.4	+1.4	GAN_SYN_ST	BOOL	Start gantry synchronization run
111.5 to 111.7	+1.5 to +1.7		BOOL	Reserved
112.0	+2.0		WORD	Reserved
	=4.0		END_STRUCT	
Checkback signals 2				
	114.0		STRUCT	Checkback signals 2
114.0	+0.0		BYTE	Reserved
115.0 to 115.1	+1.0 to +1.1		BOOL	Reserved
115.2	+1.2	GAN_E	BOOL	Gantry trip limit exceeded
115.3	+1.3	GAN_W	BOOL	Gantry limit value for warning exceeded
115.4	+1.4	GAN_SYN_R	BOOL	Gantry synchronization run ready

Table 6-13 User DB "Axis signals", continued

Address		Variable	Data type	Comments
Absolute	Relative			
115.5	+1.5	GAN_SYN_D	BOOL	Gantry grouping is synchronized
115.6	+1.6	GAN_LAX	BOOL	Gantry master axis
115.7	+1.7	GAN_AX	BOOL	Gantry axis
116.0	+2.0	SYNCF	BOOL	Fine synchronism
116.1	+2.1	SYNCG	BOOL	Coarse synchronism
116.2 to 116.7	+2.2 to +2.7		BOOL	Reserved
117.0	3.0		BYTE	Reserved
	=4.0		END_STRUCT	
Read data set				
	20.0		STRUCT	Read data set
20.0	+0.0		WORD	Reserved
22.0 to 22.2	+2.0 to +2.2		BOOL	Reserved
22.3	+2.3	MEA_A	BOOL	Measurement active
22.4	+2.4	FXS_A	BOOL	Travel to fixed stop active
22.5	+2.5	FXS_R	BOOL	Fixed stop reached
22.6 to 22.7	+2.6 to +2.7		BOOL	Reserved
23.0	+3.0	1 INC	BOOL	Increment 1
23.1	+3.1	10 INC	BOOL	Increment 10
23.2	+3.2	100 INC	BOOL	Increment 100
23.3	+3.3	1000 INC	BOOL	Increment 1 000
23.4	+3.4	10000 INC	BOOL	Increment 10 000
23.5 to 23.7	+3.5 to +3.7		BOOL	Reserved
24.0 to 24.5	+4.0 to +4.5		BOOL	Reserved
24.6	+4.6	OS_MOVA	BOOL	Oscillation motion active
24.7	+4.7	OS_A	BOOL	Oscillation active
25.0	+5.0		BYTE	Reserved
26.0	+6.0		DWORD	Reserved
30.0	+10.0		WORD	Reserved
32.0	+12.0		BYTE	Reserved
33.0	+13.0	POS_FENR	BYTE	Positioning axis error number
34.0	+14.0		BYTE	Reserved
35.0	+15.0		BYTE	Reserved
	=16		END_DS	

Table 6-13 User DB "Axis signals", continued

Address		Variable	Data type	Comments
Absolute	Relative			
Write data set				
	40.0		STRUCT	Write data set
40.0	+0.0		BYTE	Reserved
41.0	+1.0		BOOL	Reserved
41.1	+1.1	FXS_RQ	BOOL	Acknowledge fixed stop reached
41.2	+1.2	FXS_SEN	BOOL	Fixed stop sensor
41.3 to 41.7	+1.3 to +1.7		BOOL	Reserved
42.0 to 42.2	+2.0 to +2.2		BOOL	Reserved
42.3	+2.3	KLE_AKT	BOOL	Activate terminals
42.4 to 42.7	+2.4 to +2.7		BOOL	Reserved
43.0	+3.0		BOOL	Reserved
43.1	+3.1	FXS_EN	BOOL	Travel to fixed stop enabled
43.2 to 43.7	+3.2 to +3.7		BOOL	Reserved
44.0	+4.0		DWORD	Reserved
48.0	+8.0		WORD	Reserved
50.0	+10.0	HWE_MINUS	BOOL	Hardware limit switch minus
50.1	+10.1	HWE_PLUS	BOOL	Hardware limit switch plus
50.2	+10.2	SWE_2_MINUS	BOOL	2nd software limit switch minus
50.3	+10.3	SWE_2_PLUS	BOOL	2nd software limit switch plus
50.4	+10.4		BOOL	Reserved
50.5	+10.5	OS_STPR	BOOL	Oscillation, stop at next reversal point
50.6	+10.6	OS_STP	BOOL	Oscillation, Stop
50.7	+10.7		BOOL	Reserved
	=12		END_DS	

6.9.3 Description of signals

Manually triggered signals CANCEL and NC-Restart

CANCEL and NC-Restart are signals that are sent directly to the FM 357 via an OP or the “Parameterize FM 357” tool. With version V3.0 and later, they can be configured for the OP with the “ProTool” configuring tool.

CANCEL under function “Acknowledge NC alarms”
 NC-Restart under function “NC restart”

Control signals

An axis is operated and controlled by means of control signals.

Table 6-14 describes the control signals of user data block “NC signals” and their function.

Table 6-14 Control signals for user DB “NC signals”

Symbol	Name	Function
ST	NC Start	... starts movement in “Automatic, “MDI” modes (see Section 9.10). Starting a motion interrupted by NC Stop. “Edge signal”
STP	NC Stop	... interrupts movement or processing of the program. Continuation of motion with NC Start. “Edge signal”
ESP	Read-in disable	... prevents read-in (processing) of the next block. ... has effect only in “Automatic” mode.
DEL_DIST	Delete the distance to go	... deletes the distance to go
SA	Skip block	... skips identified blocks in the program. ... only effective in “Automatic” mode.
QHF	Acknowledgement Auxiliary function	The signal must be set only to acknowledge receipt of M functions. After acknowledgement, the program can continue execution. “Edge signal”
AUTOMATIC AUTO_E	Mode Automatic/ single block	... if the desired operating mode is selected (see Section 9.10). If the “Automatic single block” function is selected with “AUTO_E” in “Automatic” mode, then the “Automatic single block” mode is activated.
MDI	MDI	Possible only through an operator input in startup with “Parameterize FM 357” tool. Note: To traverse an axis under CPU control, please see Section 6.3, FC 24.
TIPPEN	Jog	If an incremental dimension is selected in “Jog” mode, then the “Incremental travel relative” mode is activated.

Table 6-14 Control signals for user DB “NC signals”, continued

Symbol	Name	Function
REFPKT	Reference-Point Approach	
EILG_KOR_WIR	Rapid traverse override active	... when traversing with rapid traverse, the override set in the “B_OVERR” byte is used.
VOR_KOR_WIR	Feedrate override active	... when traversing with feed, the override set in the “B_OVERR” byte is used.
RES	NC Reset	... triggers a reset <ul style="list-style-type: none"> • Axes are decelerated • Program execution is interrupted (execution starts at beginning of program again) • Deletion of appropriate errors
1INC 10INC 100INC 1000INC 10000INC	Increment 1 Increment 10 Increment 100 Increment 1000 Increment 10000	... selects the incremental dimension (value 1, 10, 100, 1 000, 10 000) If several increments are set at the same time, “Incremental travel relative” mode is cancelled.
KONTIN	Momentary-trigger mode (JOG)	... 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 is switched on.

Table 6-14 Control signals for user DB “NC signals”, continued

Symbol	Name	Function																																																																																																																																				
B_OVERR	Path override	<p>... presets 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” 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 ... 120 % for feed, 0 ... 100 % for rapid traverse</p> <p>An override of 0 % has the same effect as a feed disable.</p> <p>The override must be activated with EILG_KOR_WIR or VOR_KOR_WIR.</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>Setting</th> <th>Code</th> <th>Feed</th> <th>Rapid traverse</th> </tr> </thead> <tbody> <tr> <td></td> <td>Bit 4, 3, 2, 1, 0</td> <td></td> <td></td> </tr> <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>	Setting	Code	Feed	Rapid traverse		Bit 4, 3, 2, 1, 0			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
Setting	Code	Feed	Rapid traverse																																																																																																																																			
	Bit 4, 3, 2, 1, 0																																																																																																																																					
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																																																																																																																																			

Table 6-14 Control signals for user DB “NC signals”, continued

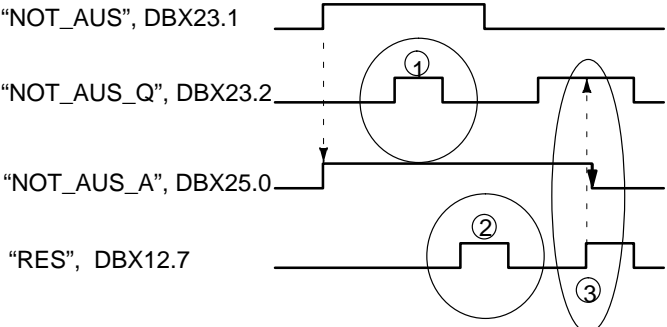
Symbol	Name	Function
NOT_AUS	EMERGENCY STOP	<p>... must be transferred to the FM 357 when the EMERGENCY STOP button is actuated.</p> <p>The FM initiates the following reactive measures:</p> <ul style="list-style-type: none"> • NC program execution is interrupted. • All axes are decelerated within the period set in parameter “Braking time EMERGENCY STOP”. • Interface signal “NC_BEREIT” (NC READY) (user DB “NC signals”, DBX25.4) is reset. • Interface signal “NOT_AUS_A (user DB “NC signals”, DBX25.0) is set. • 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.
NOT_AUS_Q	Acknowledge EMERGENCY STOP	<p>... EMERGENCY STOP status is reset by the sequence of operations shown in the following diagram.</p> <p>Interface signals user DB “NC signals”</p>  <p>① Interface signal “NOT_AUS_Q” has no effect ② Interface signal “RES” has no effect ③ Interface signals “NOT_AUS_Q” and “RES” reset “NOT_AUS_A”</p> <p>Sequence in FM after reset of EMERGENCY STOP status:</p> <ul style="list-style-type: none"> • The controller enable signal is switched through to the drive. • Follow-up mode is cancelled. • Interface signal “LR_A” (user DB “Axis signals”, DBX16.5) is set. • Interface signal “NC_BEREIT” (user DB “NC signals”, DBX25.4) is set. • Interface signal “NOT_AUS_A” (user DB “NC signals”, DBX25.0) is reset. • Error 3 000 “EMERGENCY STOP” is deleted.

Table 6-14 Control signals for user DB “NC signals”, continued

Symbol	Name	Function
SYNA_L1...8	Disable synchronized action ID1...8	... Disable modal or static synchronized action with ID no. 1...8 (see Section 10.22)
SYS_DBW_S	Write system variable	... data word for optional use in synchronized actions (see Section 10.22). Readable in the FM 357 via system variable \$A_DBW[0].

Table 6-15 describes the control signals of user data block “Axis signals” and their function.

Table 6-15 Control signals for user DB “Axis signals”

Symbol	Name	Function
POS_ANFO	Request positioning axis	... Request for axis for positioning and parameter input via the CPU
VER_RPS	Referencing delay	... the signal of the reference point switch of an axis must be transferred to this signal via an input of the user program (UP) (see Section 9.6, Figure 9-13).
DEL_DISTA	Axial deletion of distance to go	... this signal/edge deletes any existing distance to go of the axis
V_STP	Feed STOP	Set: This signal causes an interruption in the relevant axis motion Cancelled: Enable signals for axis motions
R-	Direction minus	... moves axis in negative direction. <ul style="list-style-type: none"> • moves axis in negative direction in “Jog” mode • starts movement in negative direction in “Incremental relative” and “Reference-point approach” modes
R+	Direction plus	... moves axis in positive direction. <ul style="list-style-type: none"> • moves axis in positive direction in “Jog” mode • starts movement in positive direction in Incremental relative and Reference-point approach modes.
SWN_AKT	Activate software cam	... all cam pairs assigned to this axis are activated (see Section 9.9).
RFG	Enable CL controller	... closes the position control loop of the axis. When the controller enable signal is cancelled, the position control loop is opened. The controller enable signal must always be TRUE for axes with a stepper motor. You can set and cancel the servo enable as follows: <ul style="list-style-type: none"> • using this signal (normal condition) • within the control (error condition)
DRUE	Stepper motor rotation monitoring	... can be used on axes with stepper motors without encoders to activate/deactivate the rotation monitoring (see Section 9.5).

Table 6-15 Control signals for user DB “Axis signals”, continued

Symbol	Name	Function
NFB	Follow-up mode	<p>... 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 NFB_A.</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 When the servo enable is canceled, the position setpoint follows the actual value. When the “servo enable” is reactivated, all further axis movements begin at the new actual position. • Follow-up mode = FALSE 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>When the servo enable is active, follow-up mode has no effect.</p>
EILG_UEBERL	Activate rapid traverse override	... when traversing in “Jog” mode and “Incremental relative”, this signal activates rapid traverse.
OVERR_AKT	Activate override	... when traversing, the override set in the OVERR is used.
OVERR	Override	<p>... specifies the override value for this axis.</p> <p>The specifications for the feed apply.</p> <p>The override must be activated with OVERR_AKT.</p>
GAN_SYN_ST	Start gantry synchronization run	<p>... interface signal for master axis in gantry grouping</p> <p>The FM 357 starts the synchronization process (see Section 9.13.2).</p>

Checkback signals

The checkback signals indicate the signal status of the axis and return it to the user DBs.

Table 6-16 describes the checkback signals of user data block “NC signals” and their function.

Table 6-16 Checkback signals for user DB “NC signals”

Symbol	Name	Function
PROGL PROGW PROG_ANGEH PROG_UNTB PROG_ABGB	Program running Program is waiting Program is stopped Program is interrupted Program is aborted	... shows the status of the program (see Section 10)
AHF	Change auxiliary function	... indicates that at least one auxiliary function was output The “MNR” byte must be evaluated. Output of: <ul style="list-style-type: none"> • an M function: Function number is stored in variable “MNR” (bits 0 to 6) • several M functions: The function numbers are stored in variables “MNR_1” to “MNR_5” The signal is reset in the next CPU cycle.
RES_Q	Acknowledge NC Reset	... acknowledges the triggered Reset.
AUTOMATIK_A MDI_A TIPPEN_A REFPKT_A		... indicates which operating mode is active.
MNR	M function number	... evaluation of M function output: Highest-order bit = FALSE (output of M functions) An M function was output. The M function number is stored in this BYTE (bits 0 to 6). Note: Highest-order bit = TRUE (output of M, H and T functions) More than one M function and/or at least one H function or T function was output in the NC block. Entry of M/H/T functions (see Table 6-22): <ul style="list-style-type: none"> • “MNR_1”...“MNR_5” • “HNR_1”/“HWERT_1”...“HNR_3”/“HWERT_3” • “TNR” The M function number is deleted again in the next CPU cycle. For example, see Figure 6-8

Table 6-16 Checkback signals for user DB “NC signals”, continued

Symbol	Name	Function
NOT_AUS_A	EMERGENCY STOP active	... FM 357 signal in EMERGENCY STOP status (see NOT_AUS and NOT_AUS_Q signals)
NC_BEREIT	NC ready signal	... the controller is ready for operation. This signal represents the relay contact “NC-READY”. The signal is activated if <ul style="list-style-type: none"> • the “NC-READY” relay contact is closed • all internal controller power circuits have been set up • the controller is in cyclical mode
NC_FE	NC error active	... an error with error number is active (see Section 11.2).
NC_BATFE	NC battery error active	... battery has no contact or must be replaced.
AX_REF	All axes referenced	... specifies that all axes which need to be referenced have been referenced.
AX_STEHEN	All axes stationary	... indicates that all axes of the FM 357 are at a standstill.
NC_FEOB	Error without interruption of machining	... at least one error without machining stop is active (see Section 11.2).
NC_FEMB	Error with interruption of machining	... at least one error with machining stop is active (see Section 11.2).
SYNA_LA1...8	Synchronized action ID1 ... 8 disabled	... FM 357 checkback signal, requested disable (SYNA_L1...8) is active
SYS_DBW_L	Read system variable	... data word for optional use of synchronized actions (see Section 10.22). Writeable in the FM 357 via system variable \$A_DBW[1].

Table 6-17 describes the checkback signals of user data block “Axis signals” and their function.

Table 6-17 Checkback signals for user DB “Axis signals”

Symbol	Name	Function
SYN	Synchronized	... the axis is referenced/synchronized
PEHG	Position reached, stop (target range coarse)	... the axis is located within the target range coarse or fine (see Section 9.5).
PEHF	Position reached, stop (target range fine)	
SWN_A	Software cams active	... indicates that all cam pairs assigned to an axis and the cam signal generation are active (see Section 9.9)
POS_AX	Axis is a positioning axis of the CPU	... the axis is assigned to the CPU (positioning). Acknowledgement signal of POS_ANFO

Table 6-17 Checkback signals for user DB "Axis signals", continued

Symbol	Name	Function
FR-	Go_minus	... means the axis is traveling in the direction of decreasing actual values or in the direction of voltage output "-" in OL control mode.
FR+	Go_plus	... means the axis is traveling in the direction of increasing actual values or in the direction of voltage output "+" in OL control mode. As soon as an active traversing movement is pending, the messages (FR+) or (FR-) are output depending on the traversing direction. They can only be pending as alternatives.
NFB_A	Follow-up mode active	... the follow-up mode is active in this axis.
STEHT	Axis stationary	... the axis has a velocity that is smaller than the value in the parameter "threshold velocity for stationary axis".
LR_A	Position controller active	... indicates whether the position controller or speed controller is active.
DR_A	Speed controller active	(see Section 9.3)
DRUE_FE	Error in stepper motor rotation monitoring	... the rotation monitoring circuit for this axis has detected an error.
GAN_E	Gantry trip limit exceeded	... indicates whether the value defined in parameter "Trip limit" has been exceeded (see Section 9.13.2)
GAN_W	Gantry limit value for warning exceeded	... indicates whether the value defined in parameter "Limit value for warning" has been exceeded (see Section 9.13.2)
GAN_SYN_R	Gantry synchronization ready to start	... the synchronization run can be started with interface signal "GAN_SYN_ST" (user DB "Axis signals", DBX111.4) (see Section 9.13.2)
GAN_SYN_D	Gantry grouping is synchronized	... indicates that the synchronization run is finished (see Section 9.13.2)
GAN_LAX	Gantry master axis	... axis is the master axis in the gantry grouping (see Section 9.13.2)
GAN_AX	Gantry axis	... axis is master or synchronized (slave) axis (see Section 9.13.2)
SYNCF	Fine synchronism	... status of master-value link (see Section 9.13.3)
SYNCG	Coarse synchronism	... status of master-value link (see Section 9.13.3)

Read data block

Table 6-18 describes the “Read data block” signals of user data block “NC signals” and their function.

Table 6-18 Read data set for user DB “NC signals”

Symbol	Name	Function
TASTER_1 TASTER_2	Probe 1 activated Probe 2 activated	... with the “measure” function, these signals indicate whether probe 1 or 2 is active.
SW_NO0_MINUS to SW_NO7_MINUS	Software cam minus	... displays the status of software cams 0...7 minus.
SW_NO0_PLUS to SW_NO7_PLUS	Software cam plus	... displays the status of software cams 0...7 plus.
DIG_EIN9...24 DIG_AUS9...24	Status digital inputs/out-puts on the local P-bus	... indicates the status of digital I/Os 9...24 on the local P bus
M00/M01_A	M00/M01 active	... M functions are active, for meaning see Section 10.14
M02/M30_A	M02/M30 active	

Table 6-19 describes the signals for “Read data set” of the “Axis signals” user data block, and their function.

Table 6-19 Read data set for user DB “Axis signals”

Symbol	Name	Function
MEA_A	Measurement active	... “Measurement” function is active.
FXS_A	Travel to fixed stop active	... “Travel to fixed stop” function is active.
FXS_R	Fixed stop reached	... Fixed stop has been reached
1INC 10INC 100INC 1000INC 10000INC	Increment 1 Increment 10 Increment 100 Increment 1 000 Increment 10 000	... indicates which incremental dimension is selected in “Incremental Relative” mode.
OS_MOVA	Oscillation motion active	... oscillation axis is in motion
OS_A	Oscillation active	... axis is oscillation axis
POS_FENR	Positioning axis error number	see Table 6-5, FC 24 troubleshooting

Write data block

Table 6-20 describes the “Write data block” signals of user data block “NC signals” and their function.

Table 6-20 Write data set for user DB “NC signals”

Symbol	Name	Function
SP_DIG_AUS	Disable digital outputs 9...15 and 16...24 on local P bus	... can be used to define a disable for output 9...24. The output is then set to FALSE and can no longer be influenced.
M01	Activate M01	... activates the “M01” function in “Automatic” mode.
VSP	Activate feed disable	... causes the axes of an FM 357 to stop (Feed Stop).
NC_STSP	Activate NC start disable	... inhibits starting of an NC program.
STP_SG	Activate NC stop at block boundary	... causes an NC program to stop at the block limit.

Table 6-21 describes the signals for “Write data set” of the “Axis signals” user data block, and their function.

Table 6-21 Write data set for user DB “Axis signals”

Symbol	Name	Function
FXS_RQ	Acknowledge fixed stop reached	... relevant only if parameter “Acknowledgement signal” is set (see Section 9.15). A block change can be executed. 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.
FXS_SEN	Fixed stop sensor	... relevant only if parameter “Fixed stop detection” is parameterized with “external sensor” (see Section 9.15).
KLE_AKT	Activate terminals	... activates terminal monitoring
FXS_EN	Travel to fixed stop enabled	... relevant only if parameter “Acknowledgement signal” is set (see Section 9.15). If the signal is reset before the fixed stop has been reached, an error message is output and the function aborted.
HWE_MINUS HWE_PLUS	Hardware limit switch minus Hardware limit switch 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 9.5).

Table 6-21 Write data set for user DB "Axis signals", continued

Symbol	Name	Function
SWE_2_MINUS	2nd software limit switch minus	... this signal can be used to activate the 2nd software limit switch minus or plus (see Section 9.5).
SWE_2_PLUS	2nd software limit switch plus	The 1st software limit switch is then no longer active.
OS_STPR	Oscillation, stop at next reversal point	... oscillation axis stops at reversal point
OS_STP	Oscillation, Stop	... oscillation axis entered

Auxiliary functions

Table 6-22 describes the "Auxiliary functions" signals of the user data block "NC signals" and their functions.

Table 6-22 Auxiliary functions for user DB "NC signals"

Symbol	Name	Function
MNR_1...MNR_5	M function number 1...5	... the values are entered according to "MNR"
HNR_1...HNR_3	H function number 1...3	
HWERT_1...HWERT_3	H function value 1...3 (REAL)	
TNR	T function number	

Auxiliary functions are updated by the "Change auxiliary function" signal (AHF, user DB "NC signals", DBX15.5).

Example of auxiliary function

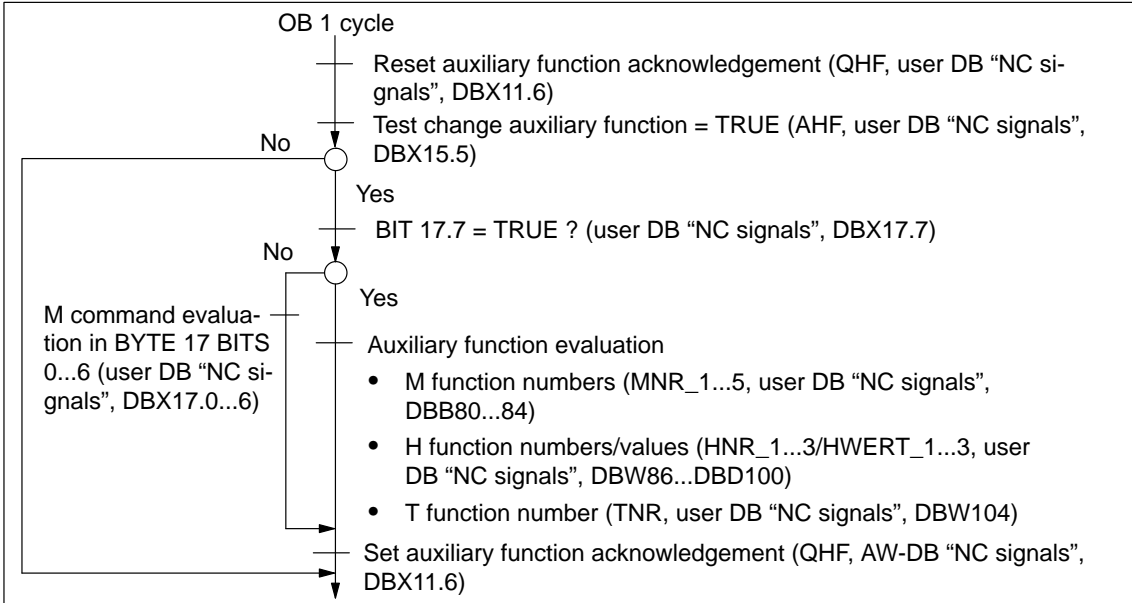


Figure 6-8 Example of auxiliary function

6.10 User handling procedures for controlling axes

The following table lists the relevant signals for controlling axes.

Table 6-23 User handling procedures for controlling axes

Motion is activated via ...	For axis type, see Section 10.2.2	Relevant control and checkback signals from ...	
		User DB "NC signals"	User DB "Axis signals"
NC program in "Automatic" mode activated via FB 4 or OP	Path axes see Section 10.5	<ul style="list-style-type: none"> • Motion control: ST, DBX11.0 STP, DBX11.1 VSP, DBX68.0 • Override: EILG_KOR_WIR, DBX12.5 VOR_KOR_WIR, DBX12.6 B_OVERR, DBX21.0 • General: RES, DBX12.7 NOT_AUS, DBX23.1 NOT_AUS_Q, DBX23.2 (after NOT_AUS) • Checkback signals: SYST_BEREIT, DBX7.0 NC_BEREIT, DBX25.4 NC_FEMB, DBX26.7 AX_REF, DBX26.2 AUTOMATIK_A, DBX16.0 PROGL, DBX15.0¹⁾ AX_STEHEN, DBX26.3¹⁾ (for error messages, see also Section 11.2)	<ul style="list-style-type: none"> • RFG, DBX12.1 • HWE_MINUS, DBX50.0 • HWE_PLUS, DBX50.1 • Checkback signals: FR-, DBX15.6¹⁾ FR+, DBX15.7¹⁾
	For positioning axes, see Section 10.5.5	<ul style="list-style-type: none"> • see above, except for override 	<ul style="list-style-type: none"> • see above • V_STP, DBX11.3 • OVERR_AKT, DBX12.7 • OVERR, DBB13
	Positioning axis (oscillation axis)	<ul style="list-style-type: none"> • see above 	<ul style="list-style-type: none"> • see above "Positioning axes" • OS_STP, DBX50.6, • OS_STPR, DBX50.5 • OS_MOVA, DBX24.6 • OS_A, DBX24.7

1) Conditional, acc. to application

Table 6-23 User handling procedures for controlling axes, continued

Motion is activated via ...	For axis type, see Section 10.2.2	Relevant control and checkback signals from ...	
		User DB "NC signals"	User DB "Axis signals"
"Jog" or "Incremental travel relative" and "Reference point approach"	as for positioning axis	<ul style="list-style-type: none"> • General: RES, DBX12.7 NOT_AUS, DBX23.1 NOT_AUS_Q, DBX23.2 (after NOT_AUS) • STP, DBX11.1 • Checkback signals: SYST_BEREIT, DBX7.0 NC_BEREIT, DBX25.4 NC_FEMB, DBX26.7 • Mode checkback signals: TIPPEN_A, DBX16.2 or REFPKT_A, DBX16.3 <p>(for error messages, see also Section 11.2)</p>	<ul style="list-style-type: none"> • RFG, DBX12.1 • V_STP, DBX11.3 • OVERR_AKT, DBX12.7 • OVERR, DBB13 • R-, DBX11.6 • R+, DBX11.7 • VER_RPS, DBX11.1 (in "Reference point approach" mode) • HWE_MINUS, DBX50.0 • HWE_PLUS, DBX50.1 • Checkback signals: FR-, DBX15.6 FR+, DBX15.7 PEHG, DBX15.1¹⁾ PEHF, DBX15.2¹⁾ SYN, DBX15.0¹⁾, according to mode
FC 24 in "Jog" or "Automatic" mode	Positioning axis (CPU axis)	<ul style="list-style-type: none"> • General: RES, DBX12.7 NOT_AUS, DBX23.1 NOT_AUS_Q, DBX23.2 (after NOT_AUS) • STP, DBX11.1 • Checkback signals: SYST_BEREIT, DBX7.0 NC_BEREIT, DBX25.4 NC_FEMB, DBX26.7 • Mode checkback signal: TIPPEN_A, DBX16.2 or REFPKT_A, DBX16.3 <p>(for error messages, see also Section 11.2)</p>	<ul style="list-style-type: none"> • POS_ANFO, DBX1.0¹⁾ • RFG, DBX12.1 • V_STP, DBX11.3 • OVERR_AKT, DBX12.7 • OVERR, DBB13 • POS_AX, DBX15.5¹⁾ • POS_FENR, DBB33 • HWE_MINUS, DBX50.0 • HWE_PLUS, DBX50.1 • FC 24 signals • Checkback signals: FR-, DBX15.6 FR+, DBX15.7 <p>(for error messages, see also Section 11.2)</p>

1) Conditional, acc. to application

6.11 Application examples

General

After installation of the FM 357 configuring package, example project **FM357_EX** is installed in **[STEP7 directory]\EXAMPLES**. The contents of the project are programming examples for “Read data” directly addressed (FB 2), “Read data” indirectly addressed (FB 2) and “Positioning” (FC 24).

After the FM357 has started (bits in user DB “NC signals”, SYST_BEREIT = TRUE and ANLAUF = FALSE), the interface can be supplied with data (see OB 1 in each example).

To traverse an axis in “Jog” mode, the following bits need to be set:

- Operating mode: User DB “NC signals”, DBX12.2 (TIPPEN (JOG)) = TRUE (all other mode bits = FALSE)
- Feed stop: User DB “Axis signals”, DBX11.3 (V_STP) = FALSE
- Controller enable: User DB “Axis signals”, DBX12.1 (RFG) = TRUE
- Direction specification: User DB “Axis signals”, DBX11.6 (R-) **or** DBX11.7 (R+) = TRUE

The OB 1 is structured in the same way as the supplied FM357OBNx-AWL source from library FM357_LI. The example calls have been inserted at the USER program marker in each OB 1.

Note

- The user program is not executed if the start-up tool is operating in TEST mode.
 - Override default setting 100 %
 - The signals of the hardware limit switches have been ignored in the example.
 - In the initial state, the parameters to be read may be set to “zero”.
-

Example 1: “Read data” directly addressed (FB 2)

See STEP 7 project FM357_EX\EXAMPLE1

To execute the example, the following blocks are needed in addition to basic functions:

- DB 120
- DB 121
- OB 1
- OB 82
- OB 100

After the FM 357 has started successfully, “JOG” mode is set. After marker M 35.0 has been set, input parameter Req of FB 2 is activated and the following FM 357 variables read:

- First R parameter (R0)
- Actual value of first axis
- Error number

Output bits M 37.1 (Error) and M 37.2 (NDR – new data received) display the states of function block FB 2. The data to be read do not become valid until bit M 37.2 is set to TRUE (NDR).

If error bit M 37.1 = TRUE, output word State should be evaluated because a function block execution error has occurred and the error number stored in MW 38 (State).

The read variables are stored in output double words RD1 (MD 40), RD2 (MD 44) and RD3 (MD 48). The S7 type for storage of the parameters is taken from the NC-VAR selector.

Note

When the R parameter and actual value are selected from the NC-VAR selector, the value to be read must be entered in Line when a number is assigned:

- Enter the following for the axis actual value: Line = axis number; area no. = 1
 - Enter the following for R parameter: Line = R number + 1; area no. = 1
-

Example 2: “Read data” indirectly addressed (FB 2)

See STEP 7 project FM357_EX\EXAMPLE2

To execute the example, the following blocks are needed in addition to basic functions:

- DB 120
- DB 121
- OB 1
- OB 82
- OB 100

Example 2 illustrates indirect addressing of R parameters.

After the FM 357 has started successfully, “JOG” mode is set. Before marker M 35.0, which activates input parameter Req of FB 2, is set, a value should be assigned to parameter Line1 (MW50). The R parameter corresponding to the setting in parameter Line1 (MW 50) is read.

Output bits M 37.1 (Error) and M 37.2 (NDR – new data received) display the states of function block FB 2. The data to be read do not become valid until bit M 37.2 is set to TRUE (NDR).

If error bit M 37.1 = TRUE, output word State should be evaluated because a function block execution error has occurred and the error number stored in MW 38 (State).

The read value of the R parameters is stored in output double word RD1 (MD 40). The S7 type for storage of the parameter has been taken from the NC-VAR selector.

Note

Addressing variables:

To address the variables of R parameters, for example, the value “zero” must be entered as the line number in the NC-VAR selector after selection of the variable. The contents of the line specified by the NC-VAR selector are checked for “zero” in FB2 or FB3. If the line contains “zero”, the setting of input parameter Line1 (in example: MW 50) is accepted, i.e. the Line1 parameter is supplied with the desired variable before FB 2 is called by the user (see Section 6.4, Addressing an NC variable).

Example 3: Positioning (FC 24)

See STEP 7 project FM357_EX\EXAMPLE3

To execute the example, the following blocks are needed in addition to basic functions:

- OB 1
- OB 82
- OB 100

Example 3 contains an example of how to program the positioning of an axis using block FC 24.

After the FM 357 has started successfully, "JOG" mode is set. After marker M36.0 (Start) has been set, the selected axis number (Parameter AxisNo) is traversed depending on the settings in parameters Pos and FRate.

Output signals InPos (M 36.1), Activ (M 36.2), StartErr (M 36.3) and Error (M 36.4) indicate the various axis states in conjunction with FC 24. If an error occurs in the execution of FC 24, then data byte 33 (POS_FENR) must be evaluated in the relevant user DB "Axis signals" (see Section 6.3, FC 24).

6.12 Technical data

Memory allocation

The memory required to operate the FM 357 is specified below for a minimum and a maximum configuration.

Table 6-24 Configuration

Function Block	Function	Comment	Block in bytes of work memory
e.g. minimum configuration of an FM 357 (2 axes)			
FB 1, 18 FC 1, 2, 5, 12, 22, 23, 28 DB 1, 5, 7 OB 1, 82, 100	Standard functions Data transfer Startup	Startup CPU cycle Diagnostic interrupt Operating modes Read data Write data Parameter definition	} approx. 10 370
DB 21, 31, 32	User DBs		
		Total	approx. 11 250
Maximum configuration of an FM 357 (4 axes)			
FB 1, 18 FC 1, 2, 5, 12, 22, 23, 28 DB 1, 5, 7 OB 1, 82, 100	Standard functions Data transfer Startup	Startup CPU cycle Diagnostic interrupt Operating modes Read data Write data Parameter definition	} approx. 10 370
DB 21, 31, 32, 33, 34	User DBs		
FC 24	Positioning	Axis positioning from CPU	approx. 620
FC 9	ASUP	Asynchronous subroutine	approx. 280
FB 2, 3, 4, 6 DB 15, 16	CPU/FM communication	Read variable Write variable Select program	approx. 6 000
		Total	approx. 18 390

In the case of a **maximum configuration with three FM 357 modules and four axes**, approx. 2 240 BYTE of user DBs (DB 22, DB 36...39, DB 23, DB 41...44) must be added.

Timers

Timers 0 to 4 are assigned or reserved internally for standard function blocks.

Processing times

FC 22, FM 357 in central configuration:

Basic functions only (I/O signals)	Approx. 4...6 ms
Including write or read data block	Approx. 11...13 ms
Including write and read data block	Approx. 16...21 ms

FC 22 in CPU cycle, FM 357 in distributed configuration:

With the following plant configuration:

- CPU 315-2 and SM 321/322 in a central rack
- IM 153-2 and an FM 357 in a distributed configuration

Basic functions only (I/O signals)	Approx. 10...14 ms
Including write or read data block	Approx. 13...17 ms
Including write and read data block	Approx. 13...22 ms

Distributed data block transmission times (FC 22, several CPU cycles):

With the following plant configuration:

- CPU 315-2 and SM 321/322 in a central rack
- IM 153-2 and an FM 357 in a distributed configuration

Write data block	Approx. 24...40 ms
Read data block	Approx. 22...37 ms



Starting Up the FM 357

7

General

This Section 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 start-up of the machine axes.

Section overview

Section	Title	Page
7.1	Installation and wiring	7-2
7.2	Starting up the FM 357	7-3
7.3	Procedure for parameterization	7-4
7.4	Testing and optimization	7-6

7.1 Installation and wiring

Information about installation

For information about installing the module, please refer to

- Section 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

- Section 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 positioning 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 positioning 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 positioning module setup is consistent with the parameterization. Check in particular that: <ul style="list-style-type: none"> • The attached encoder matches the machine data. • The wiring of the digital I/O modules matches the machine data. 	

7.2 FM 357 power-up

Important operating and display elements for power-up

The following operating and display elements are important with respect to FM 357 power-up:

- Error and status LEDs
- Start-up switch on FM 357 and CPU

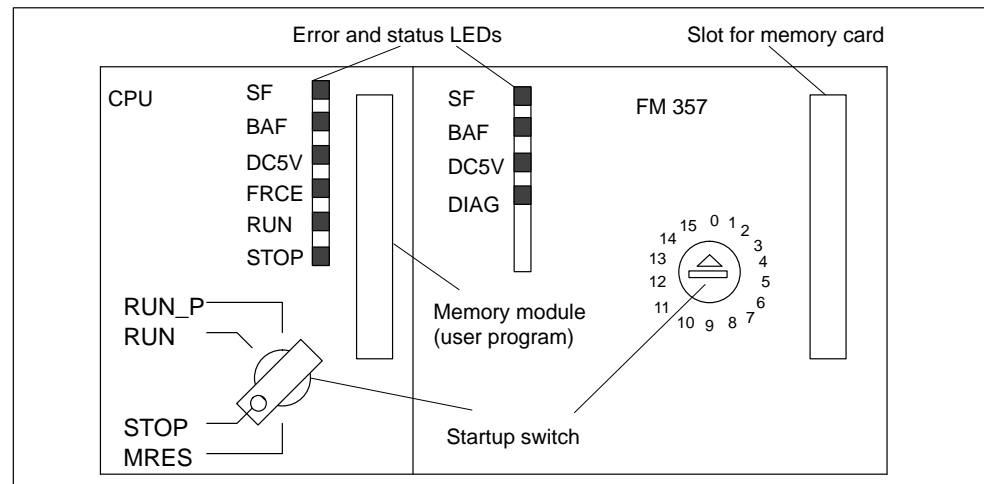


Figure 7-1 Operating and display elements for power-up

Startup switch

The FM 357 is equipped with a start-up switch (see Fig. 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.

Table 7-2 Settings with the start-up switch of the FM 357

Setting	Significance
0	Normal start-up from FLASH
1	Power-up from FLASH and start-up with default values
2	Start-up from FLASH and transition to "update mode" over the MPI interface
4	Reserved
5	Reserved
6	Starting from memory card and switchover to "Update mode" of all FM 357 software on the memory card

Power-up times

Power-up from FLASH state (start-up switch = 0) takes approximately 65 s.

Power-up from the memory card takes approximately 150 s.

Status displays (on LEDs) during power-up

Power-up status is displayed by the following indicators:

- CPU:
 - DC 5V (green) → ON
 - RUN (green) → Flashes
 - STOP (yellow) → ON
 - Further LEDs → OFF
- End of power-up:
 - RUN (green) → ON
 - STOP (yellow) → OFF
- FM 357:
 - DC 5V (green) → ON
 - Further LEDs → OFF
- End of power-up: DIAG (yellow) → Flashes (sign-of-life approx. 8 Hz)

7.3 Procedure for parameterization

Information about parameterization

For information about parameterization, please refer to

- Sections 5 and 9 in this manual
- The on-line help in "Parameterize FM 357"

Overview

The parameterization data of the FM 357 comprise:

- Machine data → for starting up the module
- User data → for start-up and adaptation of NC programs

The machine data can be processed online or offline by the parameterization tool and transferred to the module over the MPI. The parameterization tool verifies the data entered against the permissible limit values.

After transfer to the module, the machine data are stored there modally.

Checklist

Notwithstanding the input check mentioned above, the module user is responsible for ensuring that all machine data are set correctly. 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</p> <p>For power-up with default values see Section 7.2</p> <p>The default values of the machine data are stored in Table 5-1.</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 • No. of axes • 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.</p> <p>In Section 9.1 you will find further information about the individual machine data.</p>	
3	Axes	<p>Basic startup 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 9.5).</p> <p>After loading and activation, you can initiate the test and optimization procedure described in Section 7.4.</p>	
4	Functions	<p>Parameterizing the NC functions</p> <p>The following NC 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>For further information, please refer to Sections 9.6 to 9.9, 9.13, 9.15</p>	

7.4 Testing and optimization

Information about testing and optimization

After you have installed, wired up, powered up and parameterized the FM 357 module, you can test and optimize it using the test and startup interface with user program (UP). The UP is stored ready for use in S7 library FM357_LI (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 module can be controlled from the startup interface if parameter [TEST_ST] has been set.

This interface is installed with “FM 357 Parameterization”. Once the FM 357 has been parameterized, you can call it up by selecting the menu **Test ▶ Startup**.

When you call up this menu the following screen appears:

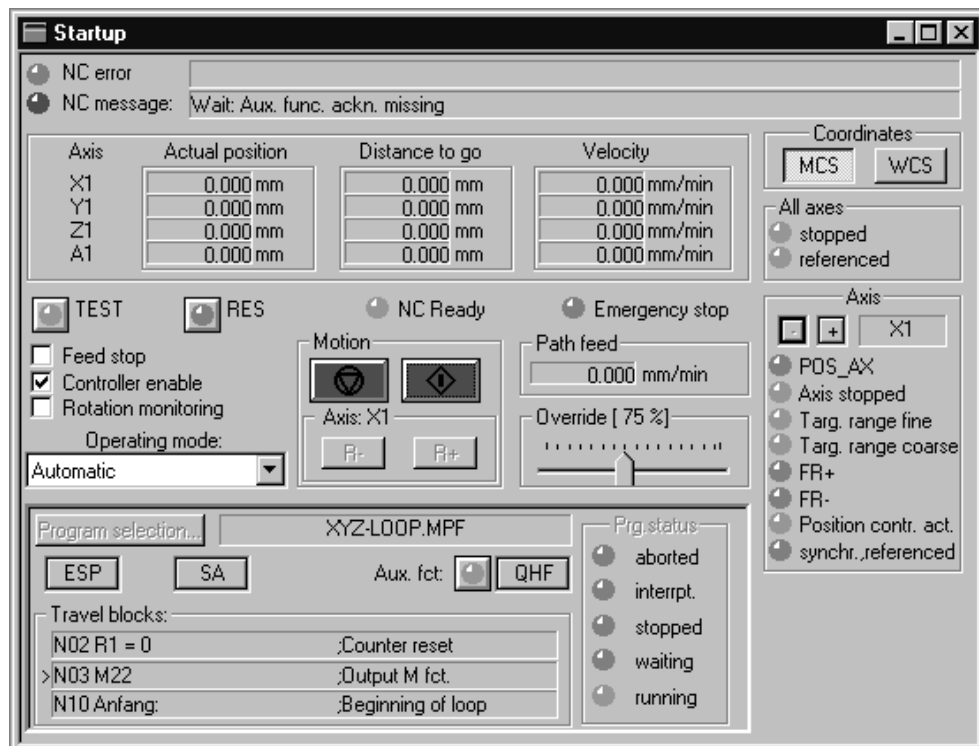


Figure 7-2 Startup interface (e.g. for “Automatic” mode)

You can also call up the following screens:

By selecting menu items **Test ▶ Troubleshooting**, you can call the following display:

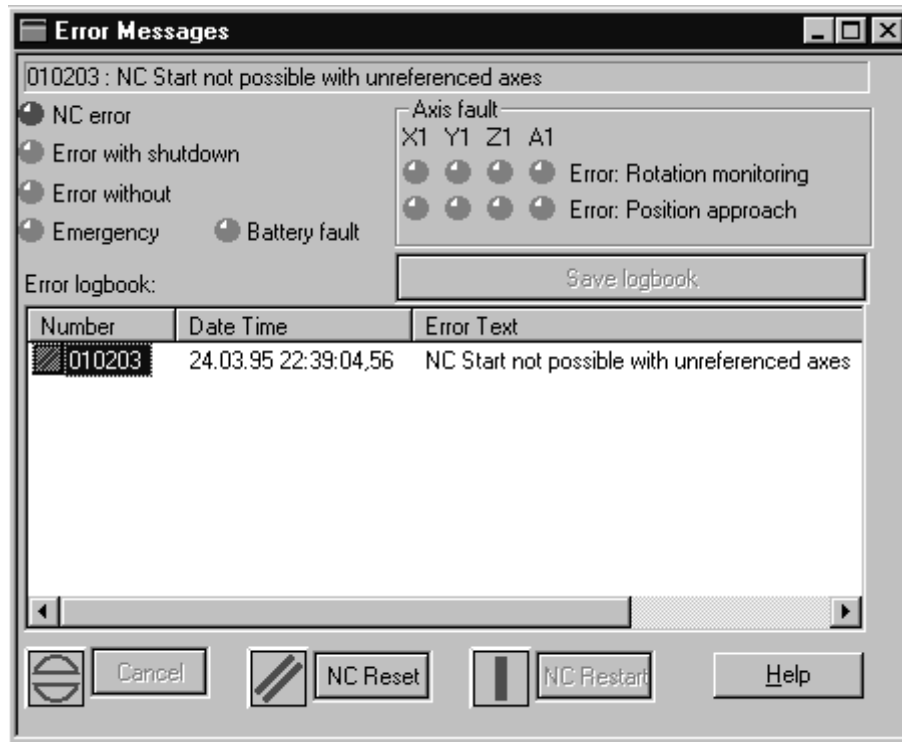


Figure 7-3 Troubleshooting

By selecting menu items **Test ▶ Service data**, you call the following display:

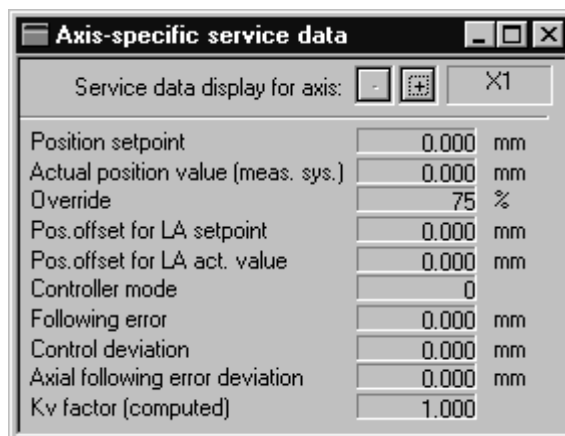


Figure 7-4 Servicing data

By calling menu items **Test ▶ Trace** , you can call the following display:

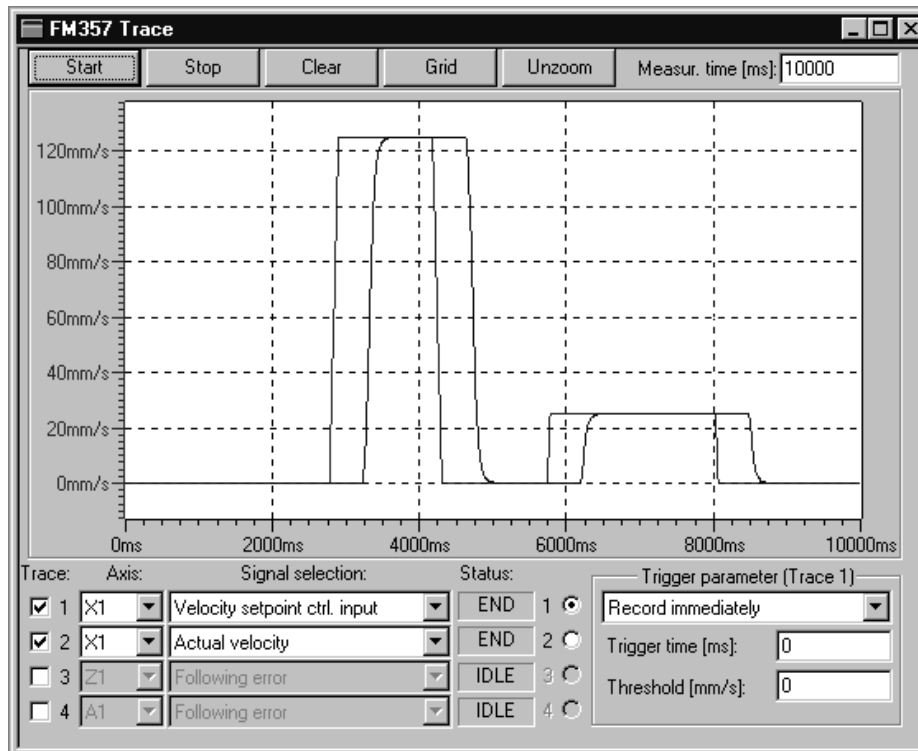


Figure 7-5 Trace

You have the possibility of recording up to four signal curves in this screen.

The following signals can be recorded:

- Following error
- Control deviation
- Contour deviation
- Actual position (incl. offset)
- Position setpoint
- Position setpoint at controller input
- Velocity setpoint at controller input
- Acceleration setpoint at controller input (not available yet)
- Actual velocity
- "Interpolation ended" signal
- "Fine target range" signal
- "Coarse target range" signal
- Accumulated actual position without offset
- Accumulated position setpoint without offset

A trigger parameter can be set for each signal curve:

- No trigger
- Record immediately
- Positive edge
- Negative edge
- Trace 1
- IPO event (see Table 10-5)

To synchronize several signal curves simultaneously, the curves can be triggered in response to signal curve 1 (trace 1).

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 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 signals must be routed over the CPU interface for the axis:

Axis 1...4	(User DB “NC signals”, DBX3.0...3.3)
Controller enable	(User DB “Axis signals”, DBX12.1)

The following signals on the interface should **not** be enabled, since these cause the movement to be disabled:

Override	(User DB “Axis signals”, DBX13.0) not at 0 %
Delete residual distance	(User DB “NC signals”, DBX11.4)
Feed Stop	(User DB “Axis signals”, DBX11.3)

Limit switches

Setting of hardware limit switches and signal check:

- Hardware limit switch plus (User DB “Axis signals”, DBX50.1)
- Hardware limit switch minus (User DB “Axis signals”, DBX50.0)

Test sequence

To test the axis, please follow the sequence specified in the flowchart overleaf:

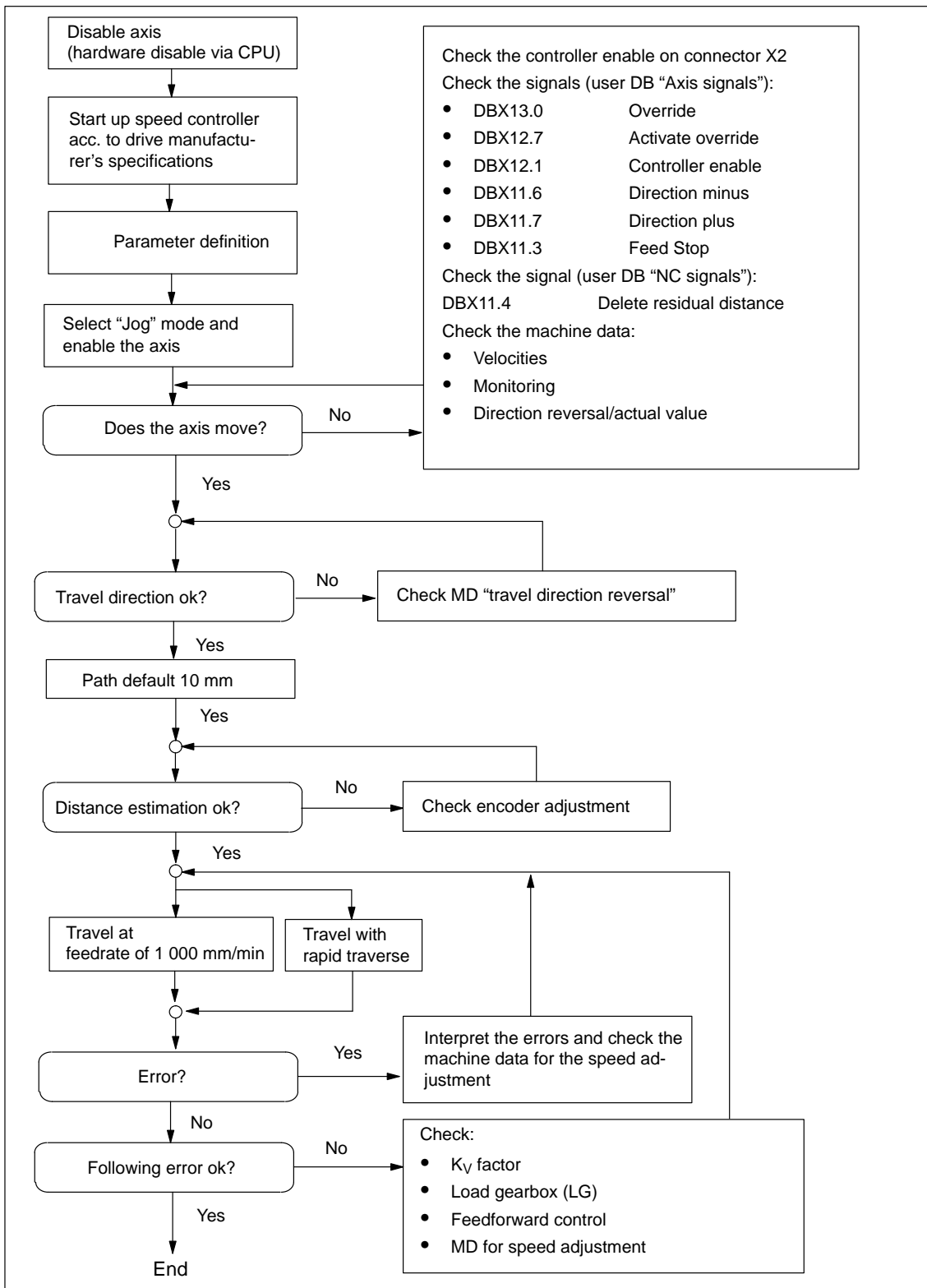


Figure 7-6 Axis testing



Human-Machine Interface

Overview

This Section provides an overview of the human-machine communication options between the operator and the FM 357 module.

To provide a human-machine interface between the operator and FM 357, an operator interface can be connected to the CPU via the MPI interface (see Figure 1-2).

The module uses the SIMATIC interface (backplane bus) to communicate with the control panel.

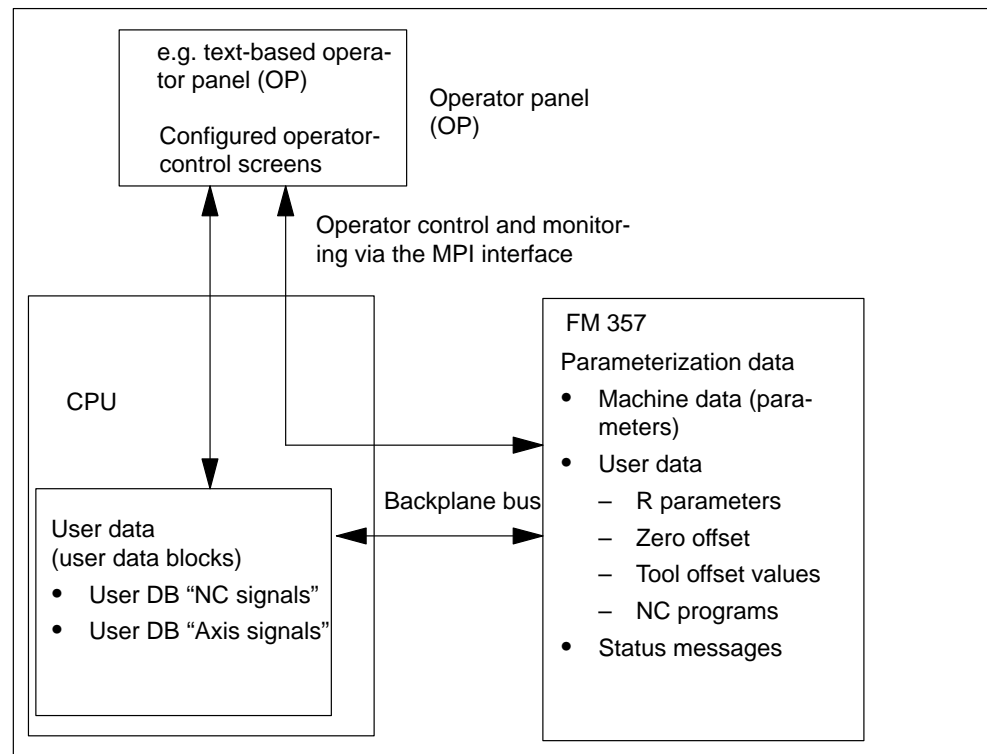


Figure 8-1 Operator control and monitoring for the FM 357

Controlling and monitoring FM data and signals in the CPU

Data and signals which can be controlled and monitored are listed in the user data blocks. These data/signals are processed by the OP or the user program.

Which elements on the FM 357 can be controlled by the operator?

The data/signals in the user DBs can be modified or extended via the keyboard of the operator panel:

- User DB “NC signals” (DB 21)
 - e.g.
 - Axis selection
 - NC Start, NC Stop
 - Operating modes
- User DB “Axis signals” (DB 31 through DB 34)
 - e.g. Override

You can activate functions of the FM 357:

e.g.

- NC program overview
- NC program selection
- Tool offset overview

Which elements on the FM 357 can be monitored?

The following data/signals can be displayed on the operator panel screen:

- Machine data
- User data
 - R parameters
 - Zero offsets
 - Tool offset data
 - NC programs
- Status messages from the user DBs or from variables on the FM 357 including
 - Operating data, e.g. actual values
 - Active NC blocks
 - Checkback signals and error conditions
 - Servicing data

The configuring package includes a preconfigured user interface for COROS OP 17 units.

8.1 Standard HMI for OP 17

Overview

This Section describes a preconfigured interface that you will need to adapt to suit your own project (e.g. FM addresses, DB nos.) for the COROS device (operator panel): OP 17

The interface must be adapted with the “ProTool/Lite or ProTool” configuring tool, version V3.0 or later. You can use it to modify, add or delete screens.

The user interface addresses:

- the user DBs “NC signals” (DB 21) and “Axis signals” (DB 31...34) in the CPU (controller: control_CPU; address = 2; slot = 0)
- the variables of the FM 357 (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 entire configuration using “ProTool/Lite” V3.0. This provides you with detailed screen descriptions.

You will find the preconfigured user interface in the following directory:

SIEMENS\STEP7\EXAMPLES\S7OP_BSP\01737_1a.pdb

Monitoring

The data to be monitored can be read and displayed directly from the FM 357 or via the user DBs of the CPU.

Operator control

Data and signals (including markers (bits) and values) are written to the user DBs of the CPU for manipulation by the operator.

User program

The OP can be used, for example, to set markers which are evaluated by the user program (e.g. select “MDI” mode on the FM 357).

OP 17 operator interface

The following diagram shows you an overview of the operator interface (menu tree) in the sample configuration of the OP 17 for the FM 357.

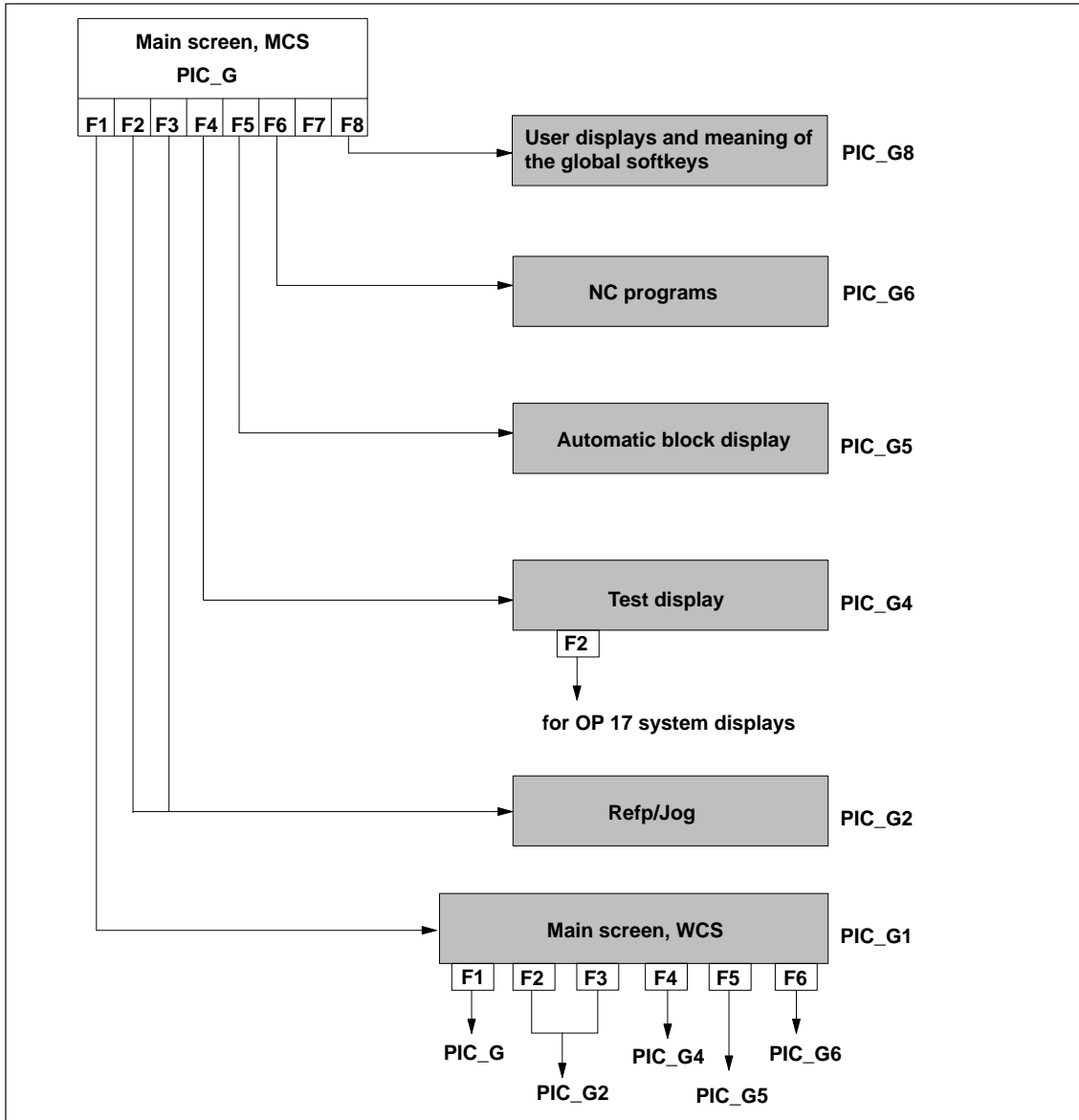


Figure 8-2 Menu tree of operator interface on OP 17

Please refer to the sample configuration in the PIC_G8 diagram for the assignment of the global softkeys.

Note

If not all four axes are used, the corresponding softkeys (K1 to K4) in the sample project should be cleared.

Main screen

The contents of individual displays can be found in the sample configuration.

For example, the following illustration shows the screen layout of PIC_G “Main screen MCS”.

FM357	Main screen, 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}			
Alarm: {VAR_210}					
WCS	Refp	Jog	Res	Auto	P Sel User

Figure 8-3 Main screen MCS PIC_G

The sample configuration is intended as a starting aid for your project. Copy the file 01737_1a.pdb. You can edit the copy to suit your application.

Note

If you do not adapt the sample configuration to your application project (e.g. 3 axes only), then error message “Variables of 4th axis do not exist” is output.

Selection of operator control 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 Section 6)
Target system = Steuerg_CPU
- Symbol list of the NC-VAR Selector
Target system = Steuerg_357

Symbol list

The currently valid symbol list is included in the sample configuration and displayed when FM 357 variables are selected. If you wish to use a new symbol list (e.g. modifications or additions to variables), then you can copy variables from the NC-VAR selector and link them into your project.

The NC-VAR selector is included in the configuring package of the FM 357. The installation procedure is described in Section 6.

You will find the symbol list in the following directory:

ProTool: Target system ► Control ► Editing ► Parameters ► Symbol list

8.2 Troubleshooting on OP 17 (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 with the “ProTool/Lite or ProTool” configuring tool, V3.0 and later on the OP.

Note

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 menu items **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 following **Input/output** dialog.

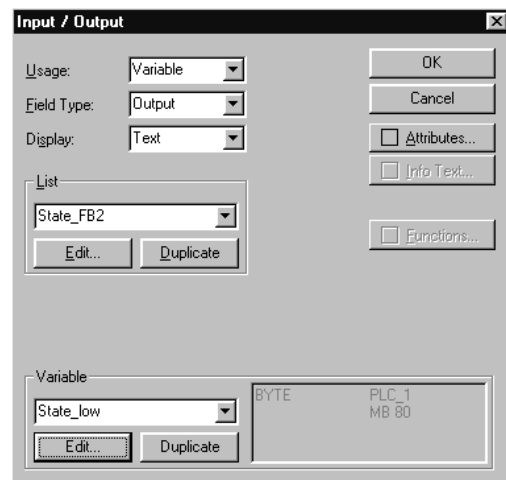


Figure 8-4 Input/output dialog

3. You must make/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

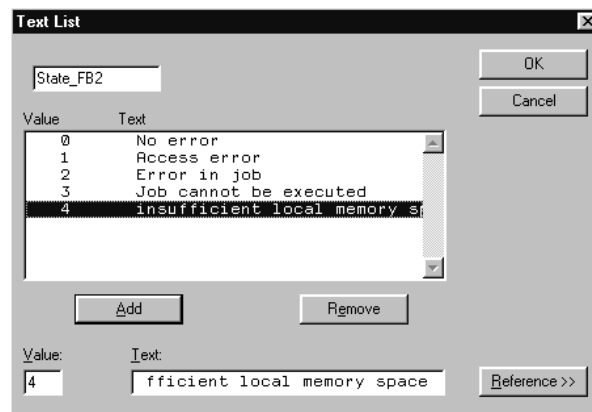


Figure 8-5 Symbol list-Text dialog

5. In dialog field **Value**, enter the error number as specified in Table 6-7 and in dialog field **Text**, the error text. When you press button **Add**, the entry is transferred to the symbol list field (see Figure 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 6-7), you should generate two of these variables so as to obtain a comprehensive troubleshooting function.

To do so, proceed as described above (from point 2. menu command **Display ► Edit/insert field**).

Once you have transferred the project to the OP and started function block FB2, the corresponding error text appears on your OP.



Description of Functions

General

Two firmware variants FM357-L and FM357-LX are available with product version 2 and later of the FM 357.

Table 9-1 Differences between FM357-L and FM357-LX

Function	FM357-L	FM357-LX	in Section
Gantry	–	x	9.13.2
Travel to fixed stop	–	x	9.15, 10.11
Oscillation	–	x	10.23
Feed interpolation	–	x	10.5.2
"Path velocity" system variable	–	x	10.22, Table 10-5
Overlaid motion in synchronized actions	–	x	9.13.4
SPLINE interpolation	–	x	10.6
Subroutine as action (synchronized action)	–	x	10.22
Static synchronized actions in all operating modes	–	x	10.22
Axial measurements in synchronous actions	–	x	10.22

The parameterization of the functions described in this Section is supported by the "Parameterize FM 357" tool.

Note

This documentation specifies all units for standard system parameters in the **metric** system of measurement.

Section overview

Section	Title	Page
9.1	Configuration	9-3
9.2	Encoder	9-8
9.3	Position control	9-15
9.4	Velocity and acceleration	9-24
9.5	Monitoring	9-30
9.6	Referencing and alignment	9-39
9.7	Output of M, T and H functions	9-49
9.8	Digital I/Os	9-52
9.9	Path switching signals (software cams)	9-56
9.10	Operating modes	9-62
9.11	NC program execution	9-64
9.12	Asynchronous subroutines (ASUB)	9-66
9.13	Coupled motion	9-69
9.14	Measurement	9-86
9.15	Travel to fixed stop	9-88
9.16	EMERGENCY STOP	9-97

9.1 Configuration

Internal system of measurement

Before you start to set other parameters, you must select the internal system of measurement. 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 FM 357 parameterizing tool and in the FM 357:

- 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.

System of measurement	Internal values	Input in 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).

Note

If you change the internal system of measurement at a later stage, 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.

Max. cycle time UP

The max. cycle time [ms] indicates to the FM 357 what the period of one OB 1 run will be.

It is evaluated for auxiliary function output in G64 mode.

The path feed is reduced during the process so that it is not necessary to wait for acknowledgement of the auxiliary function at the end of the block.

Override coding

The path override (user DB “NC signals”, DBX21.0) and the axis override (user DB “Axis signals”, DBX13.0) can be signalled as a Gray code or binary code by the CPU. Parameter “Override coding” determines how the code will be interpreted by the FM.

For further information about overrides, please refer to Section 6.9.3.

Axis number

An axis can be activated or deactivated via its axis number. Four axes are active in the default setting. 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.

Axis name

Various different names can be assigned to the different axis types of the FM 357.

- **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.

Note

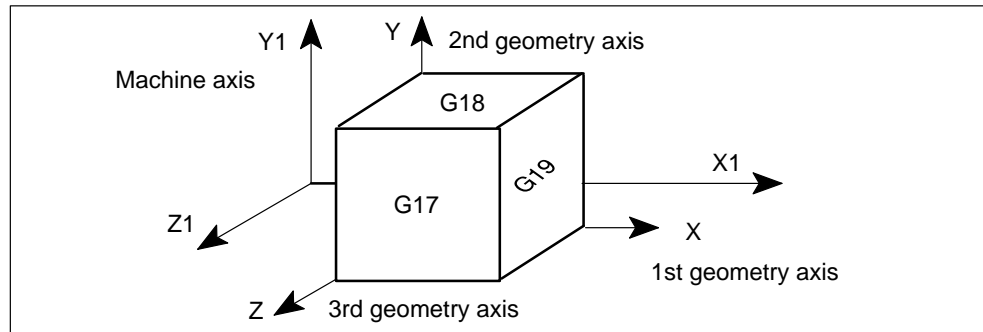
The following cannot be used as an axis name:

- Address letters (D, E, F, G, H, I, J, K, L, M, N, P, R, S, T)
 - Instructions which are used for programming
-

Assignment of axes to workpiece coordinate system

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.



Axis type

You can select the following types of axes:

- Linear axis
- Rotary axis
- Modulo rotary axis

Note

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 $\pm 399\,999.999$ inches

Programming range: $\pm 999\,999.999$ mm or $\pm 399\,999.999$ inches

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: $0 \dots 359.999^\circ$

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.

No axis-specific interface signals are output to the CPU during simulation.

- **Servo drive**

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, and a higher-level position control loop in the FM 357.

- **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.

External master value

An axis can be defined as an external master in conjunction with the master-value coupling function (see Section 9.13.3). An encoder must be connected to the corresponding measuring system interface for actual-value sensing. The FM internally generates a “simulated” master value from the actual value for injection as an input variable for the curve table.

No position control is active and no setpoints are output.

VDI output

If an axis is operated in simulation mode, parameter “VDE output” can be set to define whether the FM will transfer the interface signals user DB “Axis signals” to the CPU.

This setting allows you to test, for example, operational sequences in combination with axis motions in the CPU.

Parameters for configuration

The following parameters are relevant with respect to configuring:

Parameters	Value/Meaning	Unit
Internal system of measurement	metric = 10^{-3} (default) inches = 10^{-4}	[mm] [inches]
Max. cycle time of user program	40 (default) 10...200	[ms]
Override coding	Gray (default) The override value supplied by the CPU is interpreted as a Gray code by the FM. Binary The override value supplied by the CPU is interpreted as a binary code by the FM.	–
Number of R parameters (see Section 10.17)	100 (default) 0...100	–
Number of curve tables (see Section 9.13.3)	0 (default) 0...20	–
Number of curve segments (see Section 9.13.3)	0 (default) 0...80	–
Number of curve table polynomials (see Section 9.13.3)	0 (default) 0...160	–
Axis name	Machine axis (X1, Y1, Z1, A1 – default) Geometry axis (X, Y, Z – default) Special axis (A – default) (max. 8 characters)	–
Axis type	Linear axis = (10^{-3} mm or 10^{-4} inches) Rotary axis = (10^{-3} degrees) Modulo rotary axis = (10^{-3} degrees)	–
Drive	Simulation Servo drive Stepper motor without encoder Stepper motor with encoder	–
External master value	no (default) The axis cannot be used as an external master value. yes The axis is an external master value.	–
VDI output (for simulation)	no (default) The user DB “Axis signals” (interface signals) are not transferred to the CPU. yes The user DB “Axis signals” (interface signals) are transferred to the CPU.	–

9.2 Encoders

General

The following encoders can be connected to the measuring system interface of the FM 357:

- Incremental encoders
- Absolute encoders (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 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 encoders

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)
$I_G = \frac{S \text{ (mm)}}{4 \cdot i_{GS} \cdot R \text{ (mm)}}$	$S_G = \frac{S \text{ (mm)}}{i_{GS} \cdot R \text{ (mm)}}$

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], [inches], [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 $\left[\frac{\text{number of encoder revs}}{\text{turntable revolution}} \right]$

Note

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)	-
Encoder mounting	Motor: Indirect path encoding (default) Machine: Direct path encoding	-
Encoder type	Incremental: Incremental encoder (default) Absolute: Absolute encoder (SSI)	-
Distance per spindle revolution	10 (default) Value range 0.001 to 100 000	[mm/rev]
Load gearbox (LG)	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 $\frac{\text{No. of motor revolutions}}{\text{No. of encoder revolutions}} = \frac{1 \text{ to } 10\,000}{1 \text{ to } 10\,000}$	-

9.2.1 Incremental encoders

General

These encoders supply pulses which are added up in the FM 357 to produce an absolute value. When the FM 357 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 for adapting incremental encoders:

Parameters	Value/Meaning	Unit
Increments per encoder revolution	2048 (default) Value range 2 to 16 384 Number of increments per revolution on a rotary encoder	–
Scale division	0.01 (default) 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 → parameter “Scale division” = 0.002 mm)	[mm]
Linear measurement system is negative	no: On axis movement absolute value goes plus to plus (positive) yes: On axis movement absolute value goes plus to minus (negative)	–

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)

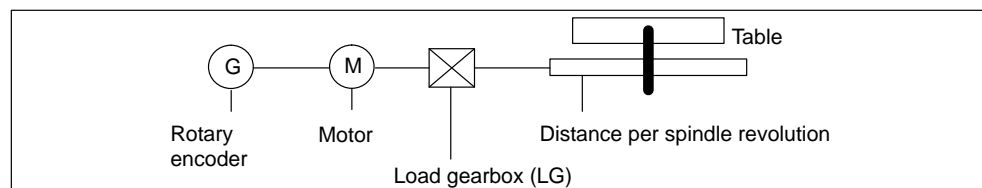


Figure 9-1 Rotary encoder on motor

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 mm = 20 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 4.6

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

Encoders with different telegram lengths can be used.

- Absolute encoders (SSI) on linear axes

Make sure the value range of the encoder is at least equal to the traversing distance of the axis.

- **Absolute encoders (SSI) on rotary axes**

Make sure the absolute value registered by the encoder is equal to precisely one revolution of the rotary axis.

Parameters for encoder adaptation

The following parameters are provided on the FM 357 for adapting absolute encoders:

Table 9-2 Parameters for absolute encoders

Parameters	Value/Meaning	Unit
Baud rate	Data transfer rate (for all encoder inputs) 250 kHz (default) 400 kHz 500 kHz 1 MHz	[kHz] [MHz]
Coding	Output code of encoder: Gray code (default) Binary code	–
Parity test	Yes (default) No	–
Parity	Uneven (default) Even	–
Measurement	Not available (default) Available	–
Sensing probe connection	Input 4 (default) Input 5	-

Table 9-2 Parameters for absolute encoders, continued

Parameters	Value/Meaning	Unit
Message frame length	25-bit multi-turn (defaults setting) 13-bit single-turn 21-bit multi-turn	–
Increments per encoder revolution	8192 only with 25-bit multi-turn and 13-bit single-turn 4096 2048 ... 2^1	–

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.

Note

The encoder covers an absolute traversing distance of 256 · 6 mm = 1 536 mm.

Connection of encoders

See Section 4.6

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) 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.

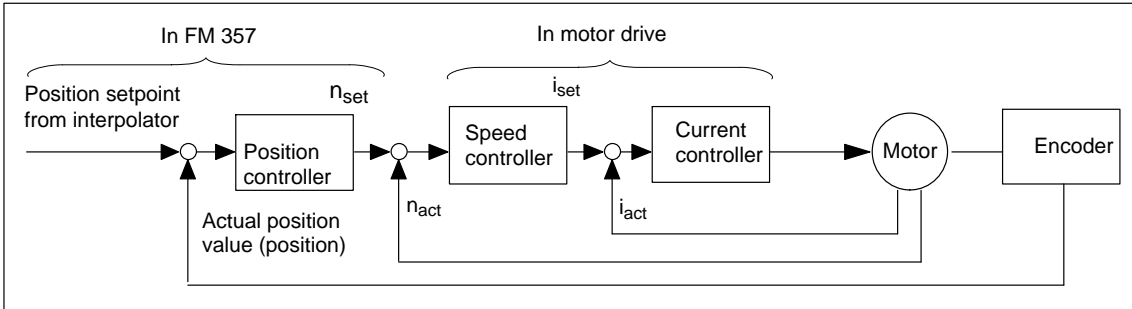


Figure 9-2 Control loops

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.

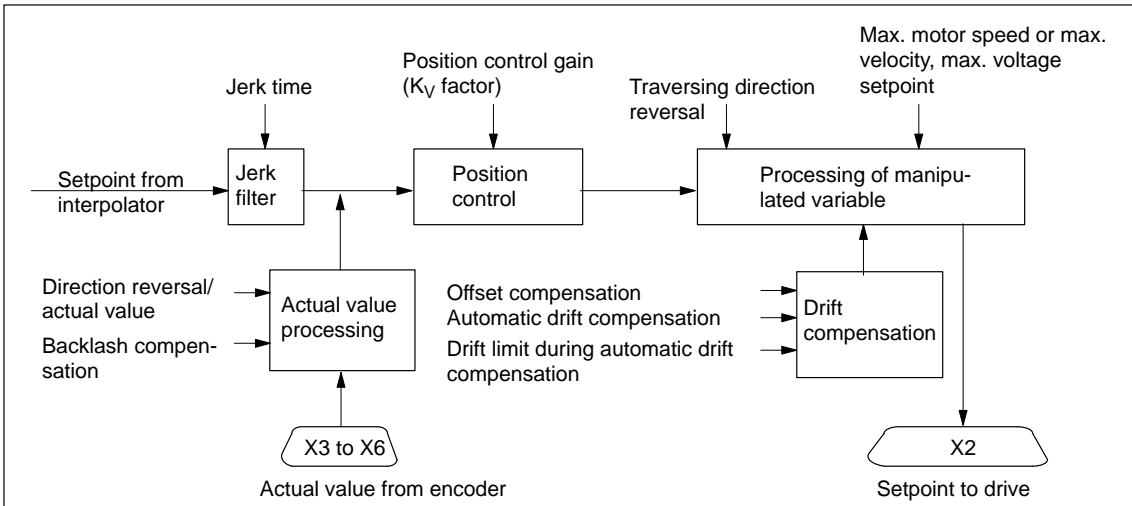


Figure 9-3 Overview of position controller

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.

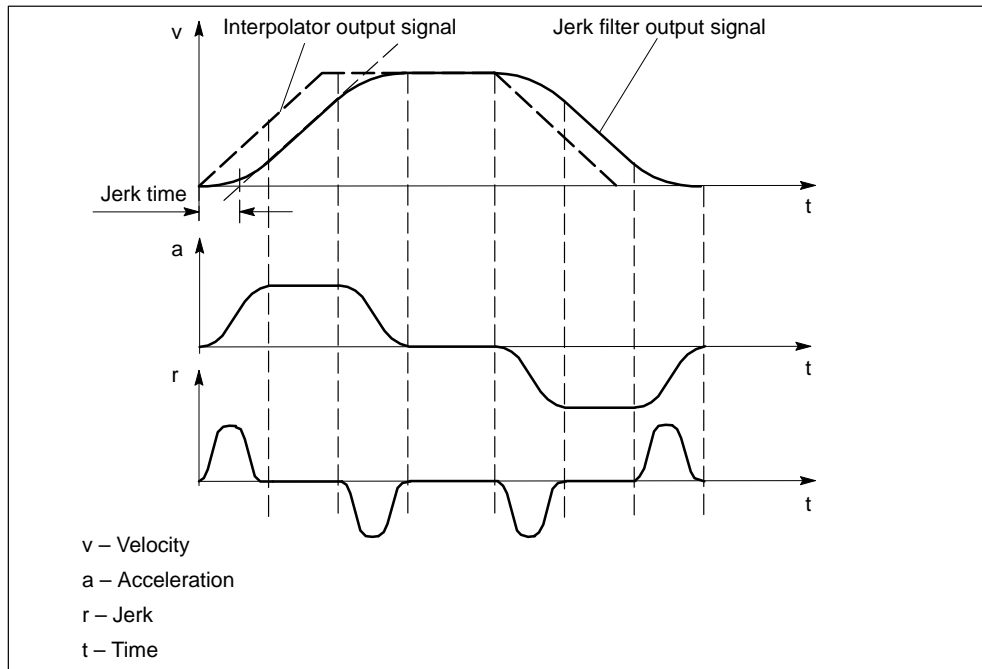


Figure 9-4 Jerk limitation on position controller level

Parameters	Value/Meaning	Unit
Jerk filter active	no Jerk filter not active (default) yes Jerk filter active	–
Jerk time	1 (default) 0 to 100	[ms]

Note

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 9.4) 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".

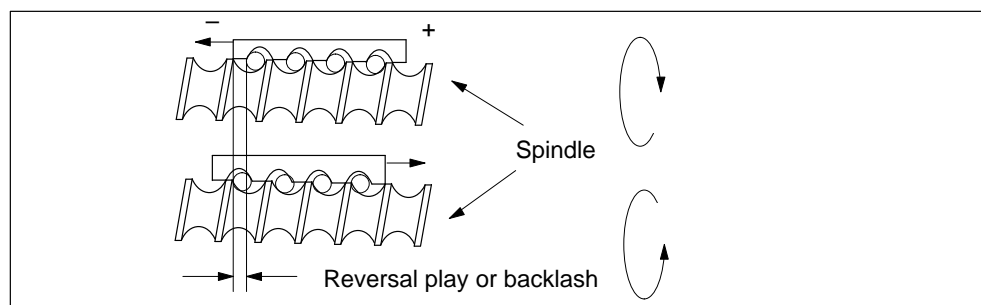
Note

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) yes Reversal	–
Backlash compensation	0 (default) –10 000 to +10 000 Positive value: on positive backlash Negative value: on negative backlash	[μm], [10^{-3} deg]

- **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.

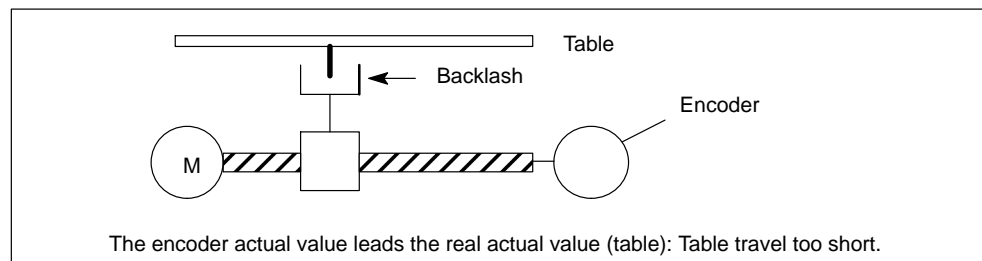


Figure 9-5 Positive backlash (normal condition)

- **Negative backlash:**

The encoder lags behind the machine part (e.g. table); the table travels too far.

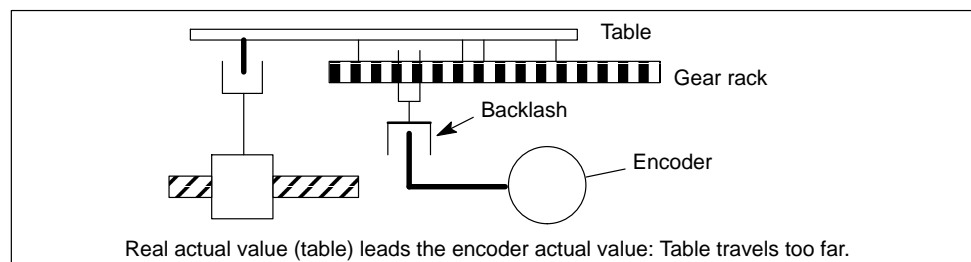


Figure 9-6 Negative backlash

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 design characteristics of the axis, the greater the achievable K_v factor and the better the axis parameters from a 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) 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 mm/min]/mm], [[10^3 degree/min)/degree]

Note

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)
	yes	Reversal

Note

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" 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) 1 to 999 999	[rev/min]
Maximum velocity V_{\max} [Axis]	10 000 (default) 1 to 999 999	[mm/min], [rev/min]
Set voltage, max.	8 (default) 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 (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” 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) 1 to 999 999	[rev/min]
Maximum velocity V_{\max} [Axis]	10 000 (default) 1 to 999 999	[mm/min], [rev/min]

The “Parameterize FM 357” 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 D/C 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 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) -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 effects cause a shift in the zero point 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”.

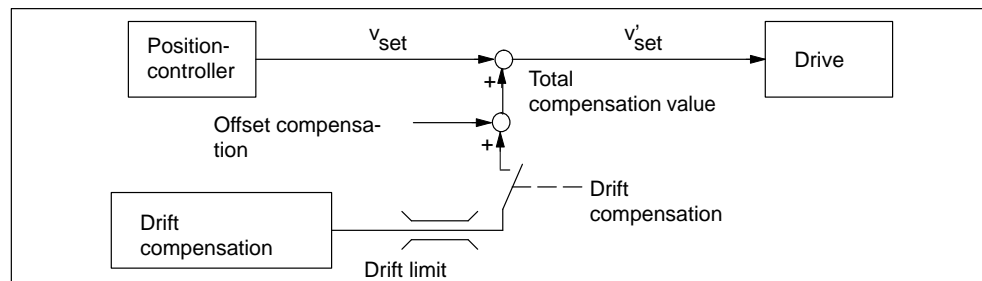


Figure 9-7 Composition of the total compensation value

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	100 (default) -3 000 to +3 000 If the drift value exceeds this parameter setting, an error is signalled and the drift restricted to the parameterized limit.	[mV]

Note

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)	–

Time constant Current 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 for current control loop	0.5 (default) 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) 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**

Note

A low acceleration and high velocity results in very long acceleration phases. This allows the controller difference to be read off easily.

9.4 Velocities and acceleration rates

Velocities

The following velocities can be set on the FM 357 for the different operating modes:

Velocity	Effective in operating mode
Maximum velocity Positioning velocity	Automatic, MDI
Axis velocity Rapid traverse override	Jogging 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 6.3).

This feedrate is valid until an axis-specific feedrate is programmed in the NC program.

Parameters	Value/Meaning	Unit
Positioning velocity	10 000 (default) 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) 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) 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 independently of the setting in the NC program (see Section 10.7.3):

BRISKA(axis) Brisk acceleration
 SOFTA(axis) Soft acceleration
 DRIVEA(axis) Drive acceleration

Parameters	Value/Meaning	Unit
Acceleration pattern	Brisk acceleration (default) Soft acceleration Drive acceleration	–
Initial setting	√ 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.

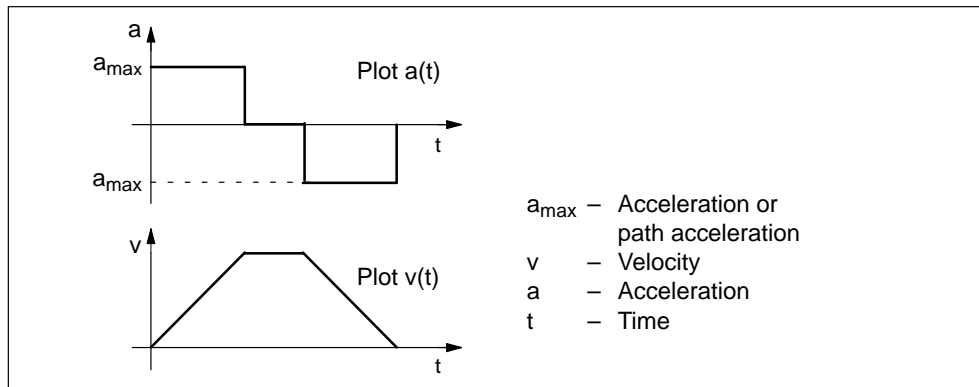


Figure 9-8 Velocity and acceleration plotted for brisk acceleration

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) 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

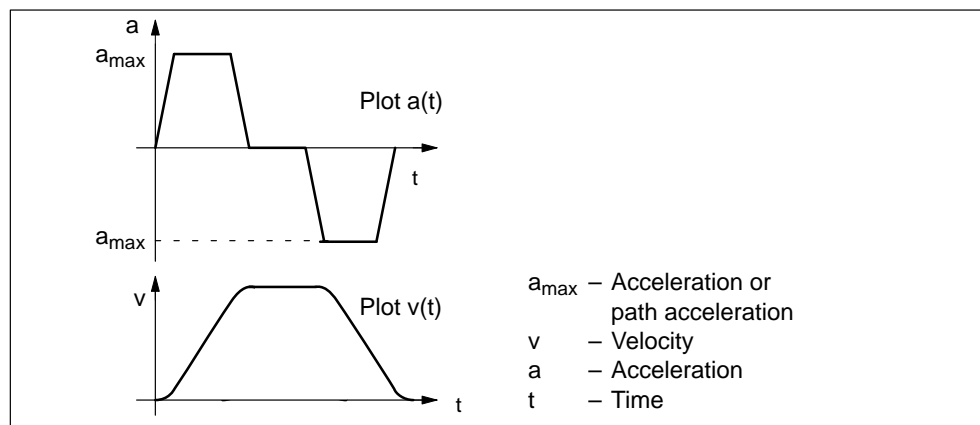


Figure 9-9 Velocity and acceleration profiles with soft acceleration

Parameters	Value/Meaning	Unit
Acceleration	1 (default) 0 to 10 000	[m/s ²], [rev/s ²]
Jerk	1 000 (default) 0 to 100 000	[m/s ³] [rev/s ³]

”Jerk” is the change in acceleration rate per time unit.

Drive 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 (drive acceleration).

The creep acceleration is valid above the defined creep velocity, while the “normal” acceleration applies below it.

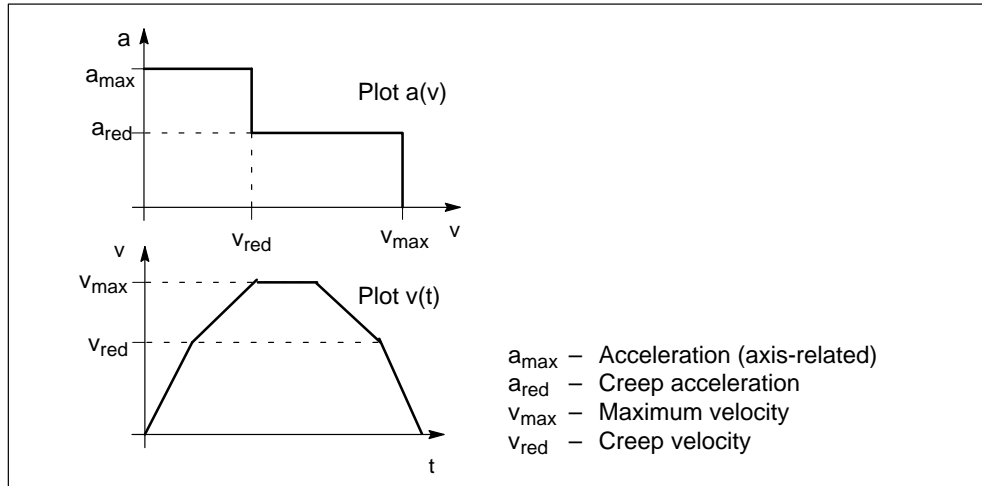


Figure 9-10 Axial acceleration and velocity characteristic

Parameters	Value/Meaning	Unit
Acceleration	1 (default) 0 to 10 000	[m/s ²], [rev/s ²]
Creep velocity	10 000 (default) 0 to 999 999	[mm/min], [rev/min]
Creep acceleration	1 (default) 0 to 10 000	[m/s ²], [rev/s ²]

Note

Drive acceleration can only be parameterized in relation to an axis. The path response results from the calculation with the axes involved.

Path action

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 independently of the setting in the NC program (see Section 10.7.3):

BRISK	Brisk acceleration
SOFT	Soft acceleration
DRIVE	Drive 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) 0 to 1 000	[m/s ²]															
Path jerk	100 (default) 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.

Note

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.

9.5 Monitoring functions

Overview

In this section, you can find information about:

- Monitoring of movements
- Monitoring of encoders
- Hardware and software limit switches

9.5.1 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 "Target range fine" after deceleration
Clamping tolerance	Interface signal "Activate terminals" (user DB, "Axis signals", DBX42.3)
Set speed	Active position control
Actual speed	Active actual values

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 the timer runs out, a check is made to determine whether the following error is smaller than the limit value set in parameter "Coarse target range" (in blocks with coarse target range) or in parameter "Fine target range" in blocks programmed with fine target range).

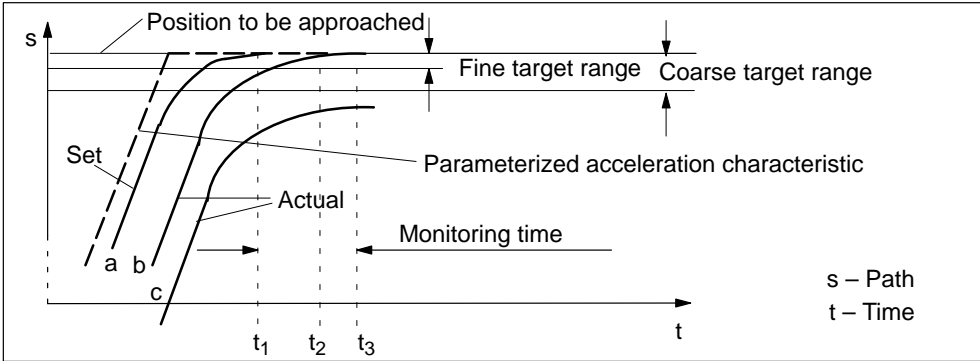


Table 9-3 Position monitoring time

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 time	1 (default) 0 to 100	[s]
Coarse target range	0.04 (default) 0 to 1 000 Must not be set smaller than the target range fine.	[mm], [degrees]
Fine target range	0.01 (default) 0 to 1 000	[mm], [degrees]

Note

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 "Fine target range" 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 (coarse target range) user DB, "Axis signals", DBX15.1)
 - Position reached, stop (fine target range) user DB, "Axis signals", DBX15.2)
-

Following error monitoring

Movement of axis

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.

Note

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) 0 to 1 000	[mm], [degrees]
Zero speed control delay	0.4 (default) 0 to 100	[s]
Zero speed range	0.2 (default) 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) 0 to 10 000	[mm/min], [rev/min]

Clamping monitor

If the axis must be clamped after positioning, then the clamping monitor can be activated with interface signal "Clamping in progress" (user DB, "Axis signals", DBX42.3).

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) 0 to 1 000	[mm], [degrees]

Speed setpoint monitoring

The speed setpoint monitor can be used to check whether the physical limitation of the drive for a **servo axis** (10 V maximum voltage for speed setpoint on analog drives) is being 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) 0 to 200 % value referred to the maximum motor speed or maximum velocity	[%]

The set speed monitoring system can also be used for test purposes.

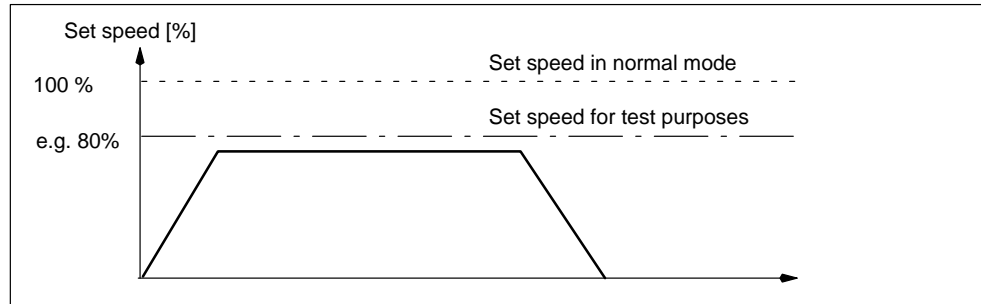


Figure 9-11 Speed setpoint monitoring

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) 0 to 100	[s]

Note

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

With this monitoring function, the actual velocity is checked for the violation of a permissible limit value which is programmed in parameter “Actual velocity”.

The monitoring system always triggers a response if the actual values returned are below the limit frequency.

Parameters	Value/Meaning	Unit
Actual velocity	11 500 (default) 0 to 9 999 999	[mm/min], [rev/min]

9.5.2 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, "Axis signals", DBX12.2) is enabled	Interface signal "Rotation monitoring stepper motor" (user DB, "Axis signals", DBX17.2) is enabled. 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) 0 to 1 500 000	[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 marker errors at which the monitor must respond is also specified.

Parameters	Value/Meaning	Unit
Zero marker monitoring	OFF: HW encoder monitoring ON (default) 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".

Rotation monitoring Stepper motor

The BERO for rotation monitoring is connected in the same way as BEROs used for referencing (see Section 9.6.2).

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, "Axis signals", DBX12.2).

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.

Parameters	Value/Meaning	Unit
Number of increments	2 000 (default) 10 to 1 000 000	–
Increment tolerance	50 (default) 10 to number of increments	–

Note

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!

9.5.3 Hardware and software limit switches

General

Possible limit switch monitors:

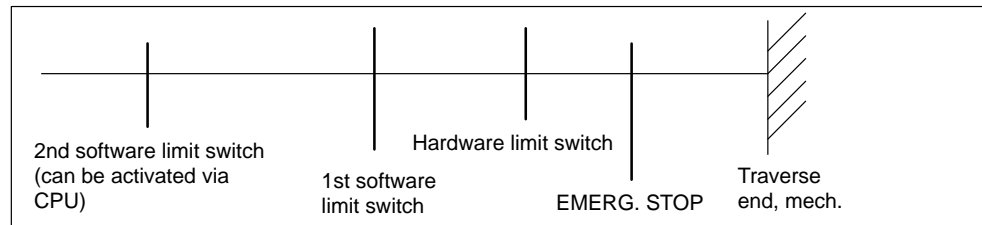


Figure 9-12 Travel limits

The following table provides an overview of the monitoring systems and their features.

Table 9-4 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 via interface signal "Hardware limit switch plus/minus" (user DB, "Axis signals", DBX50.1/50.0) 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) -100 000 000 to +100 000 000	[mm], [degrees]
1st software limit switch minus	-100 000 000 (default) -100 000 000 to +100 000 000	[mm], [degrees]
2nd software limit switch plus	100 000 000 (default) -100 000 000 to +100 000 000	[mm], [degrees]
2nd software limit switch minus	-100 000 000 (default) -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, “Axis signals”, DBX50.3/50.2). 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.

9.6 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.

Note

The following monitoring systems have no effect on a machine axis which has not been referenced or aligned:

- Working area limitation
 - Software limit switches
-

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 “Plus direction or Minus direction” (user DB, “Axis signals”, DBX11.7/11.6), depending on setting in parameter “Reference point approach”. 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.

NC Start without reference point approach

NC programs are started as a function of the setting in parameter “NC 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
NC Start without reference point approach	<p>No (default) 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) 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 “NC Reset” (user DB, “NC signals”, DBX12.7) 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, “Axis signals” DBX15.0).

9.6.1 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) 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, "Axis signals", DBX11.1) must be programmed in the user program.

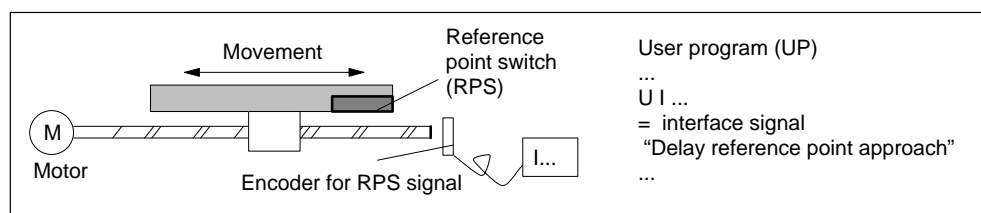


Figure 9-13 Mounting a reference point switch (RPS)

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, ...

Note

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:

$$\text{Minimum length} = \frac{(\text{Referencing velocity or creep velocity})^2}{2 \cdot \text{axis acceleration}}$$

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

Referencing parameters

The following table describes all parameters required to reference an incremental encoder:

Table 9-5 Parameters for referencing

Parameters	Value/Meaning	Unit
Reference point approach direction	Plus (default) 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) 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) –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 offset	–2 (default) –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) 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) 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 9-5 Parameters for referencing, continued

Parameters	Value/Meaning	Unit
Referencing velocity	5 000 (default) 0 to 999 999 The axis travels at this velocity towards the reference point switch (RPS).	[mm/min], [rev/min]
Creep velocity	300 (default) 0 to 999 999 The zero pulse or BERO is approached at this velocity.	[mm/min], [rev/min]
Approach velocity	1 000 (default) 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 movements

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

Travel to reference point switch

- The feedrate override and feed hold are active.
- The axis can be stopped/started with NC Stop/NC 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.

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 NC Stop/NC Start.

Travel to reference point

- The feedrate override and feed hold are active.
- The axis can be stopped/started with NC Stop/NC 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.

9.6.2 Referencing for 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 (see Section 4.7):

- X1, pin 13 for BERO of axis 1
- X1, pin 14 for BERO of axis 2
- X1, pin 15 for BERO of axis 3
- X1, pin 16 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): 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>	–

9.6.3 Alignment for 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
- if the mechanical connection between the encoder and the load was separated and not joined in the exact position.

Note

The controller cannot detect all cases where it is necessary to repeat the encoder alignment!

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) Plus direction The encoder is aligned with a known position in this direction.	–
Encoder alignment status	Not aligned (default) Enabled Aligned	–
Actual value (alignment value)	0 (default) –100 000 to +100 000 This parameter specifies the position at which the axis should be located at a known position.	[mm], [degrees]

Encoder alignment procedure

Basic procedure:

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".

Note

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.

9.7 Output of M, T and H functions

General

The M, T and H functions programmed in the NC program (see Section 10) are output at 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) Output during movement Output after movement	–

Interface signals:

The following interface signals are available for M functions:

- Interface signals as checkback signals
 - Change in auxiliary function (user DB, “NC signals”, DBX15.5)
 - M function number (user DB, “NC signals”, DBB17.0)
- Interface signals as data set signals
 - M function number 1 (user DB, “NC signals”, DBB80.0)
 - M function number 2 (user DB, “NC signals”, DBB81.0)
 - M function number 3 (user DB, “NC signals”, DBB82.0)
 - M function number 4 (user DB, “NC signals”, DBB83.0)
 - M function number 5 (user DB, “NC signals”, DBB84.0)
- Interface signals as control signals
 - Acknowledge auxiliary function (user DB, “NC signals”, DBX11.6)

Note

When one M function is programmed in the NC block, the output takes place by way of checkback signals (i.e. high-speed output).

When more than one M function is programmed in the NC block, all M functions are output by way of data block signals (i.e. low-speed output).

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 in auxiliary function (user DB, "NC signals", DBX15.5)
- T function number (user DB, "NC signals", DBW104.0)

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 9-6 H function parameters **without** group assignment

Parameters	Value/Meaning	Unit
Output options for H functions	Output prior to movement (default) Output during movement Output after movement	–

Interface signals:

- Change in auxiliary function (user DB, "NC signals", DBX15.5)
- H function number 1 (user DB, "NC signals", DBW86.0)
- H function number 2 (user DB, "NC signals", DBW92.0)
- H function number 3 (user DB, "NC signals", DBW98.0)
- H function value 1 (user DB, "NC signals", DBD88.0)
- H function value 2 (user DB, "NC signals", DBD94.0)
- H function value 3 (user DB, "NC signals", DBD100.0)

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 T5

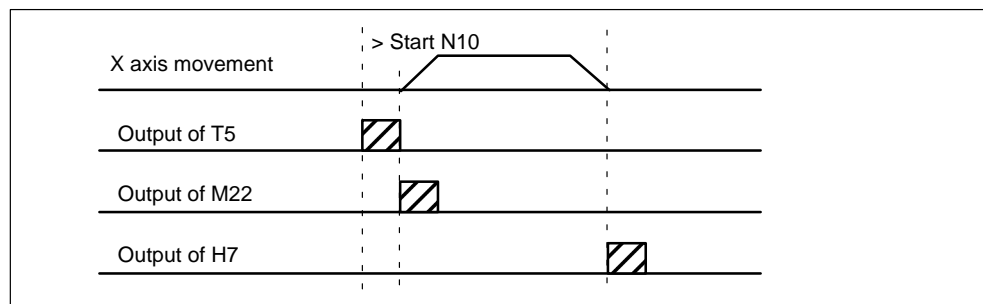


Figure 9-14 Example for output of M, T and H functions

9.8 Digital I/Os

General

You can use the following types of digital input and output on the FM 357:

Table 9-7 Digital I/Os on the FM 357

Type	Inputs		Outputs	
	Number	Function	Number	Function
On-board inputs (see Section 9.8.1)	2 4	Measurement (probes 1 and 2) for BERO signal or for starting ASUBs or used freely	None	–
I/Os over local P bus (see Section 9.8.2)	16	free: implemented with signal modules (SMs) on the local P bus	16	Free: implemented with signal modules (SMs) on the local P bus

9.8.1 Digital on-board inputs

Sensing probe inputs (X1 pins 17 and 18)

Measuring pulse inputs 1 and 2 (see Section 9.14).

In the measuring function, these two inputs are used for the connection of sensing probes.

Inputs (X1 pins 13, 14, 15 and 16)

These inputs can be used for several mutually exclusive functions.

- Use as a BERO input for axis 1 to 4 (see Section 9.6)
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.
- Use for starting asynchronous subroutines (ASUBs) (see Section 9.12)
A subroutine can be assigned to one of these inputs in the NC program. After the “ready” enable, the subroutine is started and processed over this input, depending on the 0/1 edge.

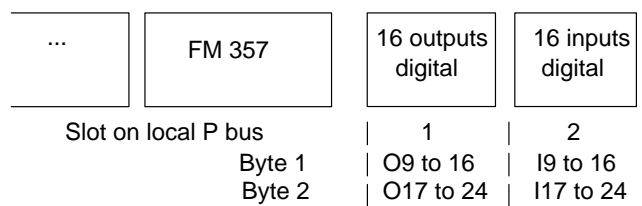
Parameter settings for hardware configuration

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 used	None (default) Slot 1 Slot 1+2	–
Module size	1 byte (default) 2 bytes	–
Byte 1	Inputs Outputs	–
Byte 2	Inputs Outputs	–
9 to 16 17 to 24	9 to 16 (default); 17 to 24 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 used	Slot 1+2	
	Slot 1	Slot 2
Module size	Byte 2	Byte 2
Byte 1	Outputs 9 to 16	Inputs 9 to 16
Byte 2	Outputs 17 to 24	Inputs 17 to 24

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:
 The status of any input can be read.
- Outputs:
 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.

Table 9-9 Status of digital inputs

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
User DB, "NC signals", DBB36	Digital input							
	16	15	14	13	12	11	10	9
User DB, "NC signals", DBB37	Digital input							
	24	23	22	21	20	19	18	17

Table 9-10 Disabling digital outputs

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
User DB, "NC signals", DBB56	Digital output							
	16	15	14	13	12	11	10	9
User DB, "NC signals", DBB60	Digital output							
	24	23	22	21	20	19	18	17

Table 9-11 Status of digital outputs

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
User DB, "NC signals", DBB38	16	15	14	13	12	11	10	9
User DB, "NC signals", DBB39	24	23	22	21	20	19	18	17

9.9 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.

When the function has been activated for the axis, it generates cam signals on the basis of the positions specified for the minus and plus cams, and outputs these positions as interface signals.

The cam signals can also be output over digital outputs on the local P bus.

The "Limit switching signals" function operates in all modes and remains active even after a Reset or EMERGENCY STOP.

9.9.1 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	Value/Meaning	Unit
Cam pair Axis number	1 0 Not assigned (default) 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 9-12 Cam position parameters

Parameters	Value/Meaning	Unit
Cam position Minus cam	0 (default) -100 000 000 to +100 000 000	[mm], [degrees]
Cam position Plus cam	0 (default) -100 000 000 to +100 000 000	[mm], [degrees]

Note

Cam positions are referred to the selected system or measurement (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.

Parameters	Value/Meaning	Unit
Lead time/delay time Minus cam	0 (default) -100 to +100 Positive value = lead time Negative value = delay time	[s]
Lead time/delay time Plus cam	0 (default) -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) 1 → 0 (inverted)	–
Signal level Plus cam	0 → 1 (default) 1 → 0 (inverted)	–

Assignment to digital outputs

This parameter defines which digital outputs are assigned to the cams. The assignment must be made in bytes.

Parameters	Value/Meaning	Unit
Assignment to digital outputs, minus cam	No assignment (default) Digital outputs 9 to 16 Digital outputs 17 to 24	–
Assignment to digital outputs, plus cam	No assignment (default) Digital outputs 9 to 16 Digital outputs 17 to 24	–

Activation

The function is activated for every axis via the following interface signal:

"Activate software cam" (user DB, "Axis signals", DBX12.0)

The successful activation of all cams of an axis is reported by the interface signal:

"Software cam active" (user DB, "Axis signals", DBX15.3)

Note

Activation in the user program can be tied to other conditions (e.g. axis referenced).

9.9.2 Generation of limit switching signals

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.

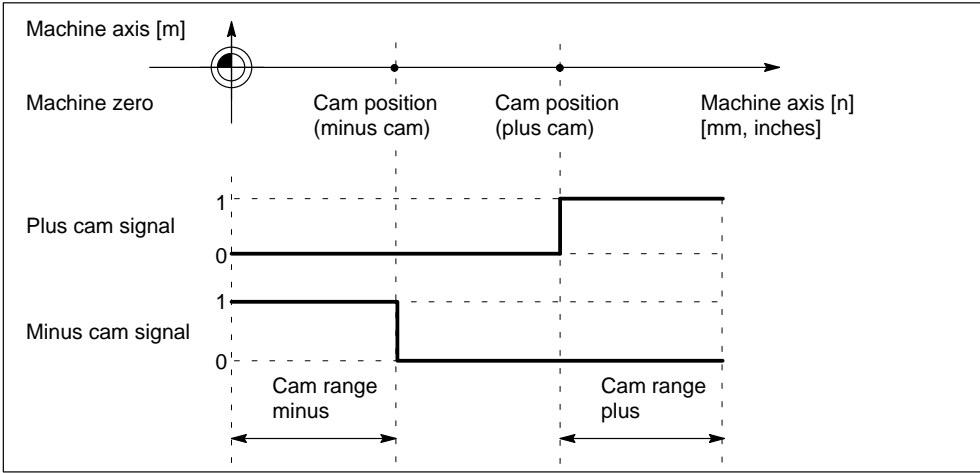


Figure 9-15 Limit switching signals for linear axis (minus cam < plus cam)

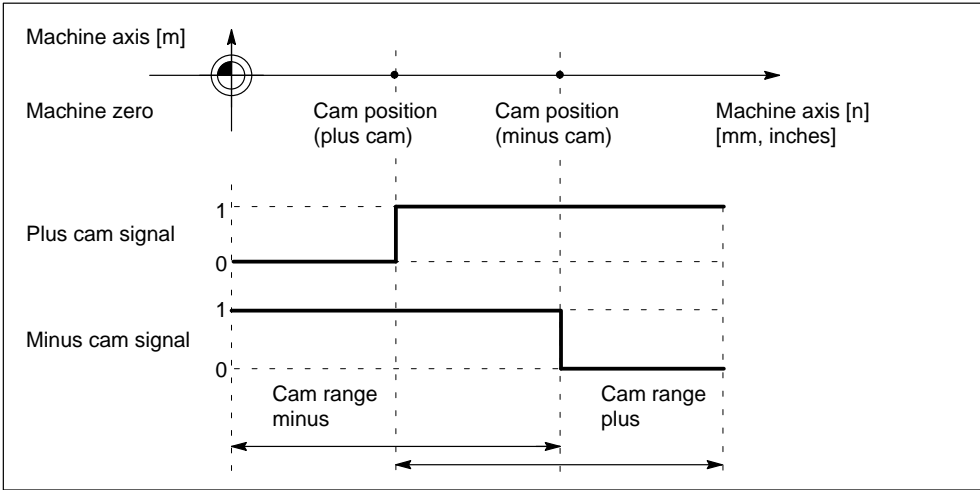


Figure 9-16 Limit switching signals for linear axis (plus cam < minus cam)

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.

Note

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.

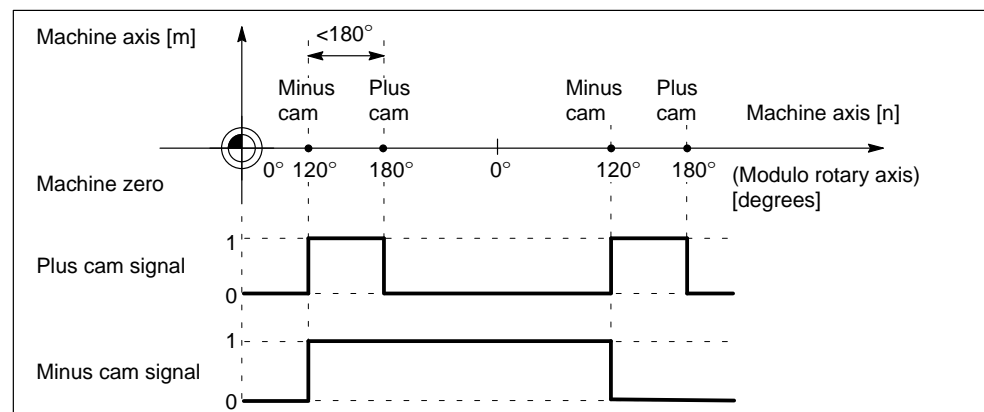


Figure 9-17 Limit switching signals for modulo rotary axis (plus cam – minus cam < 180°)

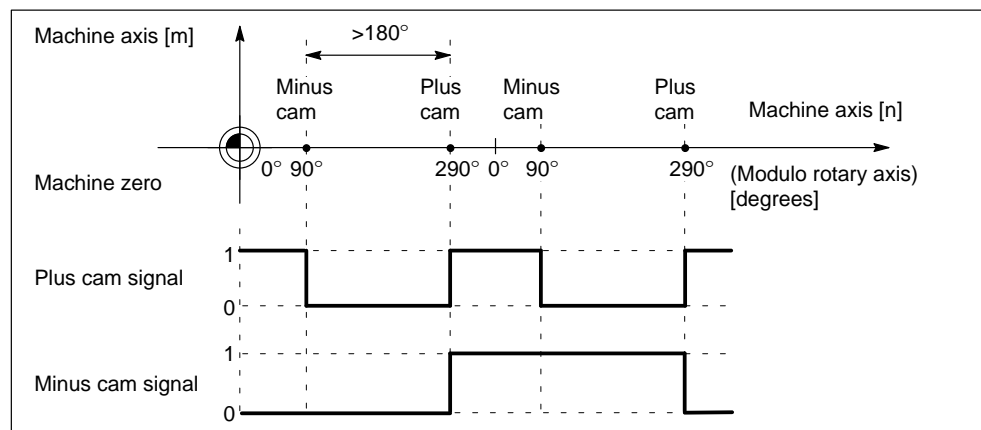


Figure 9-18 Limit switching signals for modulo rotary axis (plus cam – minus cam > 180°)

9.9.3 Output of limit switching signals

Output to digital outputs

Cam signals are output to the digital outputs on the local P bus in the position control cycle.

The assignment to the hardware bytes is defined for 8 cam pairs by parameterization.

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 9-13 Software cam minus/plus

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
User DB, "NC signals", DDB32	Software cam minus							
	8	7	6	5	4	3	2	1
User DB, "NC signals", DDB33	Software cam plus							
	8	7	6	5	4	3	2	1

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
...

```


9.10 Operating modes

General

The following modes are available on the FM 357:

Table 9-14 Operating modes and their properties

Mode	Property
Jog (J) OL control and checkback signals Parameters	<p>In this mode, the traversing movement of an axis is defined using a direction key (R+ or R-).</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: "TIPPEN" control signal Mode checkback: "TIPPEN_A" checkback signal Start movement: "R+" or "R-" control signal</p> <p>Parameter "Axis velocity" Parameter "Rapid traverse"</p>
Incremental travel relative (ITR) OL control and checkback signals Parameters	<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 traversing movement is started with the direction keys (R+ and R-).</p> <p>The movement is stopped with NC Stop and the distance to go is deleted.</p> <p>The movement is interrupted when the direction key is no longer depressed or if an NC Stop is detected while the direction key is depressed. The next time you press the direction key, the movement is continued across the distance to go. Pressing Reset aborts the movement and deletes the distance to go.</p> <p>Select mode: "TIPPEN" control signal Mode checkback: "TIPPEN_A" checkback signal Increment command: "BP" control signal Start movement: "R+" or "R-" control signal</p> <p>Parameter "Axis velocity" Parameter "Rapid traverse"</p>
Reference point approach (REF) OL control and checkback signals Parameters	<p>Approach a reference point on axes with incremental encoders.</p> <p>A direction key (R+ or R-) is used to start the reference point approach, and reference the axis in accordance with the parameter definitions.</p> <p>Select mode: "REFPKT" and "TIPPEN" control signals Mode checkback: "REF_A" and "TIPPEN_A" checkback signals Start movement: "R+" or "R-" control signal Referencing: "VER_RPS" control signal</p> <p>(see Section 9.6.1)</p>

9.11 NC program execution

General

In "Automatic" mode, NC programs can be processed independently by the FM 357. The NC programs contain instructions for moving axes and controlling the plant.

NC program execution sequence

A typical program run is as follows:

Table 9-15 Typical program run

No.	Command	Comments
1	Write NC program and load on FM 357	Using the "Parameterize FM 357" tool
2	Selection of "Automatic" mode	See Section 6.2
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 "NC Start" (user DB, "NC signals", DBX11.0); then program status (program running) is displayed
6	M02/M30/Reset	The program status (program aborted) is displayed

Select program

An NC program stored on the FM 357 can be selected by the following methods:

- From user program (UP) with FB 4 (select program)
- Using the "Parameterize FM 357" tool
- From the OP 17 (if appropriately configured)

Program states

An NC program can assume the following states while it is running:

Table 9-16 Program states

Program state	Description
Program aborted (user DB, "NC signals", DBX15.4)	The program is selected but has not been started, or a running program was aborted with Reset.
Program interrupted (user DB, "NC signals", DBX15.3)	Indicates that the NC program can continue execution on NC Start. The NC program is interrupted when you change modes, e.g. from "Automatic" to "Jogging". In "Automatic" mode, you can continue execution with NC-Start.
Program paused (user DB, "NC signals", DBX15.2)	The NC program has been stopped, e.g. by NC Stop.
Program waiting (user DB, "NC signals", DBX15.1)	The running NC program has detected a WAIT statement. The condition for the statement has not yet been fulfilled.
Program running (user DB, "NC signals", DBX15.0)	The NC program was started with NC 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 NC 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.

- 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 via interface signal “Skip block” (user DB, “NC signals”, DBX11.5)

- Programmed stop

The M01 in the NC program causes a programmed stop during program execution.

Activation via interface signal “Activate M01” (user DB, “NC signals”, DBX64.5)

9.12 Asynchronous subroutine (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.

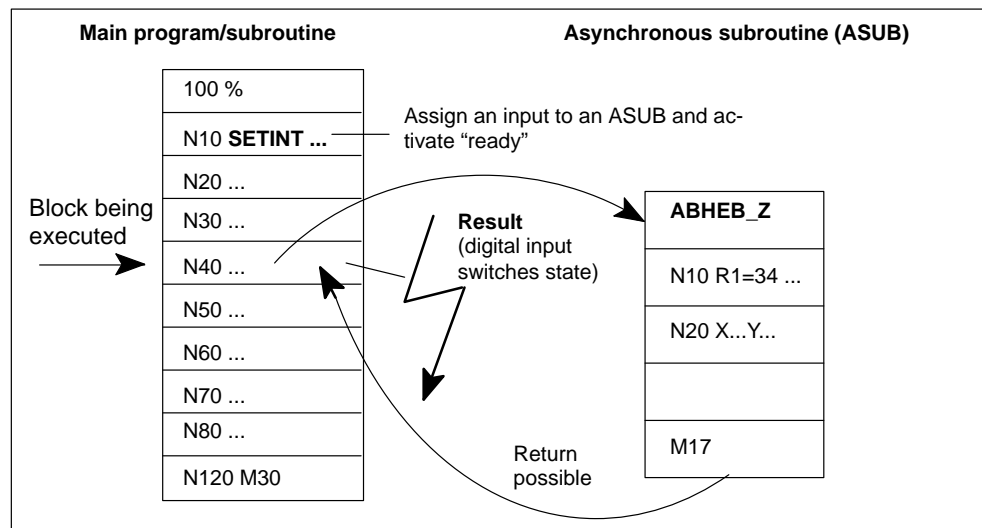


Figure 9-19 Execution of asynchronous subroutines

Instructions in NC program

The following instructions are available for programming and parameterizing ASUBs in the NC program (see Section 10.21):

Statement in NC program:

SETINT(n) PRIO=1 NAME SAVE

SETINT(n) ; Assignment of a digital input/interrupt no. (n = 1 to 4, 8)
; to an NC program to make the program an asynchronous
; subroutine

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 L ; Reposition at interruption point in
; main program/subroutine

Digital inputs for starting ASUBs

The following 4 on-board inputs are available for starting ASUBs (see Section 4.7):

- X1, pin 13 Digital input no. 1
- X1, pin 14 Digital input no. 2
- X1, pin 15 Digital input no. 3
- X1, pin 16 Digital input no. 4

The inputs can only be used if no reference point BERO is connected to this input.

Activation of an ASUB

ASUBs can be activated by two methods:

- 0/1 edge at digital input
- Call of FC ASUB (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 part 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.: REPOS L M02).

9.13 Motion coupling

Overview

In this section, you can find information about:

- Coupled motion, Section 9.13.1, page 9-69
- Gantry axes, Section 9.13.2, page 9-72
- Master-value coupling, Section 9.13.3, page 9-78
- Overlaid motion in synchronized actions, Section 9.13.4, page 9-84

9.13.1 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 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 NC program**

You can define in parameters whether the active coupled-axis groupings are canceled or retained after Reset/end of NC program.

Parameters	Value/Meaning	Unit
Active coupled-axis groupings remain valid	No: Coupled-axis groupings are dissolved (default).	–
	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.

9.13.2 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. 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-LX with product version 2 and later.

Parameterization

The following table describes all parameters required for the Gantry function.

Table 9-17 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) 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) 0 to 100 If the difference in the position actual 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, “Axis signals”, DBX115.3) set. The axis coupling is not cancelled. Important: Value = 0 → All monitoring functions are deactivated.	[mm, degrees]

Table 9-17 Gantry parameters, continued

Parameters	Value/Meaning	Unit
Trip limit	0 (default) 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 position actual 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, "Axis signals", DBX115.2) set. The gantry axes are shut down immediately.	[mm, degrees]
Trip limit for referencing	0 (default) 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 9-18 Assignment of gantry interface signals for leading and synchronized axes

Interface signal	User DB, "Axis signals"	Leading axis	Synchronized axis
Start gantry synchronization run	DBX111.4	x	
Gantry trip limit exceeded	DBX115.2		x
Gantry limit value for warning exceeded	DBX115.3		x
Gantry synchronization run ready to start	DBX115.4	x	
Gantry grouping is synchronized	DBX115.5	x	
Gantry leading axis	DBX115.6	1	0
Gantry axis	DBX115.7	1	1

Effect of other interface signals

Axis signals to axis (CPU → FM 357):

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, "Axis signals", DBX12.1) to FALSE, the synchronized axis is shut down at the same time.

Table 9-19 Effect of individual interface signals on leading and synchronized axes

Interface signal	User DB, "Axis signals"	Leading axis	Synchronized axis
Enable CL controller	DBX12.1	on both axes	
Delete residual distance	DBX11.2	axial	axial
Feed Stop	DBX11.3	on both axes	
Hardware limit switch minus/plus	DBX50.0/50.1	on both axes (axial error message)	
2nd software limit switch minus/plus	DBX50.2/50.3	axial	axial

Axis signals from axis (FM 357 → 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, "Axis signals" DBX15.6/15.7) is set simultaneously 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 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.

Referencing and synchronization process

Part 1: Referencing of leading axis

Referencing must be started in "Reference point approach" mode with interface signal "Plus direction or Minus direction" (user DB, "Axis signals" DBX11.7/DBX11.6) (see also Section 9.6, 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, "Axis signals", DBX15.0) is set for the leading axis.

Part 2: Referencing of synchronized axis

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, "Axis signals", DBX15.0) is set for the synchronized axis and the correct gantry dependency relationship restored.

If the reference point approach has been interrupted (e.g. by NC 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

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, "Axis signals", DBX115.5) 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, "Axis signals", DBX115.4) 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, "Axis signals", DBX111.4). 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, "Axis signals", DBX111.4) if the following conditions are fulfilled:

- "Reference point approach" mode must be active
- Interface signal "Gantry grouping is synchronized" (user DB, "Axis signals", DBX115.5) = 0
- Interface signal "Gantry synchronization run ready to start" (user DB, "Axis signals", DBX115.4) = 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, "Axis signals", DBX12.4)

In this case, the gantry synchronization run can be restarted directly with interface signal "Start gantry synchronization run" (user DB, "Axis signals", DBX111.4). 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:

1. Parameterization of gantry grouping and activation of parameters via parameterization tool. Parameter "Limit value for warning" must be set to 0 initially to prevent automatic start of the synchronization run and to deactivate the monitoring functions.
2. Set identical positions for the leading and synchronized axes if possible.
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 Fig. 7-4, Service 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, "Axis signals", DBX111.4).

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”.

9.13.3 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 interrelationships 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.

The “Master value coupling” function is available in product version 2 and later.

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) 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.

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 which initiate a motion stop:

- Controller enable (user DB "Axis signals" DBX12.1)
- Feed Stop (user DB "Axis signals" DBX12.3)
- NC Stop (user DB "NC signals" DBX11.1)

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 NC Reset (user DB "NC signals" DBX12.7) and end of program.

Parameters	Value/Meaning	Unit
Master value coupling remains active	<p>No (default) All active master value couplings are dissolved on NC Reset or end of program.</p> <p>Yes On NC 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 10.22, 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 10.24):

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
LEADON(FA, LE, CTAB No)	; Activation of coupling
LEADOF(FA, LE)	; 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

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 10.24).

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) 0 to 20 Maximum number of curve tables in NC memory	–
Number of curve segments	0 (default) 0 to 80 Maximum number of all curve segments	–
Number of curve table polynomials	0 (default) 0 to 160 Maximum number of all curve segments; a maximum of 3 polynomials are required for each curve segment.	–

Note

If this parameter is changed, all user data will be lost due to reorganization of the NC 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 9-20 Offset and scaling of master and slave axis positions

Parameters	Value/Meaning	Unit
Master axis position offset	0 (default) –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) –1 000 000 to + 1 000 000 Factor for master axis position (input variable for curve table)	–
Slave axis position offset	0 (default) –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) –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:

$$\mathbf{FAS} = \text{OFFSET_FA} + \text{SCALE_FA} * \text{CTAB}(\text{OFFSET_LA} + \text{SCALE_LA} * \mathbf{LA})$$

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: $LA_{\min} \leq (\text{OFFSET_LA} + \text{SCALE_LA} * \mathbf{LA}) \leq LA_{\max}$

Parameters OFFSET_FA and SCALE_FA can shift the slave axis position out of the definition range of the curve table.

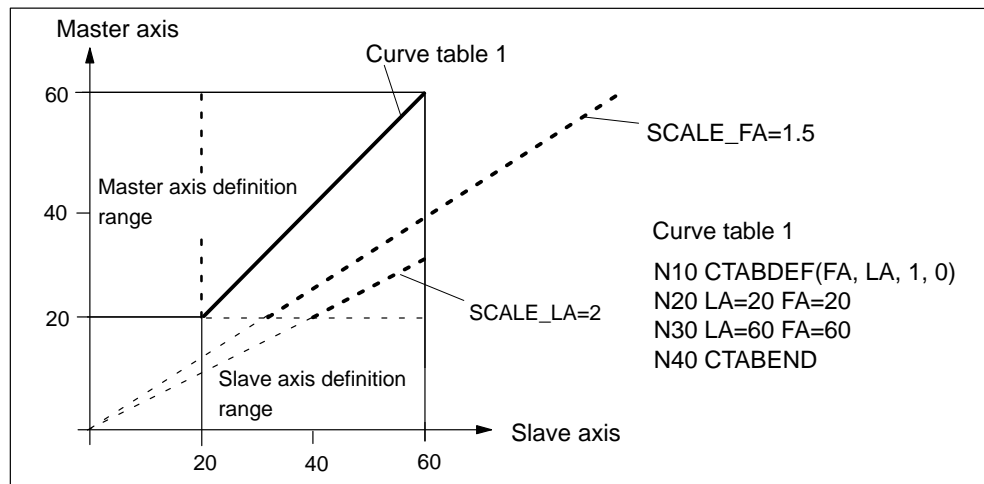


Figure 9-20 Example of scaling of master and slave axis positions

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]` ; "Slave axis position offset" parameter

`$$SA_LEAD_SCALE_IN_POS[FA]` ; "Scaling of slave axis position" parameter

`$$SA_LEAD_OFFSET_OUT_POS[FA]` ; "Master axis position offset" parameter

`$$SA_LEAD_SCALE_OUT_POS[FA]` ; "Scaling of master axis position" parameter

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.

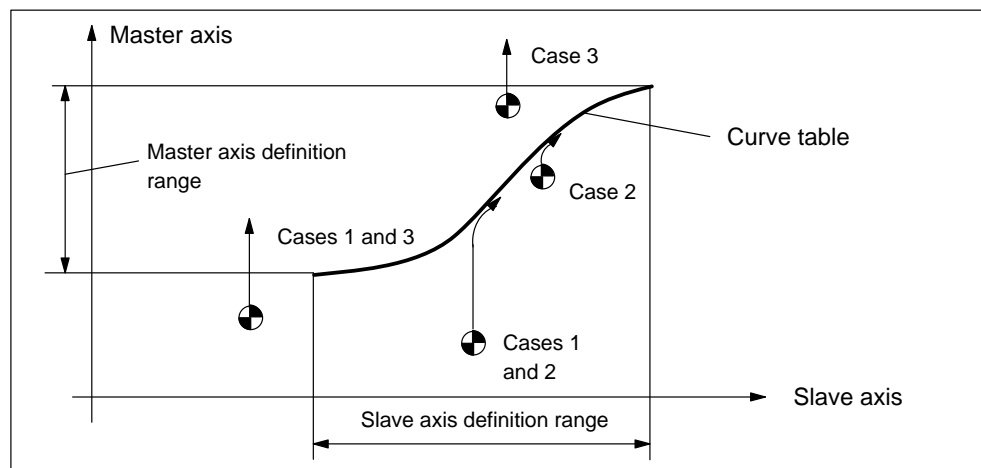


Figure 9-21 Example of synchronization

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 “Axis signals”, DBX116.0)
- Coarse synchronism (User DB “Axis signals”, DBX116.1)

Parameters	Value/Meaning	Unit
Threshold value for coarse synchronism	1 (default) 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) 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 10.22).

9.13.4 Overlaid motion in synchronized actions

General

You can start an overlaid motion in the action part of synchronized actions (see Section 10.22) 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 FM357-LX version with product version 2 and later.

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) 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) 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) 0 to 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 until the probe switches to FALSE. The compensation is visible only in the machine coordinate system.

9.14 Measurement

General

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 μs on the rising edge and 150 μs on the falling edge of the probe. The measurement uncertainty depends on this delay time and the approach speed to the sensor probe.

Connection of probe

The FM 357 is equipped with on-board inputs for connection of the probe (see Section 4.7):

	Connection
Measuring pulse input 1	X1 Pin 17
Measuring pulse input 2	X1 Pin 18
Switching response of inputs:	0 V (non-deflected state) 24 V (deflected state)

Programming of measurement 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 (product version 2 and later):

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
\$AA_MM1...4[axis] ; Measured value of trigger event 1 to 4 in machine coordinate system
\$AA_MW1...4[axis] ; Measured value of trigger event 1 to 4 in workpiece coordinate system
\$A_PROBE[n] ; Probe status,
n = Number of measuring input
\$AA_MEA ACT[axis] ; Status of axial measurement

Interface signals

The status of the probe inputs is displayed via signals:

- Interface signal "Probe 1 actuated" (user DB, "NC signals", DBX30.1)
- Interface signal "Probe 2 actuated" (user DB, "NC signals", DBX30.2)
- Interface signal "Measurement active" (user DB, "Axis signals", DBX22.3)

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.
- Process measured value

9.15 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-LX with product version 2 and later.

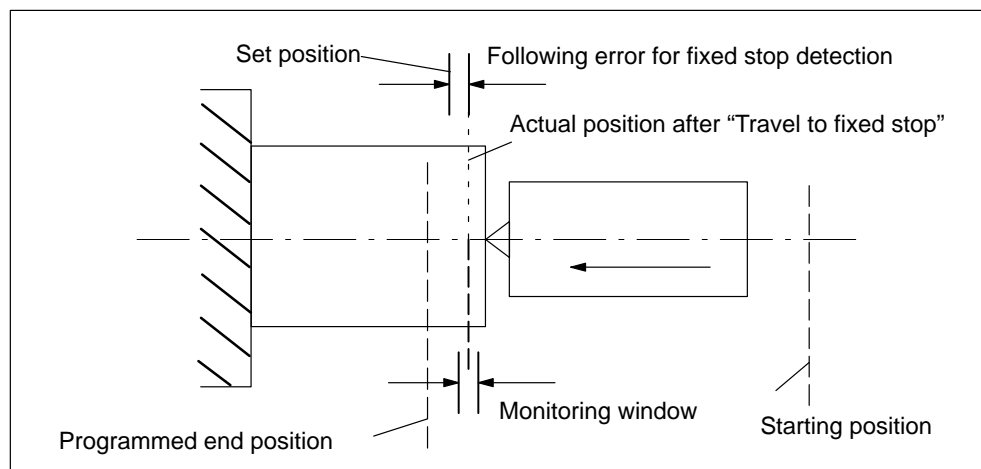


Figure 9-22 Example of travel to fixed stop

Requirements of the drive

The “Travel to fixed stop” function can be used **only** for axes with analog drives.

Requirement:

- 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 9-21 Parameters for travel to fixed stop, continued

Parameters	Value/Meaning	Unit
Clamping torque	5 (default) 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	5 (default) 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) Error message is output No Error message is not output	—

9.15.2 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 is therefore capable of presetting a variable clamping torque.

Setpoints must be injected via terminals 24/20.

Hardware connection

Figure 9-23 shows the hardware connections between the FM357, signal module (SM) and SIMODRIVE 611-A (FDD).

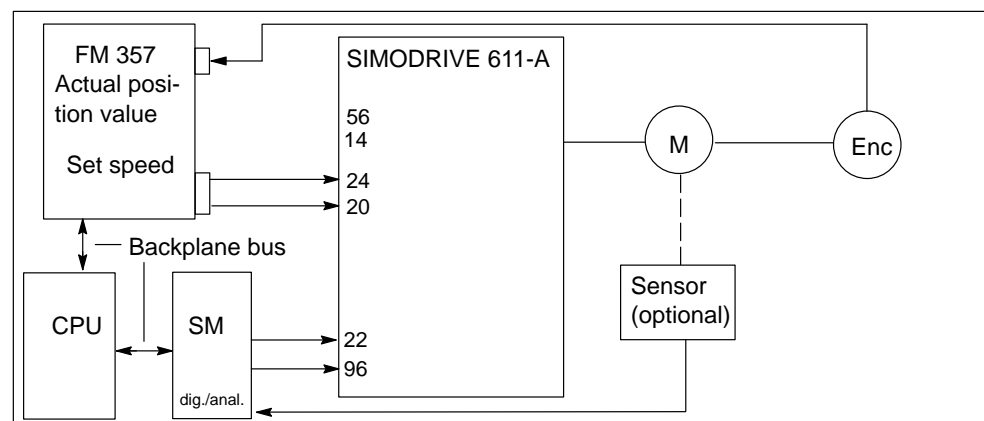


Figure 9-23 Hardware connections between FM 357, signal module and SIMODRIVE 611-A (FDD)

9.15.3 Functional sequence

Selection

The function is activated by instruction FXS[axis]=1. The FM 357 sends interface signal "Travel to fixed stop active" (user DB, "Axis signals", DBX22.4) to the CPU.

The CPU must then activate the current limitation on the drive (terminal 96) and send acknowledgement "Enable travel to fixed stop" (user DB, "Axis signals", DBX43.1) to the FM module.

The FM 357 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, "Axis signals", DBX41.2) set.

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 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, "Axis signals", DBX22.5) to the CPU.

If 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 outputs interface signal "Acknowledge fixed stop reached" (user DB, "Axis signals", DBX41.1). The FM 357 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, “Axis signals”, DBX22.4) reset.

The CPU must then deactivate the current limitation (terminal 96).

It then acknowledges by way of interface signal “Enable travel to fixed stop” (user DB, “Axis signals”, DBX43.1). The block is ended in the FM 357 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 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, “Axis signals”, DBX22.4)
- “Fixed stop reached” (user DB, “Axis signals”, DBX22.5)

If current-controlled operation is activated, 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:

- “Enable travel to fixed stop” (user DB, “Axis signals”, DBX43.1)
- “Acknowledge fixed stop reached” (user DB, “Axis signals”, DBX41.1)

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" with SIMODRIVE 611-A.

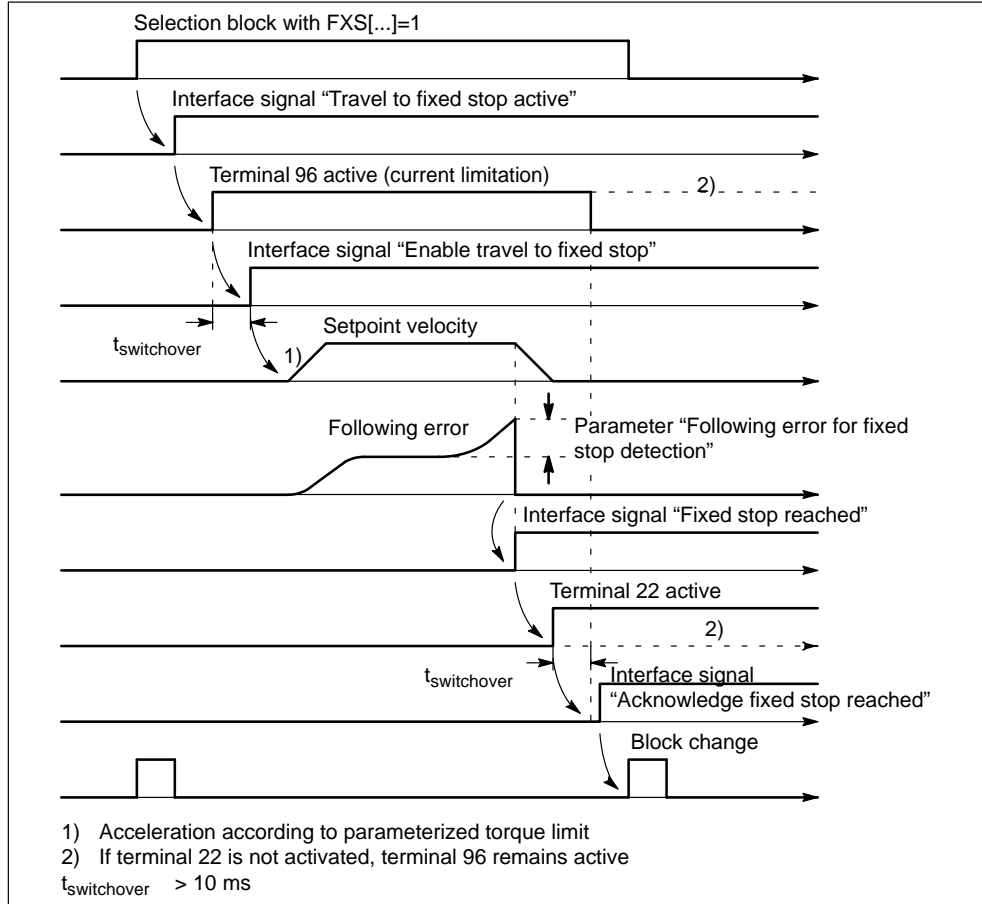


Figure 9-24 Diagram for "Fixed stop reached" with SIMODRIVE 611-A

The following diagram shows the sequence of events for selection block with FXS[...]=1 and "Fixed stop not reached" with SIMODRIVE 611-A.

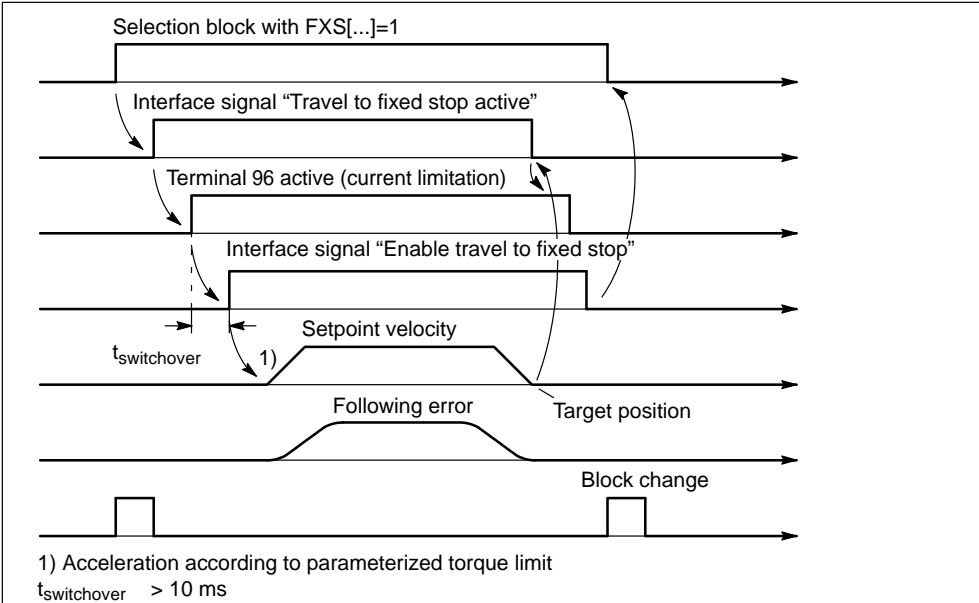


Figure 9-25 Diagram showing "Fixed stop not reached" with SIMODRIVE 611-A

The following diagrams shows function deselection with FXS[...]=0 on a SIMODRIVE 611-A.

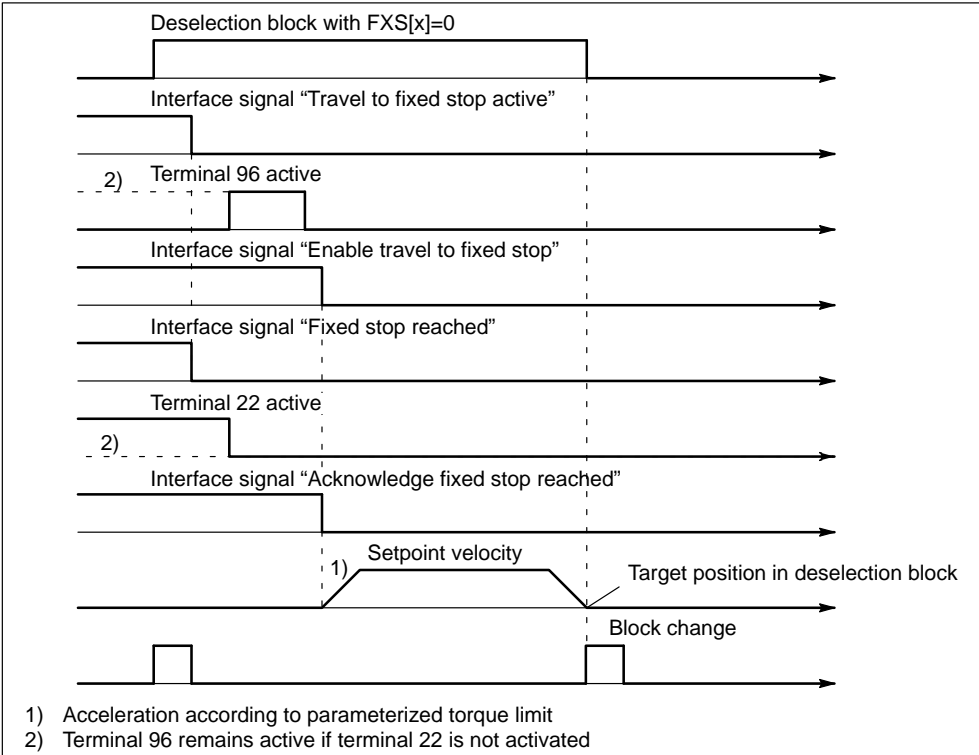


Figure 9-26 Diagram showing deselection "Fixed stop reached" with SIMODRIVE 611-A

9.15.4 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 an NC 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, "Axis signals", DBX12.1) 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.

Note

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.

9.16 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.

The function is available with product version 2 and later.

Parameters

The following parameters are relevant for the EMERGENCY STOP function:

Parameters	Value/meaning	Unit
EMERGENCY STOP braking time	0.05 (default) 0.02 to 1 000	[s]
Cutout delay controller enable EMERGENCY STOP	0.1 (default) 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:

NC Stop (STP) = 1 (user DB "NC signals", DBX11.1)
 Controller enable (REG) = 0 (user DB "Axis signals", DBX12.1)
 Follow-up mode (NFG) = 1 (user DB "Axis signals", DBX12.4)

The axis signals must be set for all axes which need to be braked.

The NC program is stopped with NC Stop in the FM 357 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:

- NC Stop (STP) = 0 (user DB “NC signals”, DBX11.1)
- Controller enable (REG) = 1 (user DB “Axis signals”, DBX12.1)
- Follow-up mode (NFG) = 0 (user DB “Axis signals”, DBX12.4)

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 NC Start in “Automatic” mode. After NC Start, the axes first approach the interruption point. The NC program then continues from this position.

Figure 9-27 shows a possible sequence of operations.

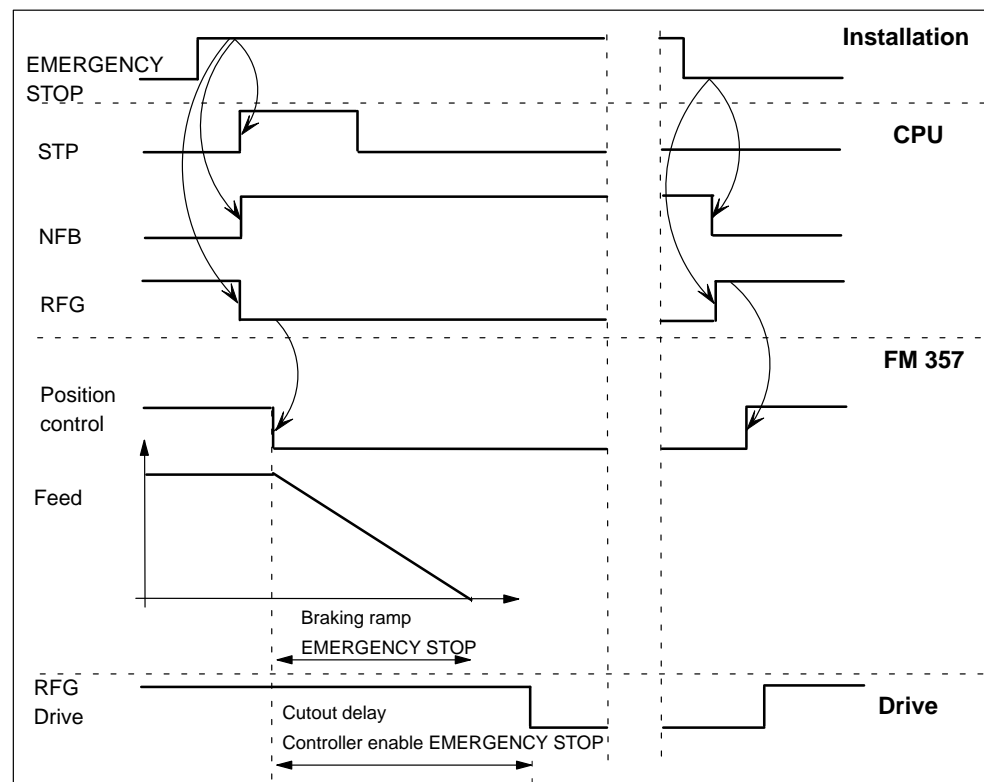


Figure 9-27 EMERGENCY STOP sequence

Error messages

When the controller enable signal is cancelled while axes are moving, error message “Controller enable reset during motion” is output.

The error can be reset with CANCEL or NC Start.



NC Programming

10

Overview

You can program the instructions 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" tool.

Note

All units in this document are given in the **metric** basic system.

Section 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)	10-29
10.5	Programming axis movements	10-30
10.6	Splines (ASPLINE, CSPLINE, BSPLINE)	10-40
10.7	Path response	10-46
10.8	Dwell time (G4)	10-54
10.9	Coupled motion (TRAILON, TRAILOF)	10-54
10.10	Measurement (MEAS, MEAW)	10-56
10.11	Travel to fixed stop (FXST, FXSW, FXS)	10-60
10.12	Stop preprocessor (STOPRE)	10-62
10.13	Working area limitation (G25, G26, WALIMON, WALIMOF)	10-62
10.14	M functions	10-64
10.15	H functions	10-66
10.16	Tool offset values (T functions)	10-67
10.17	R parameters (arithmetic parameters)	10-69
10.18	System variables: \$P_, \$A_, \$AC_, \$AA_	10-72
10.19	Program jumps (GOTOF, GOTOB, LABEL, IF)	10-78
10.20	Subroutine technology (L, P, RET)	10-80
10.21	Asynchronous subroutines (ASUB)	10-83
10.22	Synchronized actions	10-87
10.23	Oscillation	10-104
10.24	Master value coupling	10-108
10.25	Speed feedforward control (FFWON, FFWOF)	10-112
10.26	Overview of statements	10-113

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.

You can display the percentage of free system resources by selecting menu commands **Target System ► FM Properties...** in the "Parameterize FM 357" tool.

10.1.1 Program structure and program name

Structure and contents

The program consists of a sequence of blocks in which the necessary instructions 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 a 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 32 characters; the first 24 characters are displayed
- You cannot use the space or tab character

Example: MPF100

10.1.2 Statements

General

For an overview of all available programming statements, please refer to Section 10.26.

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

Figure 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 10.20, 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 to R99 (REAL type) are available for optional use by the user, e.g. as arithmetic parameters.

System variables

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 format

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.

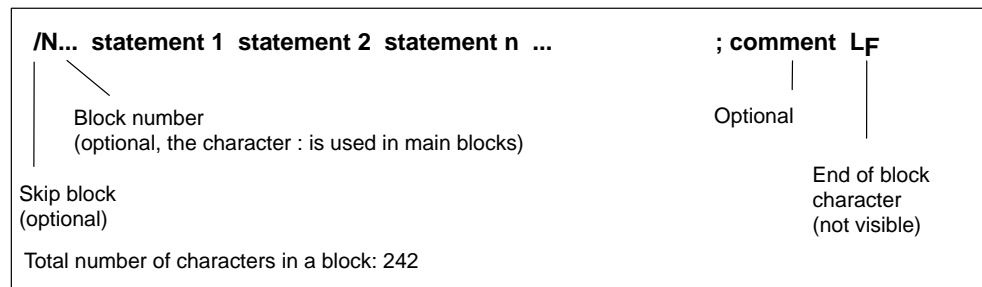


Figure 10-2 Block structure

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... LF
```

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.

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 Klaus Mustermann
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:

If you are using an OP, a corresponding display must be configured.
Messages are not displayed by the “Parameterize FM 357” tool.

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.

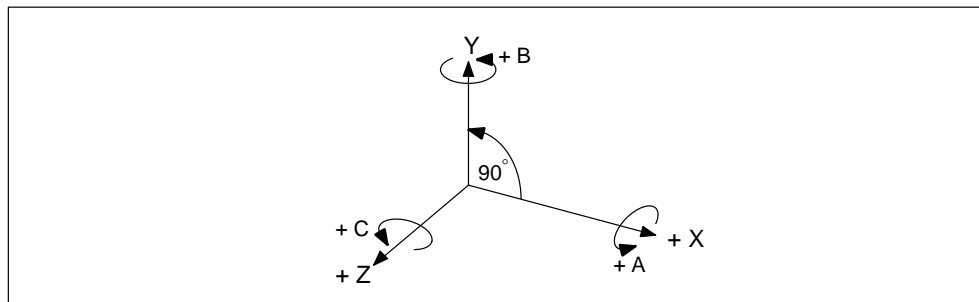


Figure 10-3 Definition of axis directions

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.

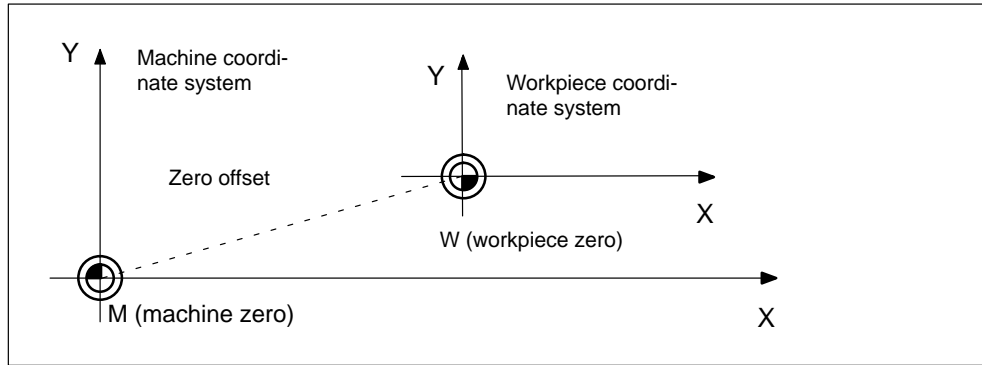


Figure 10-4 Machine and workpiece coordinate systems

10.2.2 Axis types

General

The FM 357 uses the following different types of axis:

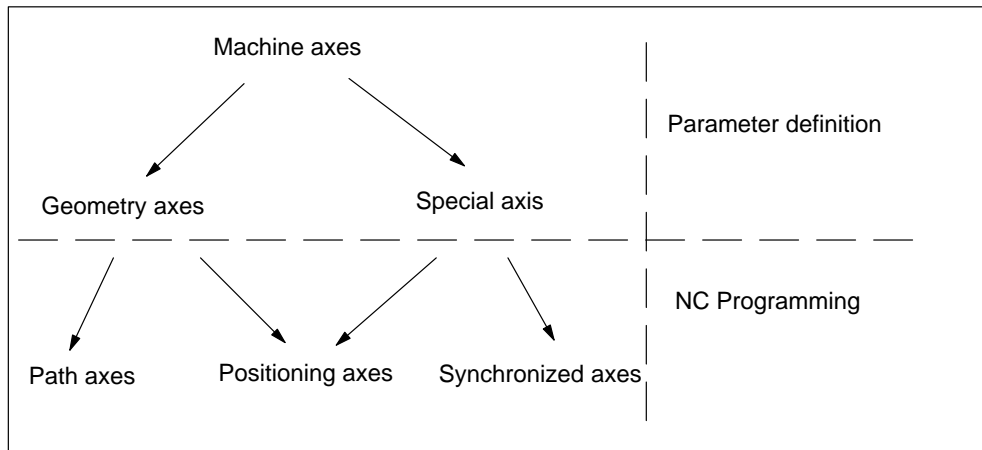


Figure 10-5 Relationship between axis types

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:

± 0.001 to 10^8 mm or ± 0.0001 to 10^8 inches

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

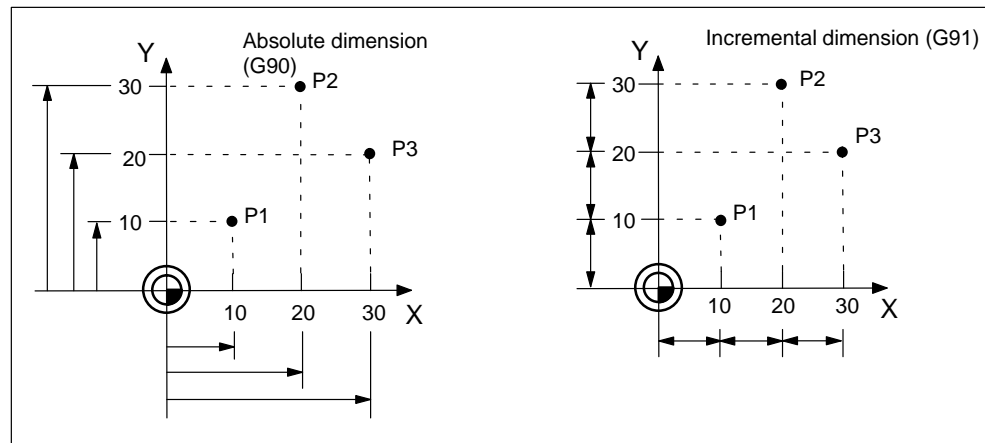


Figure 10-6 Absolute and incremental dimensioning

Absolute dimension 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 dimension 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 dimension for rotary axes (DC, ACP, ACN)

General

Special statements for defined approach conditions are provided for rotary axes (traversing range 0 to 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

...

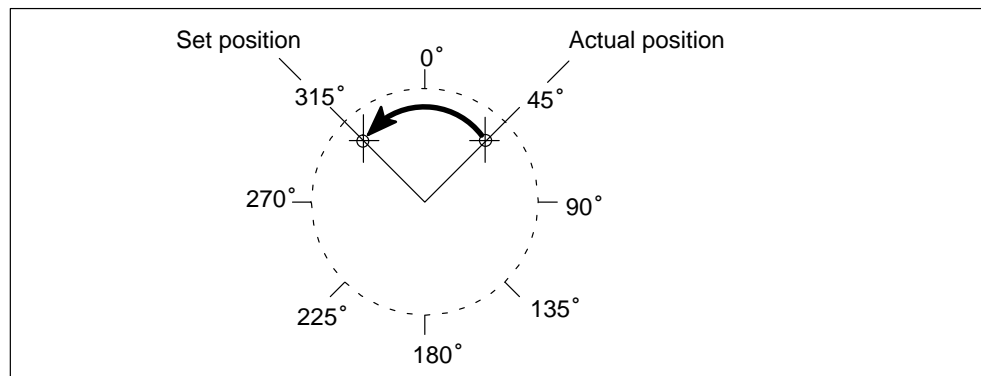


Figure 10-7 Move rotary axis across shortest path

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

...

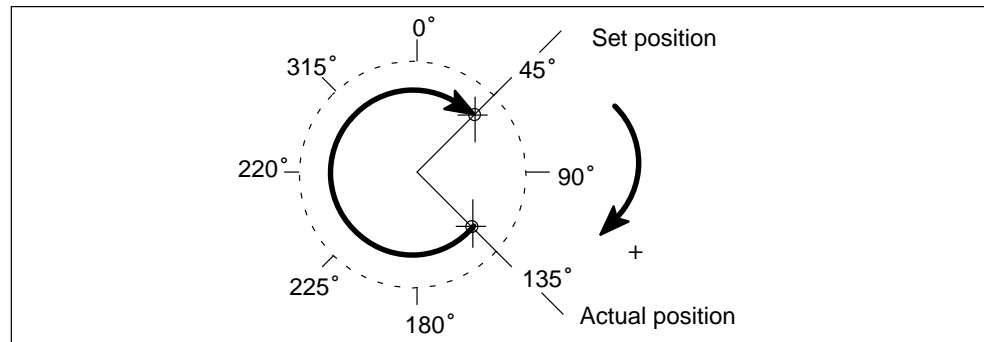


Figure 10-8 Move rotary axis in positive direction to absolute position

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

...

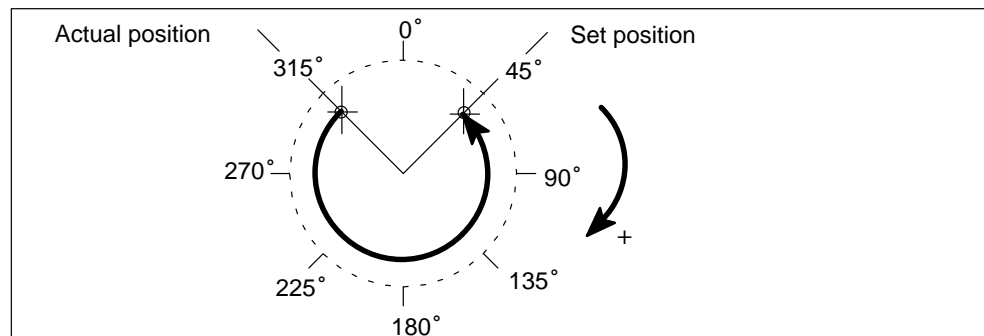


Figure 10-9 Move rotary axis in negative direction to absolute position

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 Programming a polar coordinate (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 to 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

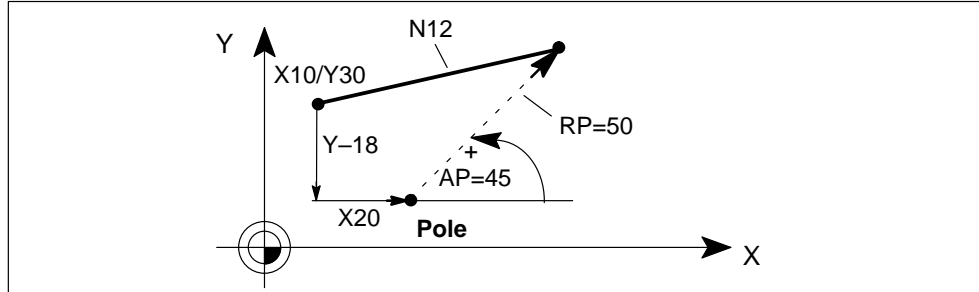


Figure 10-10 Programming G110

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

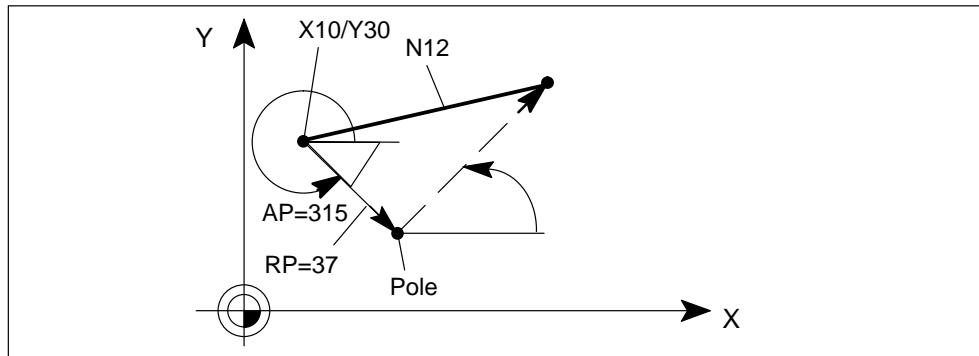


Figure 10-11 Programming G110 (in polar coordinates)

Example 3: G111 X... Y...

N10 G111 X20 Y18 ; Pole

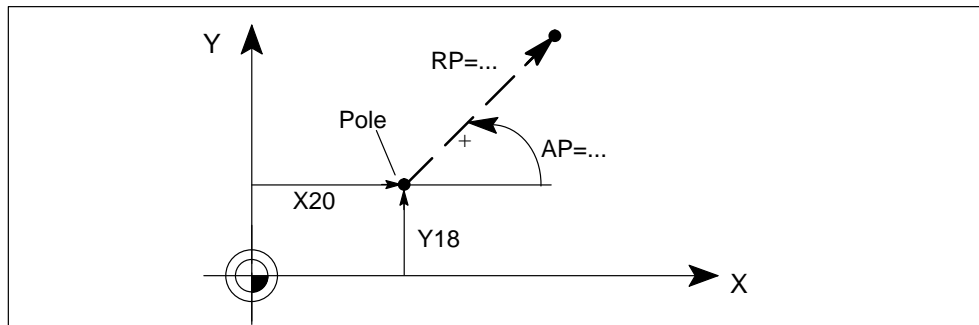


Figure 10-12 Programming G111

10.2.6 Dimensions inch and metric (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.

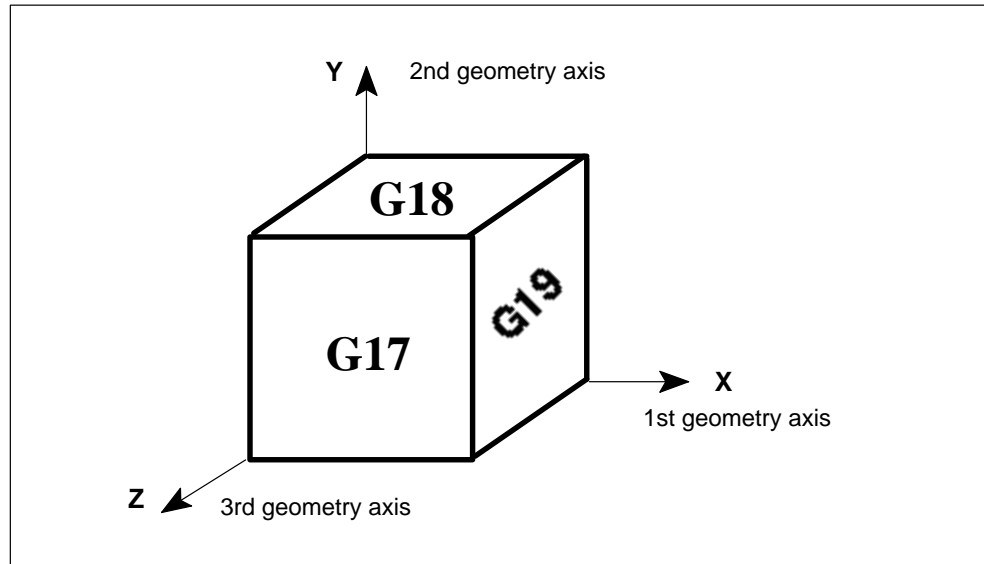


Figure 10-15 Plane and axis assignment

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 offset (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 offset (G54, G55, G56, G57, G500, G53)

General

The values for the settable zero offset are entered in the relevant data field (operator input via the parameterization tool and/or 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:

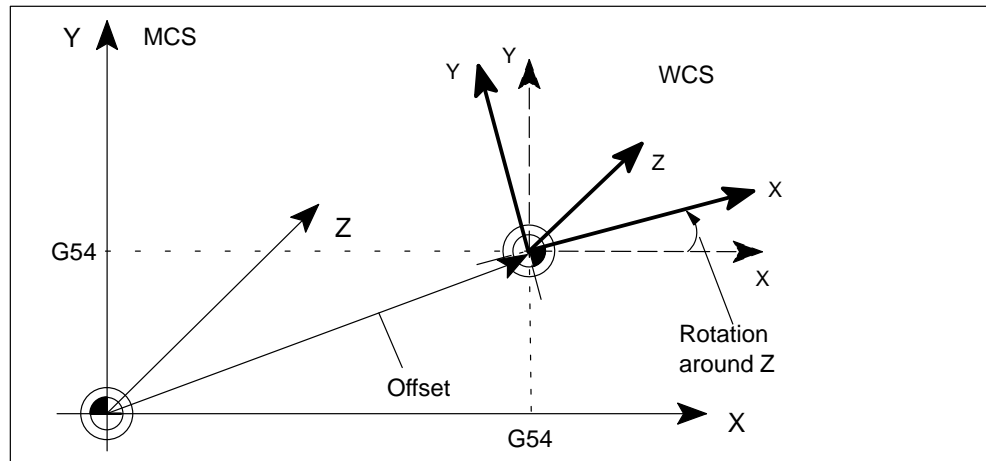


Figure 10-16 Settable zero offset G54 (shift and rotation)

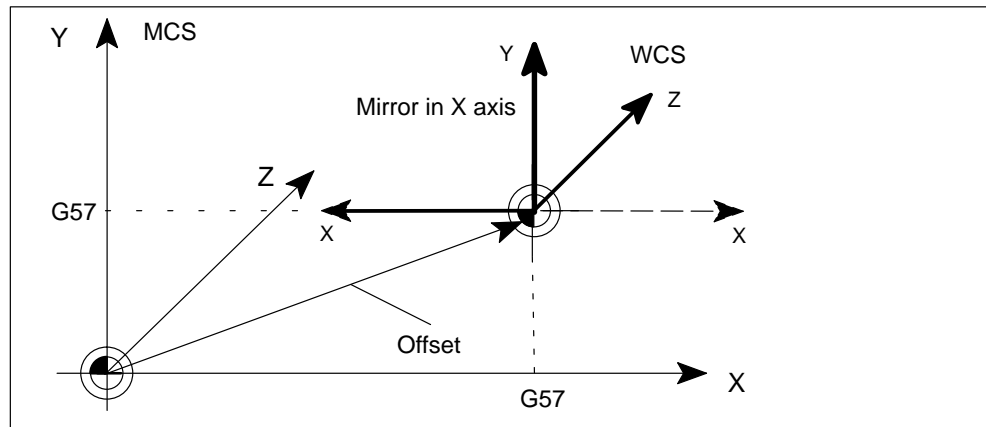


Figure 10-17 Settable zero offset G57 (shift and mirror)

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 offset (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)

Note

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.

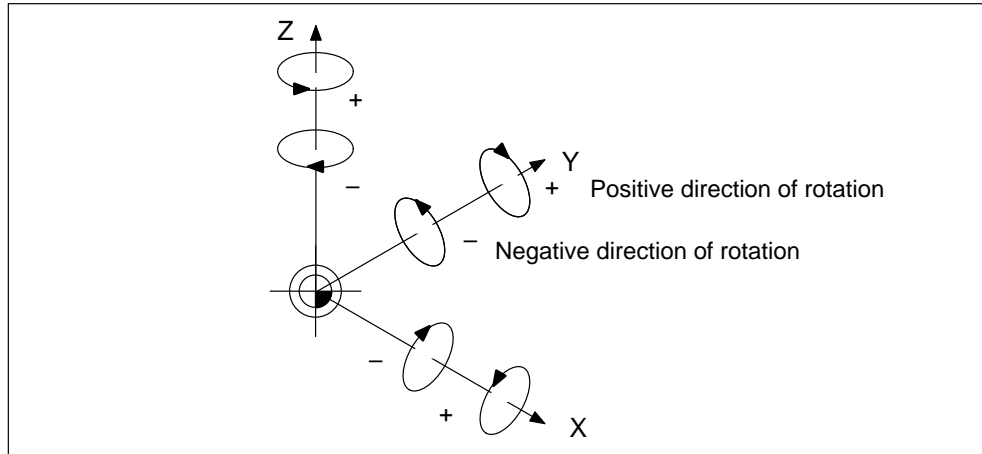


Figure 10-18 Directions of the angle of rotation

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
```

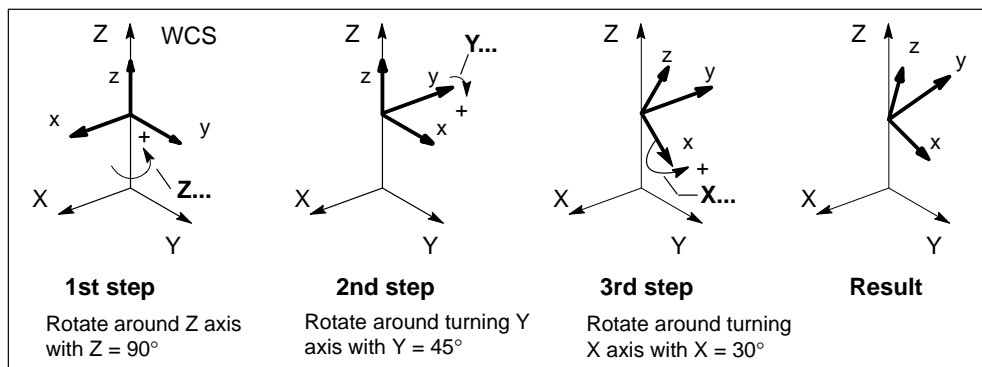


Figure 10-19 Order of rotation for three angle dimensions in **one** block

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.

Note

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=...
```

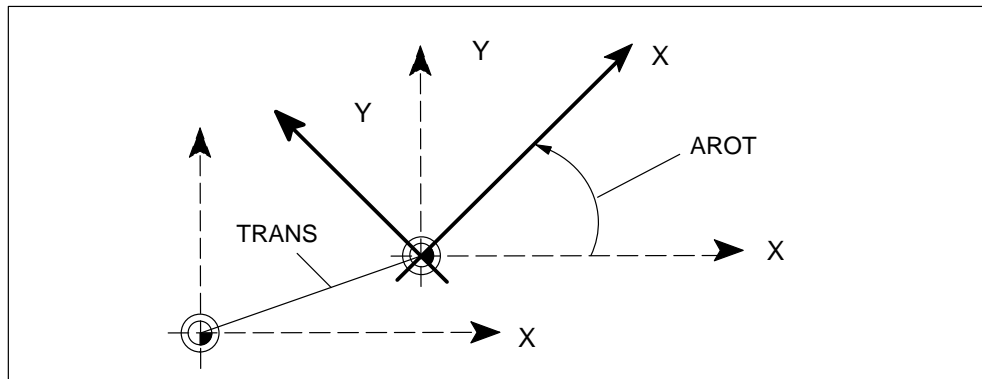


Figure 10-20 RPL – Shift, then rotate

2. **Rotate, then shift:**

```
N10 ROT RPL=...
N11 ATRANS X... Y...
```

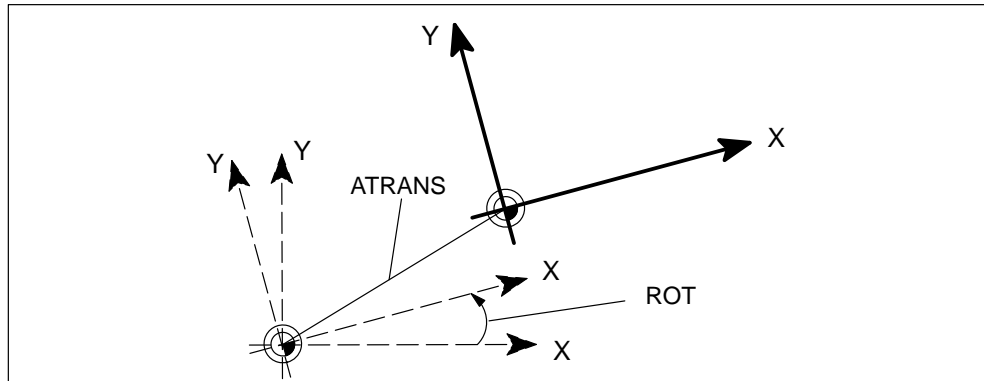


Figure 10-21 RPL – Rotate, then shift

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
```

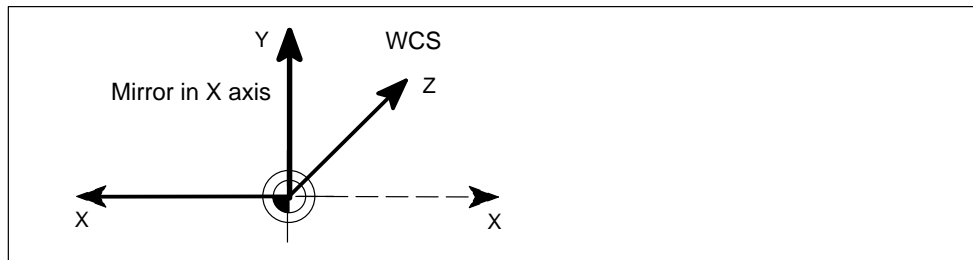


Figure 10-22 Mirror in X axis

10.4 Setting an 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.
```

Note

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)
- Feed interpolation (FNORM, FLIN, FCUB)
- Linear interpolation with rapid traverse (G0)
- Linear interpolation with feed (G1)
- Positioning movements (POS, POSA, WAITP)
- Circular interpolation (G2, G3, I, J, K, CR)

10.5.1 Programming of feedrates (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-LX in product version 2 and later.

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.

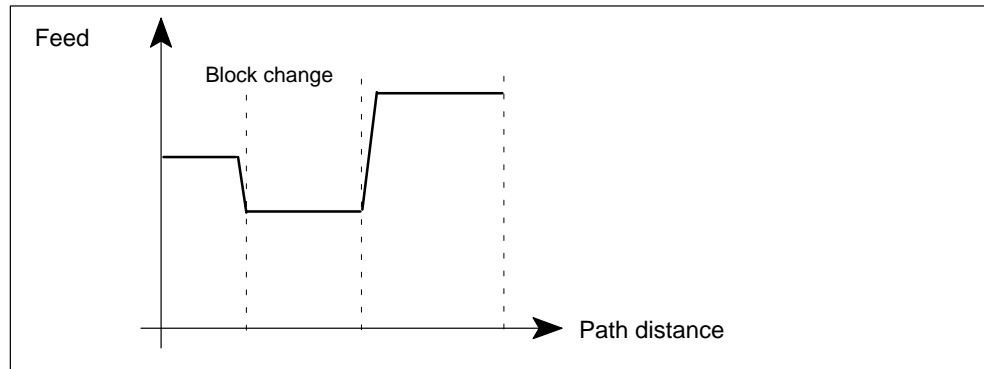


Figure 10-23 Example of constant feed characteristic

FLIN

The feedrate has a linear characteristic from the current feed to the programmed value at the block end.

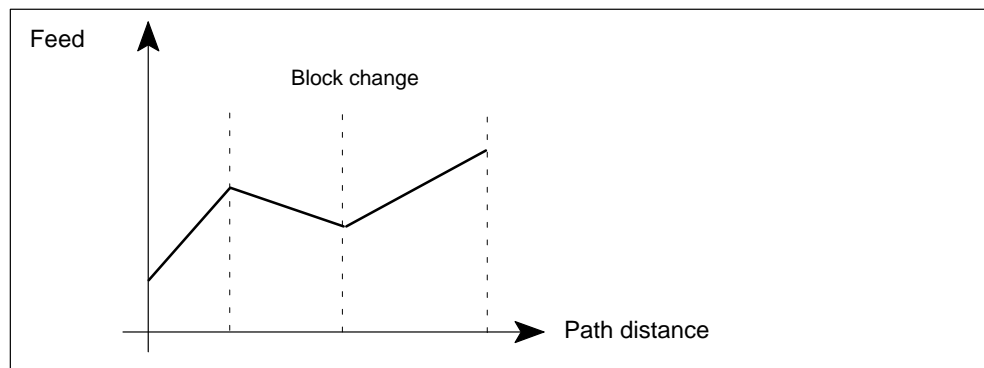


Figure 10-24 Example of linear feed characteristic

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 10.6).

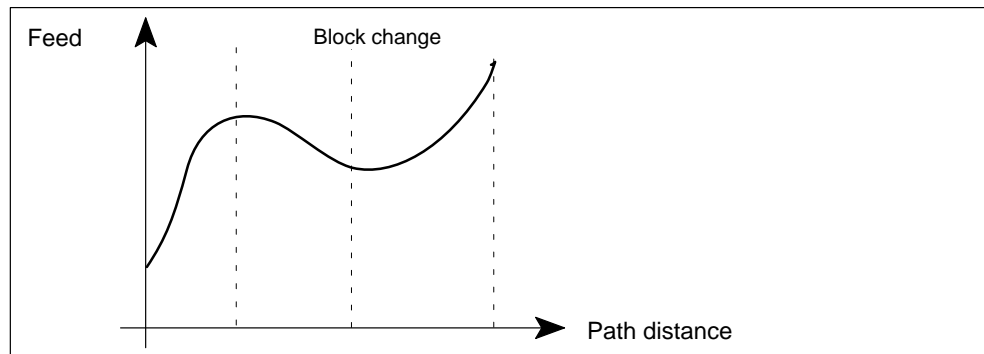


Figure 10-25 Example of cubic feed characteristic

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 FCUP 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

```

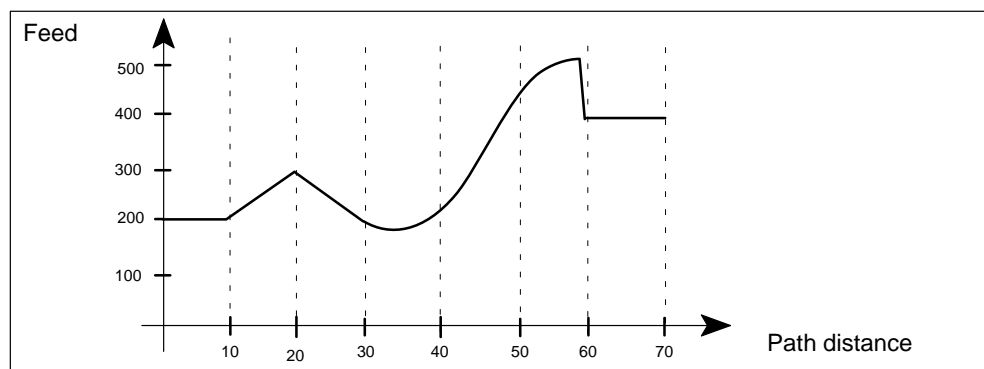


Figure 10-26 Example of feed interpolation

10.5.3 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

...

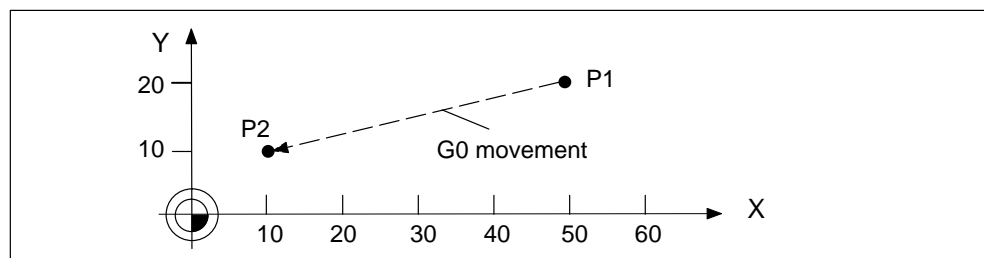


Figure 10-27 Linear interpolation with rapid traverse

10.5.4 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

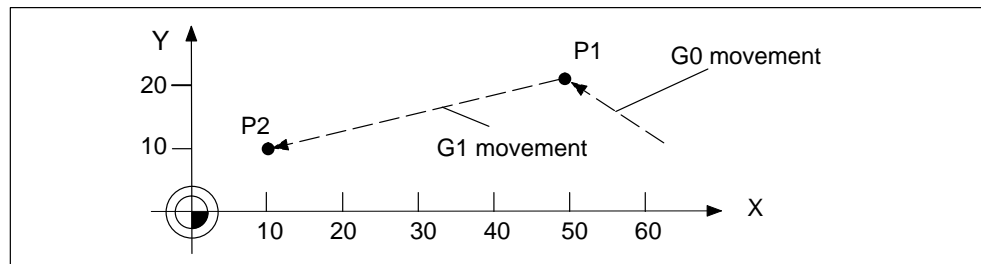


Figure 10-28 Linear interpolation with feed

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.

10.5.5 Positioning motions (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...  
...
```

10.5.6 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 to        ; 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 to the selected plane.

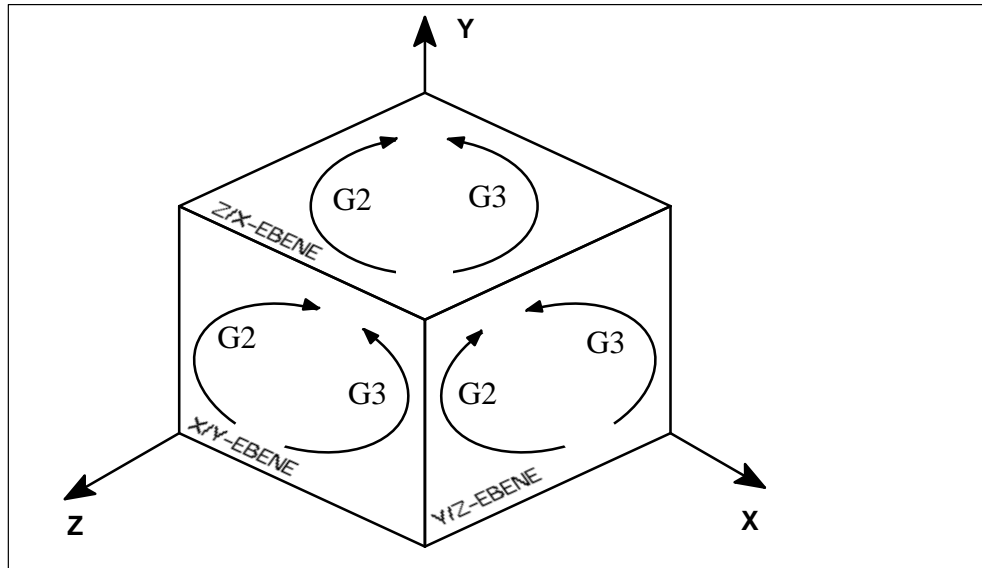


Figure 10-29 Direction of circle rotation in the planes

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

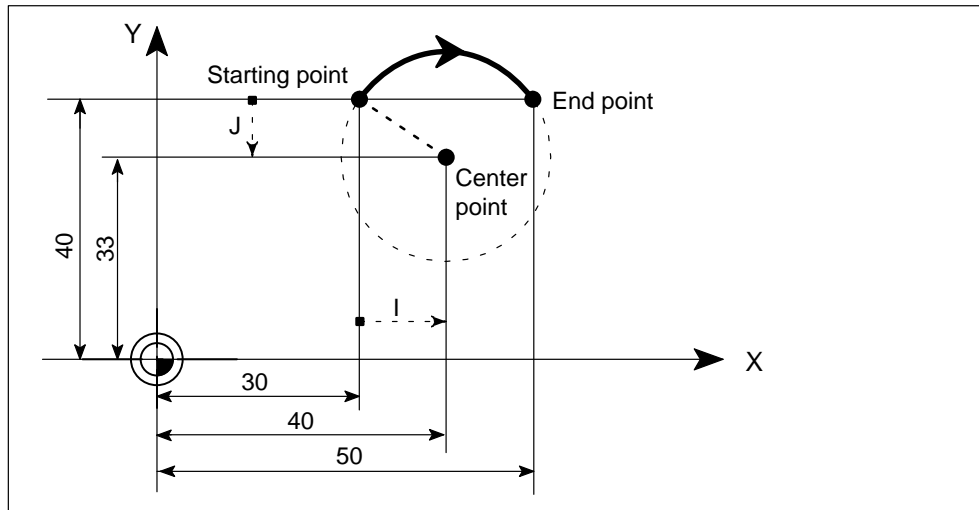


Figure 10-30 Example of center point and end point dimensions

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

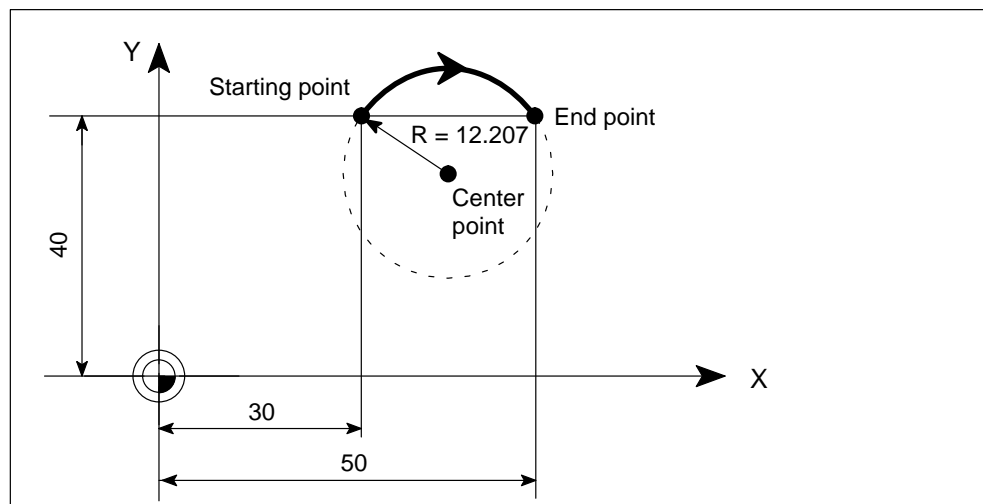


Figure 10-31 Example of end point and radius dimensions

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.

10.6 Spline (ASPLINE, CSPLINE, BSPLINE)

General

The function is available for the FM 357-LX only in product version 2 and later.

Spline interpolation allows you to connect programmed sequences of points through continuous curve transitions.

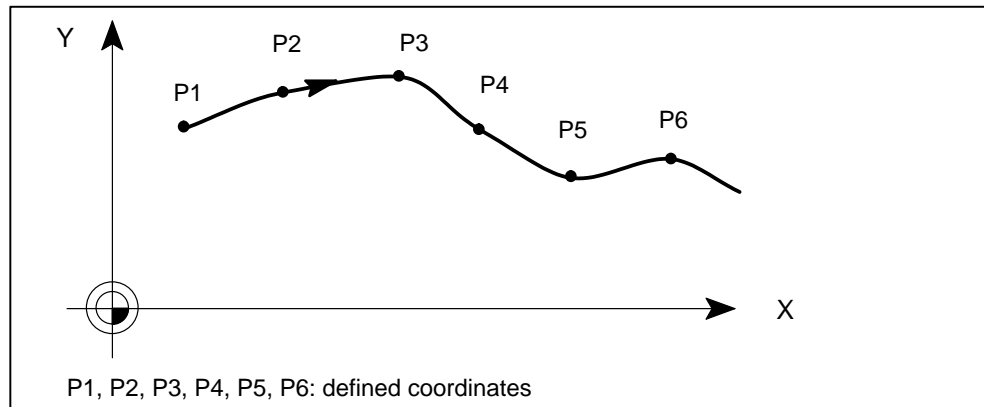


Figure 10-32 Spline interpolation

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
...

```

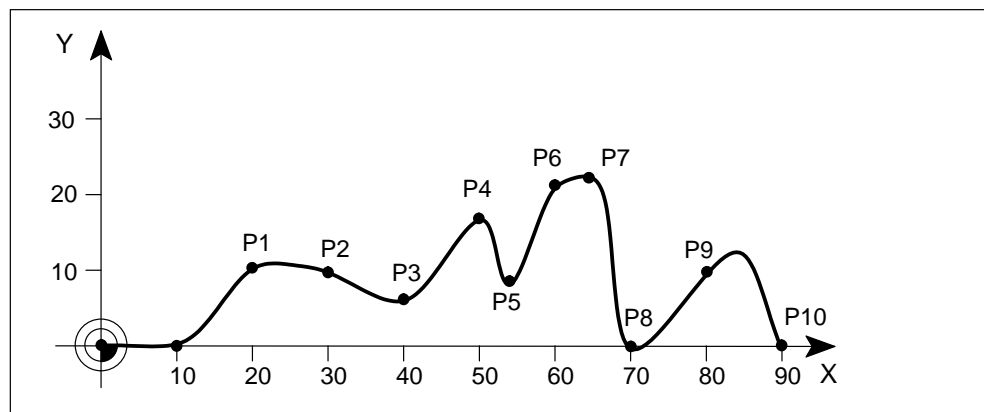


Figure 10-33 ASPLINE

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

```

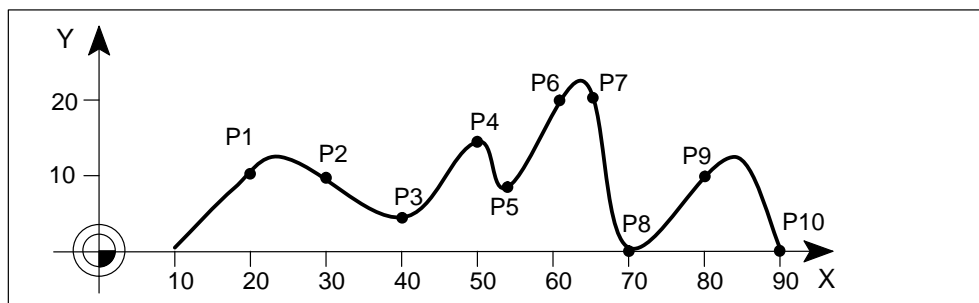


Figure 10-34 CSPLINE

Note

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 ASPLINE or CSPLINE. Once the spline has been started, you cannot make changes.

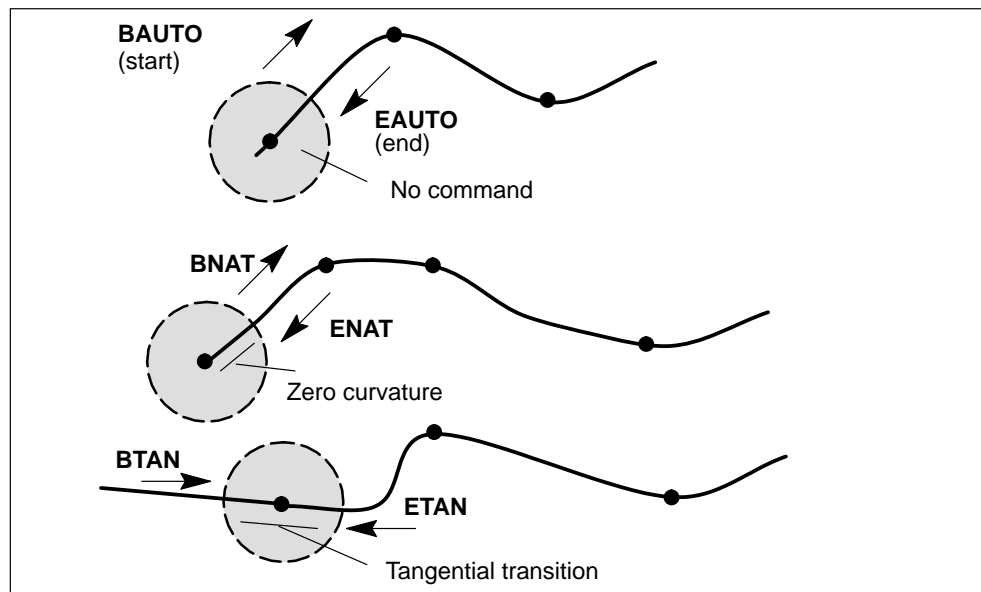


Figure 10-35 Conditions for ASPLINE and CSPLINE

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
...
```

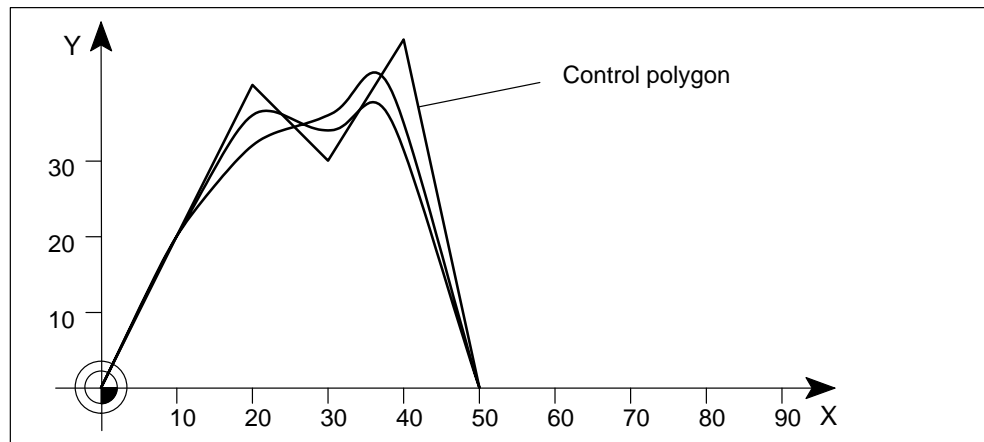


Figure 10-36 BSPLINE, associated control polygon

Spline group SPLINEPATH

All path axes participating in the spline must be included in one statement.

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
```

```
...
```

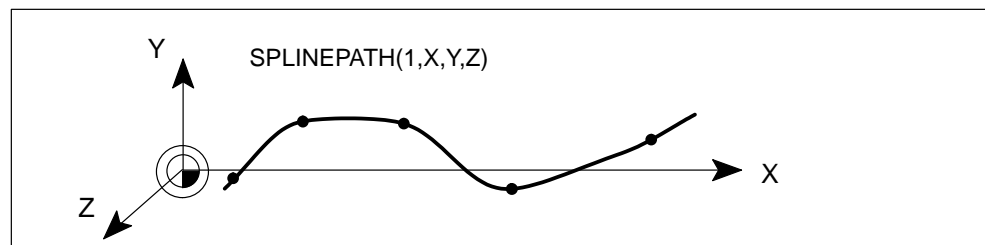


Figure 10-37 Spline grouping, e.g. with three path axes

10.7 Path action

Overview

In this section, you can find information about:

- Exact stop (G9, G60), target range (G601, G602)
- Continuous-path mode (G64, G641, ADIS, ADISPOS)
- Acceleration response (BRISK, SOFT, DRIVE)
- Programmable acceleration (ACC)

10.7.1 Exact stop (G60, G9), target range (G601, G602)

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 is always valid for positioning axes.

Target range fine and coarse can be defined in parameters.

The G command G601 or G602 determines when the block has ended. If you want to execute a rapid traverse with target range coarse, this must be programmed in the G602 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

The G601 and G602 statements 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 program advances to the next block once all axes have reached the "Fine target range" (value in machine data).

Coarse target range G602

The program advances to the next block once all axes have reached the "Coarse target range" (value in machine data).

Programming example

```

N10 G1 G60 G601 X100 Y100 F200
N15 G0 G53 Z0 ; Block change on fine target range
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 9.5.1).

Example:

```

N1 X... Y... G60 G601 ; or G602
N2 Y...
    
```

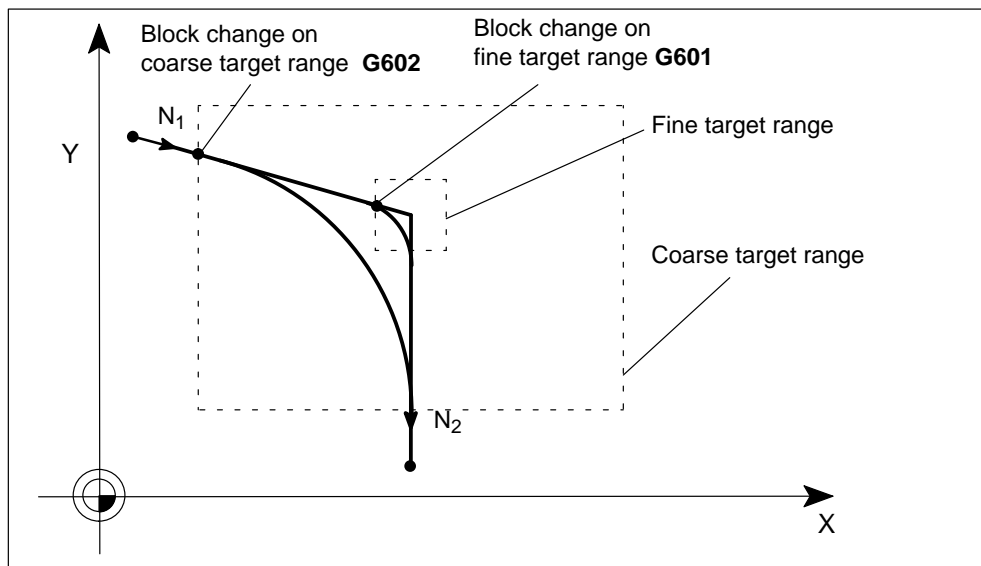


Figure 10-38 Block change depending on the size of the exact stop limit

10.7.2 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.

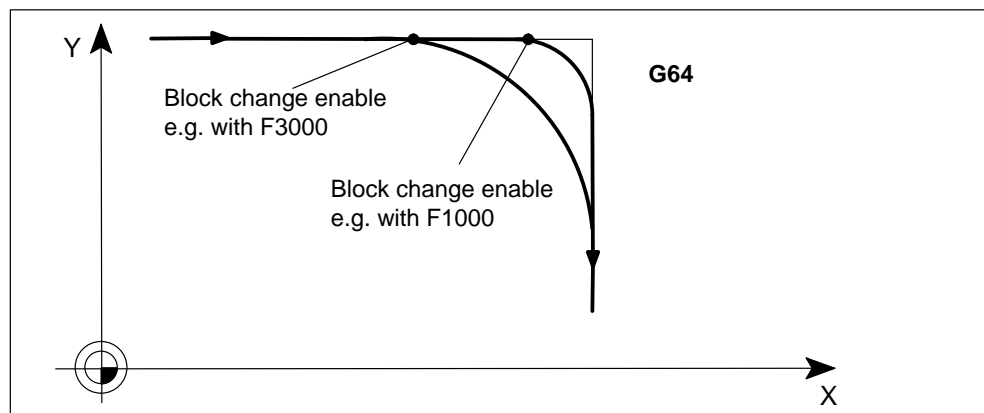


Figure 10-39 Velocity-dependent machining of contour corners with G64

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 G90 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.

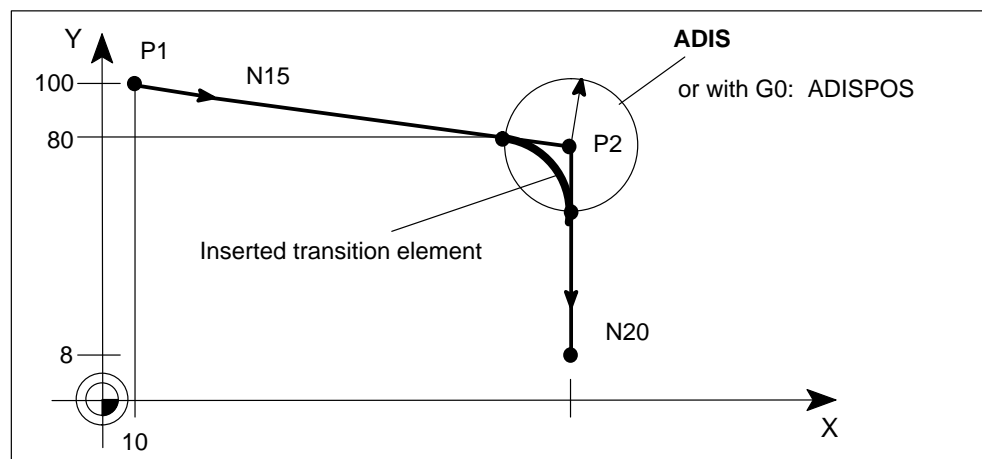


Figure 10-40 Continuous-path mode with rounding clearance: G641 with ADIS/ADISPOS

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 36 % 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.

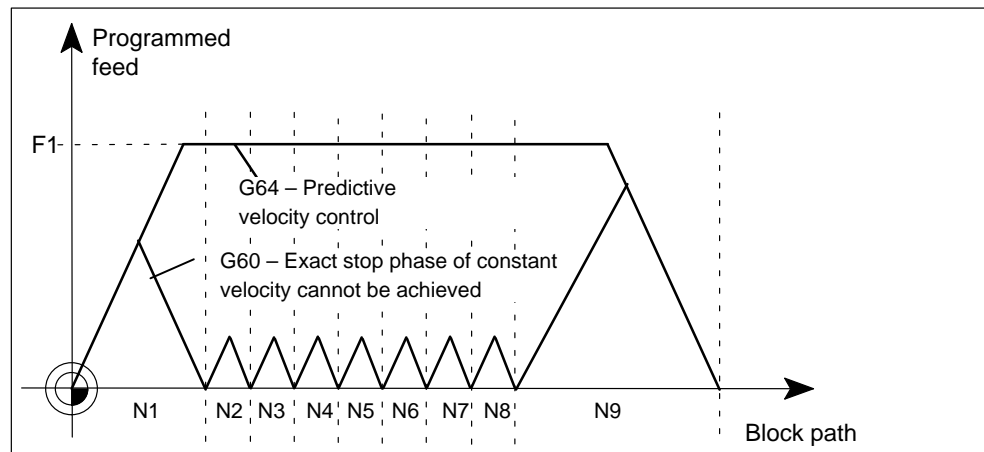


Figure 10-41 Comparison of velocity response in G60 and G64 with short paths

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.

10.7.3 Acceleration patterns (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
```

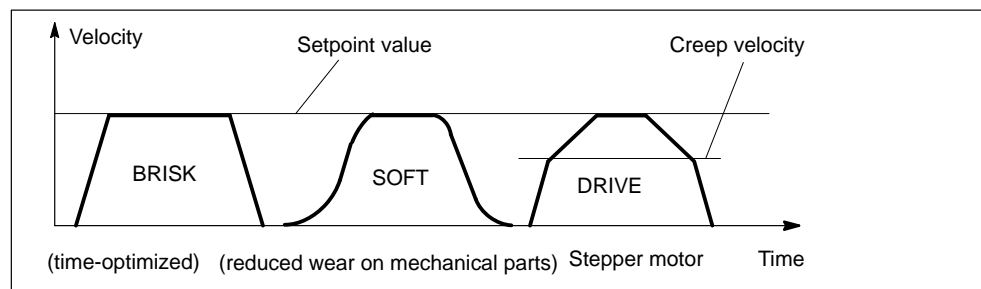


Figure 10-42 Acceleration characteristic with BRISK / SOFT / DRIVE

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.

10.7.4 Programmable acceleration (ACC)

General

The programmable acceleration rate function allows the parameterized axis acceleration pattern to be altered.

Programming

ACC[machine axis name]=... Programmable acceleration

ACC

Statement ACC can be programmed to alter the parameterized axis acceleration rate by a percentage $> 0\% \leq 200\%$.

Example:

N10 ACC[X]=50 ; X axis accelerates at 50% of parameterized axis acceleration rate

The programmable acceleration is active in all interpolation types of operating modes **Automatic** and **MDI**.

The ACC statement has immediate effect and is modal.

With the statement ACC[MachineAxisName]=100, the programmable acceleration is deactivated on RESET or at the end of the program.

10.8 Dwell time (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
...
```

10.9 Motion coupling (TRAILON, TRAILOF)

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.

Programming

TRAILON(slave axis, ; Define and activate a
leading axis, coupling factor) ; coupled-axis grouping, modally

TRAILOF(slave axis, master axis) ; Deactivate the coupling for a
; master axis

Activate TRAILON

The slave axis, the master axis and the coupling factor must be specified.

A slave axis can be activated simultaneously in up to two coupled-axis groupings.

The coupling factor specifies the desired ratio between the paths of the slave axis and the leading axis.

$$\text{Coupling factor} = \frac{\text{Path of leading axis}}{\text{Path of slave axis}}$$

If a coupling factor is not specified, a coupling factor of 1 is automatically valid.

The factor is entered as a decimal fraction. A negative value causes the leading axis and the slave axes to move in opposite directions.

Deactivate TRAILOF

The slave axis and the master axis must be specified.

This statement always causes an exact stop at the end of the block.

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
...

```

Note

For a further description of motion coupling, please see Section 9.13.1.

10.10 Measurement

Overview

In this section, you can find information about:

- Block-specific measurement (MEAS, MEAW)
- Axial measurement (MEASA, MEAWA)

10.10.1 Block-specific measurement (MEAS, MEAW)

General

In block-specific measurement mode, the positions of all axes programmed in the block as the probe switches are acquired and stored. 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)

General

The axial measurement function is available with product version 2 and higher.

Several measurement jobs for different axes can be programmed simultaneously in the same block. Up to four trigger events and assigned measurement values can be acquired for each job.

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

Mode: 0 Abort measurement job (use in synchronized actions)
1 Reserved
2 Activate measurement job, trigger events sequentially

TE_1 to 4: (trigger events 1 to 4)
1 rising edge probe 1
-1 falling edge probe 1
2 rising edge probe 2
-2 falling edge probe 2

\$AA_MM1 to 4[axis] ; Measured value of trigger event 1 to 4
in machine coordinate system

\$AA_MW1 to 4[axis] ; Measured value of trigger event 1 to 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: Meas. job conditions for axis not fulfilled
1: Meas. job conditions for axis fulfilled

\$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.

Trigger events are analysed in the order in which the events are programmed. The same trigger event may not be programmed more than once in the same job.

Only one trigger event can be acquired in the servo cycle. The interval between two trigger events must therefore be greater than 2 * 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 to 4[axis] and \$AA_MW1 to 4[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_MEA ACT[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-LX) (see Section 10.22).

In this case, variables \$AA_MW1 to 4[axis] and \$AC_MEA[n] are not available.

Programming example

```

...
N10 MEASA[X]=(2,1,-1) G1 F100 X100      ; Measurement job for X axis:
                                           ; Trigger event 1: Positive edge
                                           ; Trigger event 2: Negative edge
                                           ; Probe 1
                                           ; Abort motion and delete
                                           ; distance to go
N20 STOPRE                               ; Preprocessing stop for
                                           ; synchronization
N30 IF $AA_MEA ACT[X]==0 gotof ERROR     ; Check measurement
N40 R10=$AA_MM1[X]                       ; Store measuring position 1
N50 R11=$AA_MM2[X]                       ; Store measuring position 2
...
ERROR: -

```


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 9.15.

The function is available for the FM 357-LX in product version 2 and later .

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.

Note

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 that a block change can take place and the status 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
```

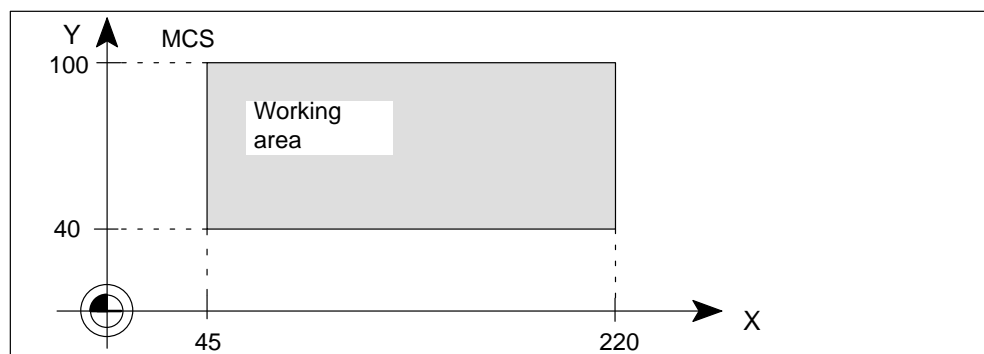


Figure 10-43 Working area limitations G25 and G26

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 to 99

Programming

M... ; M function

Output options for M 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 9.7.

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 to 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 to 99

Programming

H... ; H function

Output options for H 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 9.7.

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 $\pm 99\,999.9999$ Resolution 0.0001

Example:

N10 H10=123.4567 ; H function 10 transfers value 123.4567 to the UP

If you program an H function without specifying a value, the value zero is passed to the user program.

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 to 29

Programming

T1 to T29 ; Select tool T1 to 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.

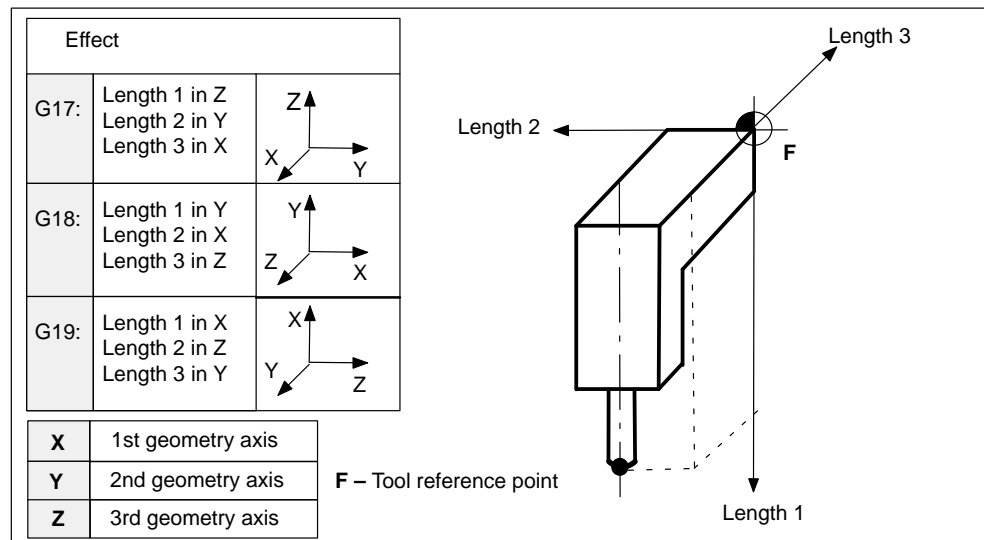


Figure 10-44 Three-dimensional effect of tool length compensation

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 9.7.

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 X10

; Y15 to Y35 Y0 to Y15

; Traversing path:

; X 25 mm

; Y 20 mm

N25 T0 ; Deselect T1

N30 G0 X50 ; Traversal of X axis without application of offset

...

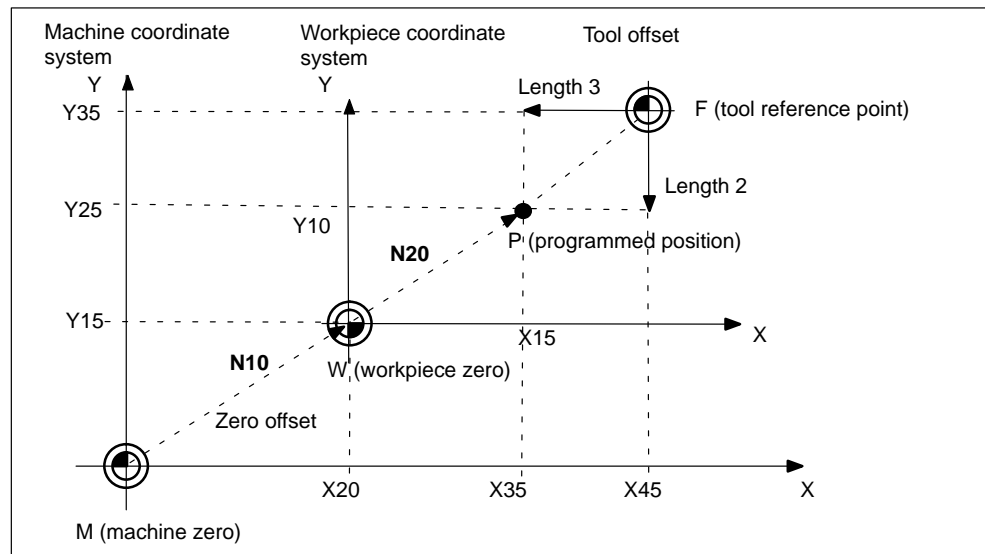


Figure 10-45 Effect of tool offset and zero offset in the G17 plane

10.17 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) are available
```

Value assignment

Values in the following range can be assigned to arithmetic parameters:

0 to $\pm 9999\ 9999$ (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: $\pm 10^{-300}$ to 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 processing are indicated by parentheses. 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)

Indirect programming

The indirect programming method may be used for R parameters.

Example:

N10 R10=7 ; Standard programming

R[R10]=9 ; The value 9 is assigned to parameter R7

Operators/Arithmetic functions

The following operators/arithmetic functions may be used in R parameters.

Table 10-1 Operators and arithmetic functions

	Significance
Operators	
+	Addition
-	Subtraction
*	Multiplication
/	Division
Arithmetic functions	
SIN()	Sinus
COS()	Cosine
TAN()	Tangent
SQRT()	Square root
POT()	Square
ABS()	Absolute value
TRUNC()	Integer component (truncate)

Example:

N10 R13=SIN(25.3) ; sin 25.3°
 N15 R15=SQRT(POT(R1)+POT(R2)) ; Inner parenthesis is removed first
 Significance: $R15 = \sqrt{R1^2 + R2^2}$ L_F

Compare operations

The result of a compare operation can be assigned as a value or used to formulate a jump condition. Complex expressions can also be compared.

Table 10-2 Compare operators

Operators	Significance
= =	Equal
< >	Not equal to
>	Greater than
<	Less than
> =	Greater or equal to
< =	Less than or equal to

The result of a comparison operation is always a BOOL type.

Example:

R2=R1>1 ; R2=TRUE if R1 > 1
 R1<R2+R3
 R6==SIN(POT (R7)) ; if R6 = SIN (R7)²

10.18 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

\$P_ ; **Programmed** data
\$A_, \$AC_ ; **Current** general data
\$AA_ ; **Current axis-specific** 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 variables

The following table provides an overview of all available system variables.

Table 10-3 System variables

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
User variables				
\$Rn	Arithmetic parameters in static memory	r / w	r / w	REAL
\$AC_MARKER[n] n = 0 to 7	Marker variable, counter	r / w	r / w	INT
\$AC_PARAM[n] n = 0 to 49	Arithmetic parameters 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
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
Measurement				
\$AA_MEAAct[axis]	Status of axial measurement 0: Meas. job conditions for axis not fulfilled 1: Meas. job conditions for axis fulfilled	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
\$AA_MM[axis]	Measured value in MCS with MEAS	r	r	REAL
\$AA_MMi[axis]	Measured value in MCS with MEASA i: Trigger event 1 to 4	r	r	REAL

r = read, w = write

Table 10-3 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
\$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 at positioned 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
\$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

r = read, w = write

Table 10-3 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
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 (* the function is available for the FM 357-LX in product version 2 and later)				
\$AC_VACTB	Path velocity in MCS*		r	REAL
\$AC_VACTW	Path velocity in WCS*		r	REAL
\$AC_VC	Additive path feed override*		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_VACTW[axis]	Axis velocity, actual value in WCS		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
Master value coupling				
\$AA_LEAD_TYP[axis]	Type of master value 1: Actual value 2: Setpoint 3: Simulated master value	r	r	INT
\$AA_LEAD_SP[axis]	Position of simulated master value	r	r	REAL
\$AA_LEAD_SV[axis]	Velocity of simulated master value	r	r	REAL
\$AA_LEAD_P[axis]	Position of real master 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

r = read, w = write

Table 10-3 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
\$AA_COUP_ACT[axis]	Type of axis coupling of slave axis 0: Not coupled 3: res. 4: res. 8: Slave axis 16: Master axis	r	r	INT
\$SA_LEAD_OFF-SET_IN_POS[FA]	Slave axis position offset	r / w		REAL
\$SA_LEAD_SCALE_IN_POS [FA]	Scaling for slave axis position	r / w		REAL
\$SA_LEAD_OFF-SET_OUT_POS[FA]	Master axis position offset	r / w		REAL
\$SA_LEAD_SCALE_OUT_POS[FA]	Scaling for master 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_LIMIT[axis]	Limit for overlaid movement 0: Not reached 1: Reached in positive direction 2: Reached in negative direction		r	
CPU variables				
\$A_DBW[0] \$A_DBW[1]	Data word from CPU, FM can read Data word to CPU, FM can write (freely programmable by user)	r w	r w	INT
Motion overlay				
\$AA_OFF[axis]	Overlaid motion		r / w	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
Trace				
\$AA_SCTRACE[axis]	Generation of an IPO event (trigger event)			BOOL

r = read, w = write

Table 10-3 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
States				
\$AC_STAT	Current FM status 1: Aborted 2: Active 4: Interrupted 8: Reset		r	INT
\$AC_PROG	Current status of NC program 1: Program running 2: Program waiting 4: Program stopped 8: Program interrupted		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: Neutral axis 1: Path axis 2: Positioning axis from NC program 3: Positioning axis from synchr. act. 4: Positioning axis from CPU	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
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

r = read, w = write

10.19 Program jumps (GOTOF, GOTOB, 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 forwards
GOTOB LABEL	; Unconditional jump backwards
IF condition GOTOF LABEL	; Conditional jump forwards
IF condition GOTOB LABEL	; Conditional jump backwards
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 32 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.

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

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 ...
```

10.20 Subroutine system (L, P, RET)

General

There is no fundamental difference between a main run 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

L...	; Subroutine call, subroutine name
P...	; Repeat program
M2	; End of subroutine
RET	; End of subroutine

Subroutine name

Address word L... must be used for subroutines. 31 decimal places are available for the value (integers only).

Please note: Leading zeros after the L address are differentiated.

Example:

L123 is not the same as L0123 or L00123 !!

These are 3 different subroutines.

A name can be chosen freely, subject to the following rules:

- The first two characters must be letters
- These can be followed by letters, digits and an underscore, (not spaces or tabs)
- The name may be up to 32 characters in length

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
    
```

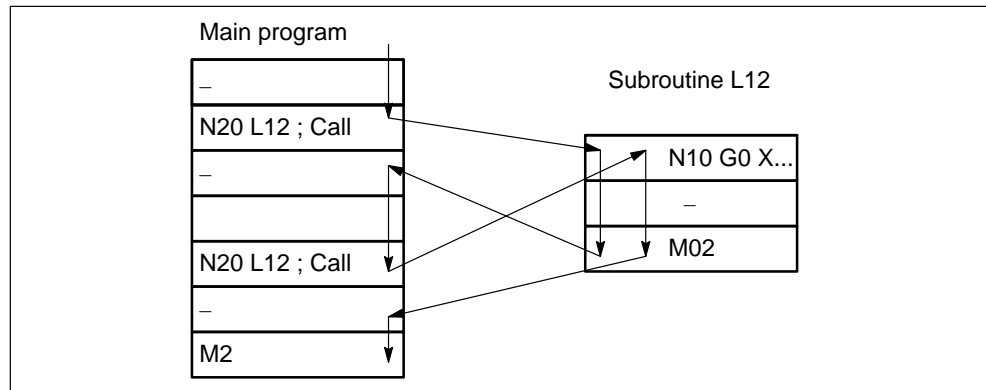


Figure 10-46 Example of a program run with double subroutine call

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.

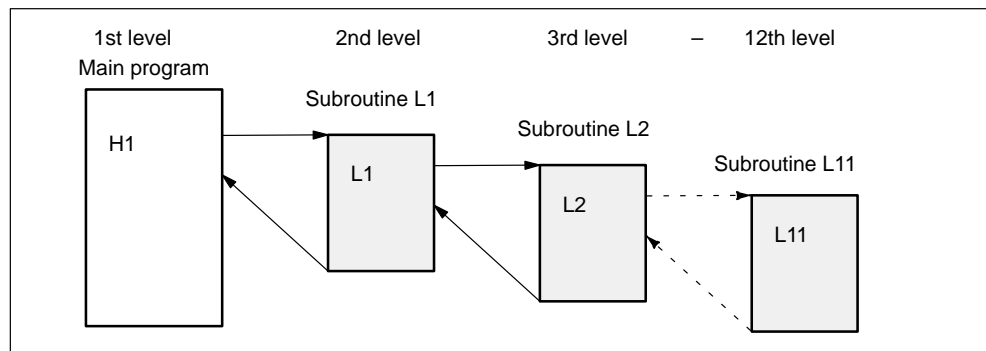


Figure 10-47 Nesting depth

You can also call subroutines in ASUBs. You must keep an appropriate number of levels free for their execution.

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 branch back to the calling program.

Program repetition P...

If a subroutine needs to be processed several times in succession, then the number of required runs 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

10.21 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 FM357 has 4 on-board inputs (inputs 0 to 3) 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=m NAME

SETINT(n) ; Assign a digital input/interrupt no. (n = 1 to 4, 8)
 PRIO = m ; Define priority (m = 1 to 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

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) 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

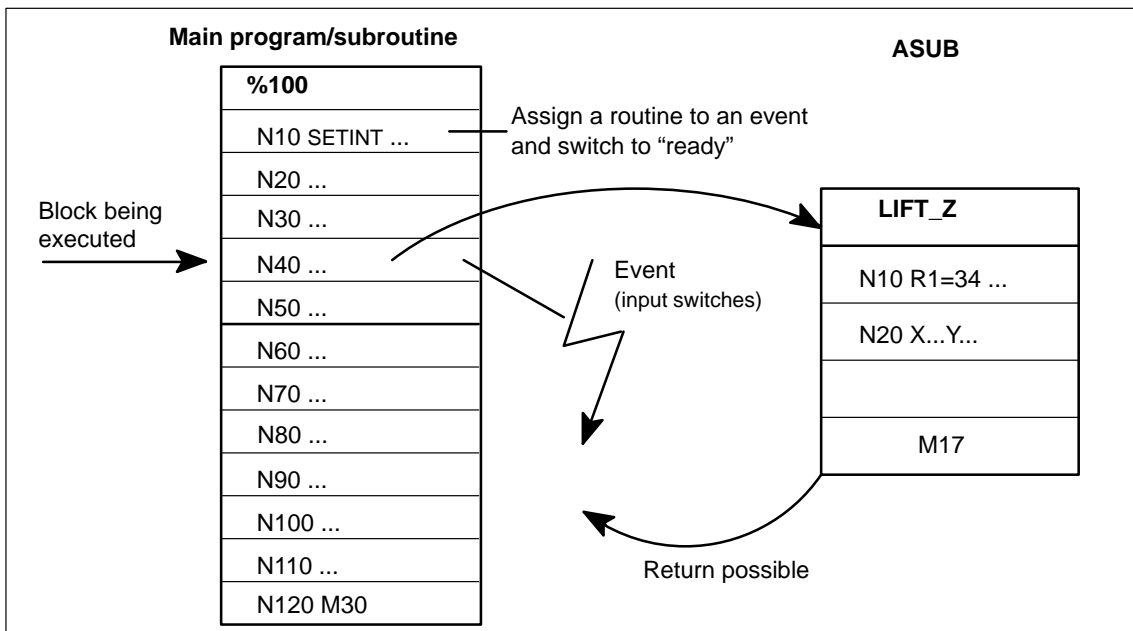


Figure 10-48 Working with ASUBs

10.22 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)
- Scanning/execution frequency
- Condition
- Action

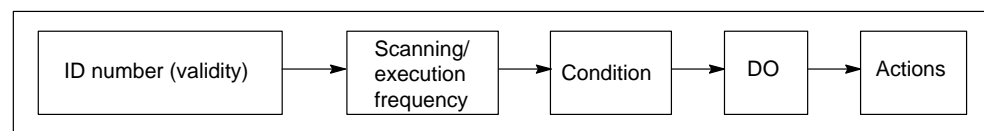


Figure 10-49 Structure of motion synchronous actions

A maximum of 320 storage elements are available. A synchronized action needs at least 5 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 to 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.

These actions are cancelled by NC Reset (user DB, "NC signals", DBX12.7) or end of program.

- **IDS = n (static synchronized action) n = 1 to 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 are cancelled by NC-RESET (user DB, "NC signals", DBX12.7).

Static synchronized actions in any operating mode are available for the FM 357-LX only in product version 2 and later.

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 10-4 and 10-5.

Note

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*$AA_IM[X]-$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 \$PC_AKTID[n] supply the status (active/not active) of a synchronized action, n is the ID No.

Actions within synchronized actions

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 synchronized 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 WHEN $A_IN[10]==FALSE DO RDISABLE  
N15 G0 X100
```

N15 is not processed as long as the condition for RDISABLE is fulfilled.

On the 0/1 edge change at digital input 10, block N15 and all subsequent blocks are enabled for processing. 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.

DELD Delete distance to go without preprocessing stop for path axes
DELD(axis) Delete distance to go without preprocessing stop for positioning axes

DELD statement does not initiate a preprocessing stop. Modal functions, e.g. continuous-path mode, are not interrupted if the trigger event does not take place.

This method of deleting the distance to go requires more time to react to a fulfilled condition.

DELD can be programmed in both non-modal and in modal or static synchronized actions.

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.

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 endlessly in the programmed direction. An end position can be preset on the fly or the axis stopped.

Value > 0: Axis motion in positive direction

Value < 0: Axis motion in negative direction

Value == 0: Stop axis motion

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.

Subroutines as actions

The function is available for the FM 357-LX only in product version 2 and later.

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, subroutine 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 9.13.1.

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
N30 ID=4 EVERY $AA_IW[X]>200 DO TRAILOF(Y,X) POS[Y]=0
N40 ID=5 EVERY $A_IN[9]==1 DO PRESETON (X1,0)

```

The X axis (conveyor belt) is traversing as an endless 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=5). 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 and the Y axis traverses back to position 0 (ID=4).

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 9.13.3) 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 9.13.3.

```

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 to 4) ; Axial measurement without
; deletion of distance to go

The function is available for the FM 357-LX only in product version 2 and later.

For details about programming and operating mode of the Measurement function, please see Sections 10.10 and 9.14.

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 job started from the NC program cannot be affected by a synchronized action.

The measurement result is stored in system variables.

\$AA_MM1 to 4[axis] ; Measured value of trigger event 1 to 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 to 20 , FW 0 to 10
N40 X40 Y40 ; LW 20 to 40 , FW 10 to 40
N50 X60 Y70 ; LW 40 to 60 , FW 40 to 70
N60 X80 Y80 ; LW 60 to 80 , FW 70 to 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 MOV[X]=1 FA[X]=10000 PRESTON(X1,-20)
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 ($AA_MEACTION[Y]==0)AND($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 signals SYNA_L1 to SYNA_L8 (user DB, "NC signals", DBX110.0 to DBX110.7) can be used by the CPU to disable synchronized actions between ID No. 1 and 8.

Arithmetic operations in synchronized actions

Complex calculations can be performed in the condition and statement components of synchronized actions (see Tables 10-4 and 10-5).

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 the same data type may be programmed in an expression.

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.

The following operators may be used in synchronized actions.

Table 10-4 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
>	Greater than
<	Less than
> =	Greater or equal
< =	Less than or equal
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 10-5 System variables

System variable	Significance	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 to 7	Marker variable, counter	r / w	r / w	INT
\$AC_PARAM[n] n = 0 to 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
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
Measurement				
\$AA_MEA[act][axis]	Status of axial measurement 0: Meas. job conditions for axis not fulfilled 1: Meas. job conditions for axis fulfilled	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
\$AA_MM[act][axis]	Measured value in MCS with MEAS	r	r	REAL
\$AA_MMi[act][axis]	Measured value in MCS with MEASA i: Trigger event 1 to 4	r	r	REAL
\$VA_IM[act][axis]	Measured encoder actual value in MCS	r	r	REAL
\$AA_ENC_ACTIVE[act][axis]	Validity of encoder actual values	r	r	BOOL

r = read, w = write

Table 10-5 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
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 at positioned 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
\$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

r = read, w = write

Table 10-5 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
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 (* the function is available for the FM 357-LX in product version 2 and later)				
\$AC_VACTB	Path velocity in MCS*		r	REAL
\$AC_VACTW	Path velocity in WCS*		r	REAL
\$AC_VC	Additive path feed override*		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_VACTW[axis]	Axis velocity, actual value in WCS		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
Master value coupling				
\$AA_LEAD_TYP[axis]	Type of master value 1: Actual value 2: Setpoint 3: Simulated master value	r	r	INT
\$AA_LEAD_SP[axis]	Position of simulated master value	r	r	REAL
\$AA_LEAD_SV[axis]	Velocity of simulated master value	r	r	REAL
\$AA_LEAD_P[axis]	Position of real master 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

r = read, w = write

Table 10-5 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
\$AA_COUP_ACT[axis]	Type of axis coupling of slave axis 0: Not coupled 3: res. 4: res. 8: Slave axis 16: Master axis	r	r	INT
\$SA_LEAD_OFF-SET_IN_POS[FA]	Slave axis position offset	r / w		REAL
\$SA_LEAD_SCALE_IN_POS [FA]	Scaling for slave axis position	r / w		REAL
\$SA_LEAD_OFF-SET_OUT_POS[FA]	Master axis position offset	r / w		REAL
\$SA_LEAD_SCALE_OUT_POS[FA]	Scaling for master 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_LIMIT[axis]	Limit for overlaid movement 0: Not reached 1: Reached in positive direction 2: Reached in negative direction		r	
CPU variables				
\$A_DBW[0] \$A_DBW[1]	Data word from CPU, FM can read Data word to CPU, FM can write (freely programmable by user)	r w	r w	INT
Motion overlay				
\$AA_OFF[axis]	Overlaid motion		r / w	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
Trace				
\$AA_SCTRACE[axis]	Generation of an IPO event (trigger event)			BOOL

r = read, w = write

Table 10-5 System variables, continued

System variable	Significance	Access to NC programs	Access to synchronized actions	Type
States				
\$AC_STAT	Current FM status 1: Aborted 2: Active 4: Interrupted 8: Reset		r	INT
\$AC_PROG	Current status of NC program 1: Program running 2: Program waiting 4: Program stopped 8: Program interrupted		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: Neutral axis 1: Path axis 2: Positioning axis from NC program 3: Positioning axis from synchr. act. 4: Positioning axis from CPU	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
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

r = read, w = write

Execution of a synchronized actions

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.

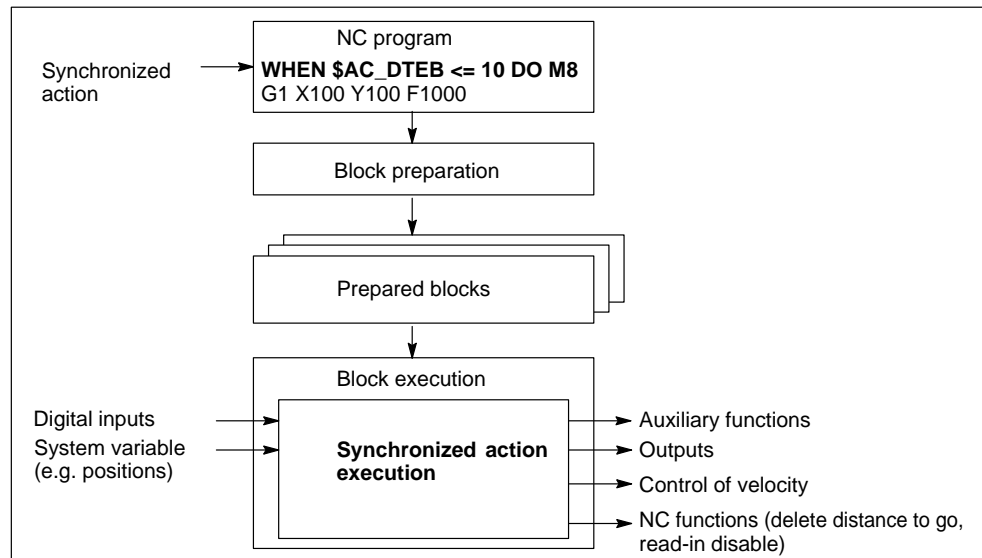


Figure 10-50 Execution of a synchronized action

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.)

10.23 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-LX in product version 2 and later.

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 to 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 to 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 reset.

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 OSCCTRL[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 OSCCTRL[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 10.22.

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[axis]
      DO POS[X]=IC(10) FA[X]=200
N100 ID=2 WHENEVER $AA_STAT[X]==1 DO $AA_OVR[Z]=0
N110 OS[Z]=0 ; Oscillation Off
N120 WAITP(Z) ; Enable Z for another motion
N120 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)

10.24 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 9.13.3.

The master value coupling function is available in product version 2 and later.

Programming

CTABDEF(FA, LA, CTAB No, TYP)	; Begin curve table definition
CTABEND	; End curve table definition
CTABDEL(CTAB No)	; Delete a curve table
CTAB(LW, CTAB-No)	; Read out slave value for a master
CTABINV(FW, LAB, CTAB No, GRAD)	; Read out master value for a slave
LEADON(FA, LE, CTAB No)	; Activate coupling
LEADOF(FA, LE)	; 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
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.

Period curve table:

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.

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.

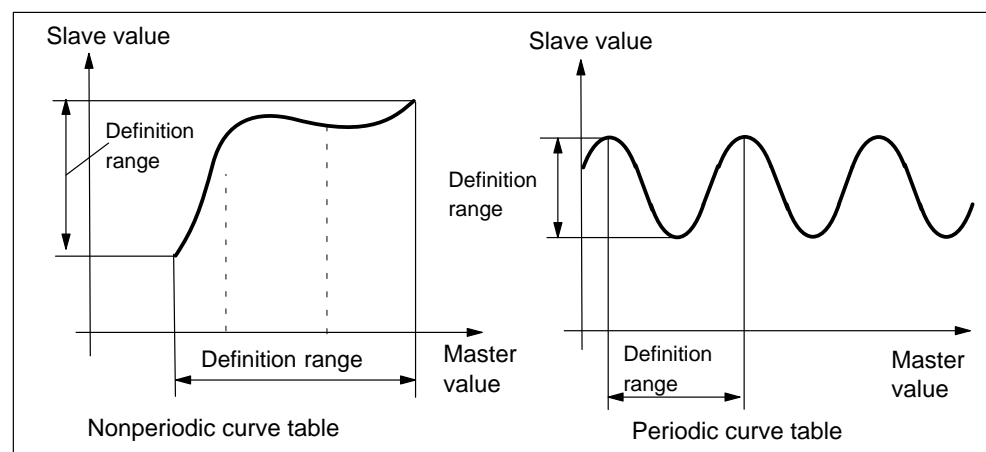


Figure 10-51 Example of periodic and nonperiodic curve tables

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:

```
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
```

Deletion of a curve table CTABDEL(CTAB No.)

This statement deletes the curve table stored under CTAB No.

A curve table can be overwritten by the CTABDEF statement, no warning is output during the overwrite operation.

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 to 20;
                       ; Slave value: Start value...0
N130 X100 Y6          ; 3rd interpolation point: Master value: 20 to 100;
                       ; Slave value: 0 to 6
N140 X150 Y6          ; 4th interpolation point: Master value: 100 to 150;
                       ; Slave value: 6 to 6
N150 X180 Y0          ; 5th interpolation point: Master value: 150 to 180;
                       ; Slave value: 6 to 0
N200 CTABEND          ; End of definition; the curve table is generated in its
                       ; internal representation as a max. 3rd degree
                       ; polynomial; the method by which the curve with the
                       ; specified interpolation points is calculated will depend
                       ; on the modally selected interpolation method (in this
                       ; example: Spline interpolation); the NC program status
                       ; before the definition was started is restored.
```

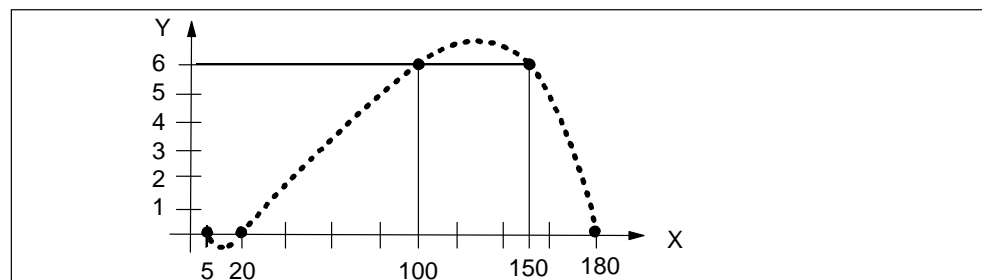


Figure 10-52 Example of curve table definition

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 10.22).

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
$AA_LEAD_TYP[LA]       ; Type of master value coupling
                        ; 0: Actual value
                        ; 1: Setpoint
                        ; 2: Simulated master value

```

10.25 Speed feedforward control (FFWON, FFEOF)

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
FFEOF ; 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.

10.26 Overview of statements

Table 10-6 Overview of statements

Statement	Significance	Information/value range	Section
ABS()	Absolute value	Parameter calculation	10.17
AC	Absolute dimension, axis-specific	Non-modal	10.2.3
ACC	Programmable acceleration	0 to 200 %	10.7.4
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		10.7.2
ADISPOS	Rounding clearance for rapid traverse		10.7.2
AMIRROR	Programmable mirror additive	Group 3, non-modal	10.3.2
AP	Polar angle	± 0.00001 to 360°	10.2.5
AROT	Programmable rotation additive	Group 3, non-modal	10.3.2
ASPLINE	Akima spline	Group 1, modal	10.6
ATRANS	Programmable zero offset additive	Group 3, non-modal	10.3.2
BAUTO	Start of spline curve, no command	Group 19, modal	10.6
BNAT	Start of spline curve, zero curvature	Group 19, modal	10.6
BRISKA()	Brisk acceleration for positioning axes		10.7.3
BRISK	Brisk acceleration for path axes	Group 21, modal	10.7.3
BSPLINE	B spline	Group 1, modal	10.6
BTAN	Start of spline curve, tangential transition	Group 19, modal	10.6
CANCEL()	Deletion of modal or static synchronized actions	Synchronized action	10.22
CLRINT()	Clear assignment between digital input and NC program		10.21
COS()	Cosine	Degrees	10.17
CR	Circle radius		10.5.6
CSPLINE	Cubic spline	Group 1, modal	10.6
CTABDEF()	Beginning of curve table definition	Synchronized action	10.24 10.22
CTABEND()	End of curve table definition	Synchronized action	10.24 10.22
CTAB()	Read out slave value for a master value	Synchronized action	10.24 10.22
CTABINV()	Read out master value for a slave value	Synchronized action	10.24 10.22
DC	Absolute dimensions for rotary axes, shortest path	Non-modal	10.2.4

Table 10-6 Overview of statements, continued

Statement	Significance	Information/value range	Section
DELD	Delete distance to go without preprocessing stop for path axes	Synchronized action	10.22
DELD()	Delete distance to go without preprocessing stop for positioning axes	Synchronized action	10.22
DELDTG	Delete distance to go with preprocessing stop for path axes	Synchronized action	10.22
DELDTG()	Delete distance to go with preprocessing stop for positioning axes	Synchronized action	10.22
DISABLE()	Deactivate ASUB		10.21
DO	Action component	Synchronized action	10.22
DRIVEA()	Drive acceleration for positioning axes		10.7.3
DRIVE	Drive acceleration for path axes	Group 21, modal	10.7.3
EAUTO	End of spline curve, no command	Group 20, modal	10.6
ENABLE()	Activate ASUB		10.21
ENAT	End of spline curve, zero curvature	Group 20, modal	10.6
ETAN	End of spline curve, tangential transition	Group 20, modal	10.6
EVERY	Scanning/execution frequency	Synchronized action	10.22
F	Path feed	Path velocity in mm/min, inch/min, degrees/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		10.23
FCUB	Cubic feedrate (spline)		10.5.2
FFWON	Activate feed control		10.25
FFWOF	Deactivate feed control		10.25
FL	Limit feed for synchronized axes	Modal	10.5.1
FNORM	Constant feedrate		10.5.2
FLIN	Linear feedrate		10.5.2
FROM	Scanning/execution frequency	Synchronized action	10.22
FXS[]	Select/deselect travel to fixed stop	Modal	10.11
FXST[]	Clamping torque	Modal	10.11
FXSW[]	Monitoring window	Modal	10.11
GOTOB	Jump backwards		10.19
GOTOF	Jump forwards		10.19
G0	Linear interpolation with rapid traverse	Group 1, modal	10.5.3
G1	Linear interpolation with feed	Group 1, modal	10.5.4

Table 10-6 Overview of statements, continued

Statement	Significance	Information/value range	Section
G2	Circular interpolation clockwise	Group 1, modal	10.5.6
G3	Circular interpolation counterclockwise	Group 1, modal	10.5.6
G4	Dwell	Group 2, non-modal	10.8
G9	Exact stop	Group 11, non-modal	10.7.1
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	10.7.1
G601	Block change on target range fine	Group 12, modal	10.7.1
G602	Block change on target range coarse	Group 12, modal	10.7.1
G64	Continuous-path mode	Group 10, modal	10.7.2
G641	Continuous-path mode with programmed rounding clearance	Group 10, modal	10.7.2
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 to 99	10.15
I	Interpolation parameter	1st geometry axis	10.5.6
IC	Incremental dimension, axis-specific	Non-modal	10.2.3
ID	Number of synchronized action	Modal	10.22
IDS	Number of a static synchronized action	Modal	10.22
IF	Conditional program jumps		10.19
J	Interpolation parameter	2nd geometry axis	10.5.6
K	Interpolation parameter	3rd geometry axis	10.5.6

Table 10-6 Overview of statements, continued

Statement	Significance	Information/value range	Section
L	Subroutine name and call		10.20
LEADON()	Activation of coupling	Synchronized action	10.24 10.22
LEADOF()	Deactivation of coupling	Synchronized action	10.24 10.22
LOCK	Disable synchronized action	Synchronized action	10.22
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 to M45, M70	Disabled		10.14
M...	Unassigned M functions:	0 to 99 (except for permanent and disabled)	10.14
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 10.22
MIRROR	Programmable mirror absolute	Group 3, non-modal	10.3.2
MOV[]	Positioning motion without end position	Synchronized action	10.22
MSG	Output messages		10.1.3
N	Block number, subblock		10.1.3
OS[]=n	Oscillation On/Off	n = 1, Oscillation On n = 0, Oscillation Off	10.23
OSCTRL[]	Control statement		10.23
OSE[]	End position		10.23
OSNSC[]	Number of residual strokes		10.23
OSP1[] OSP2[]	Position of reversal point 1 / 2		10.23
OST1[] OST2[]	Stop time at reversal point 1 / 2		10.23
P	Subroutine passes	1 to 9999	10.20
PRIO	Define priority	1 to 128	10.21
PROC	Define an ASUB		10.21
PL	Node distance for spline		10.6

Table 10-6 Overview of statements, continued

Statement	Significance	Information/value range	Section
POS[]	Positioning movement with impact on block changeover		10.5.5
	Positioning movement to end position	Synchronized action	10.22
POSA[]	Positioning movement without impact on block changeover		10.5.5
POT	Square		10.17
PRESETON	Set actual value	Synchronized action	10.4 10.22
PW	Point weight for spline		10.6
R	Arithmetic parameter	R0 to R99	10.17
RDISABLE	Programmed read-in disable	Synchronized action	10.22
RET	End of subroutine	No output to user program	10.20
REPOS	Reposition at interruption point in main program/ subroutine		10.21
RESET()	Reset synchronized action	Synchronized action	10.22
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
SD	Degree of B spline		10.6
SETINT()	Assign a digital input	1 to 4	10.21
SAVE	Restore interruption position and current processing status		10.21
SIN()	Sinus		10.17
SOFTA()	Soft acceleration for positioning axes		10.7.3
SOFT	Soft acceleration for path axes	Group 21, modal	10.7.3
STOPRE	Block preprocessor stop		10.12
SQRT()	Square root		10.17
T	Tool number	0 to 49	10.16
TAN()	Tangent	Degrees	10.17
TRAILON	Define and activate coupled-axis grouping	Synchronized action	10.9 10.22
TRAILOF	Deactivate coupled-axis grouping	Synchronized action	10.9 10.22
TRANS	Programmable zero offset absolute	Group 3, non-modal	10.3.2
TRUNC()	Integer component (truncate)		10.17
UNLOCK()	Enable synchronized action	Synchronized action	10.22
WAITP()	Wait until position reached		10.5.5

Table 10-6 Overview of statements, continued

Statement	Significance	Information/value range	Section
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	10.22
WHENEVER	Scanning/execution frequency	Synchronized action	10.22
\$A_	Current general data	System variable	10.18 10.22
\$AA_	Current axis-specific data	System variable	10.18 10.22
\$AC_	Current general data	System variable	10.18 10.22
\$P_	Programmed data	System variable	10.18
\$Rn_		System variable	10.18 10.22
\$VA_		System variable	10.18 10.22
:	Block number, main block		10.1.3
/	Skip block		10.1.3
+	Addition	Operator	10.17
-	Subtraction	Operator	10.17
*	Multiplication	Operator	10.17
/	Division	Operator	10.17
=	Assignment	Operator	10.17
==	Equal	Comparison operator	10.17
< >	Not equal	Comparison operator	10.17
>	Greater than	Comparison operator	10.17
<	Less than	Comparison operator	10.17
> =	Greater or equal	Comparison operator	10.17
< =	Less than or equal	Comparison operator	10.17



General

The FM 357 provides a system of diagnosis for

- Faults on the module and the connected I/Os
- Faults which occur during operation of the module

Localization of faults

The following tools are available for localizing faults on the FM 357:

- Status and error display on LEDs
- Error messages sent to the CPU and to HMI units (human-machine interfaces)

Error messages

Error messages of the FM 357 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*

The FM 357 is organized into the following areas:

- Communication Module (COM) – communication with the CPU and with operator panels and programming devices
- Numerical Control Kernel (NCK) – with block preparation, traversing range, etc.

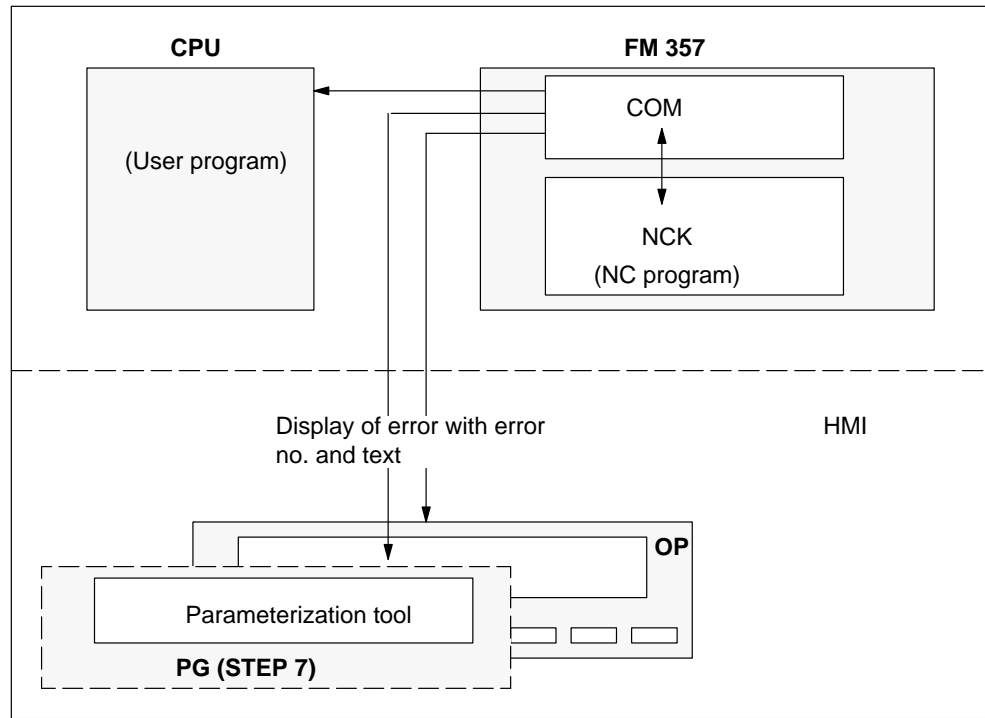


Figure 11-1 Troubleshooting

Section overview

Section	Title	Page
11.1	LED indicators	11-3
11.2	Error messages and their effect	11-7
11.3	Error list	11-9

11.1 Display by LEDs

Status and error displays

The FM 357 uses the following status and error displays:

- **SF** – Group error
- **BAF** – Battery error
- **DC5V** – Logic supply
- **DIAG** – Diagnostics

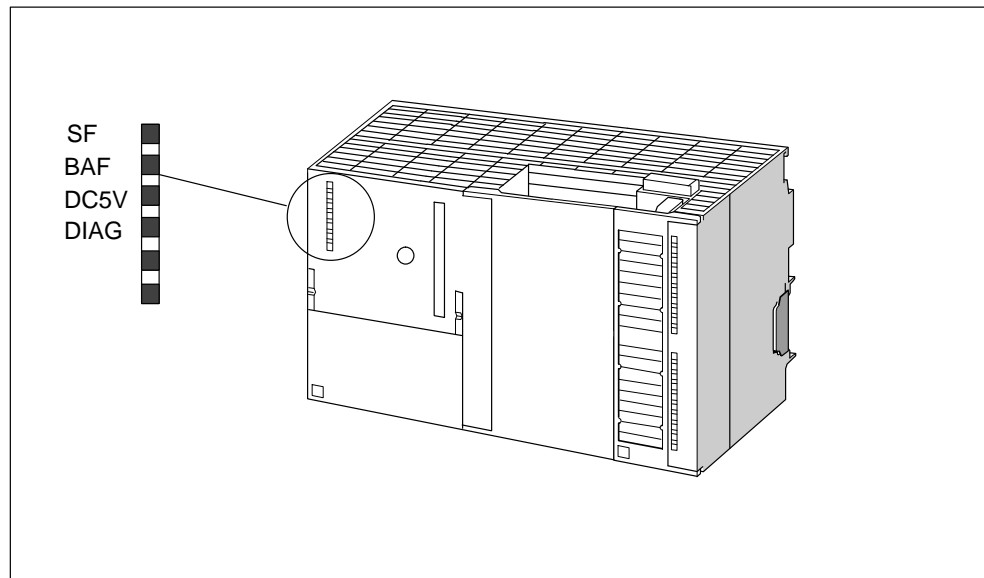


Figure 11-2 Status and error displays of the FM 357

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.

Table 11-1 Status and error displays

Display	Meaning	Explanation
SF (red) LED ON	Group error	This LED indicates an error condition in the FM 357. To eliminate the error you may need to: <ul style="list-style-type: none"> • Recommission the system • Update the firmware • Replace the FM 357
BAF (red) LED – ON LED flashing	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. If this LED flashes, you need to change the battery.
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

The following LED display indicates that the FM 357 is operating without an error:

- LED SF: OFF
- LED BAF: OFF
- LED DC5V: ON
- LED DIAG: Fast regular flashing = NC sign of life (3 Hz)

Error displays

Table 11-2 provides an overview of the LED error displays on the FM 357.

Legend:

- 1** = ON
- 0** = OFF
- n/1** = Periodic flashing n times
- x** = No meaning for the error described

Table 11-2 Summary of LED error displays

SF	BAF	DC5V	DIAG	Significance	Note
Hardware/power supply					
1	x	1	x	Hardware error ¹⁾	Replace the FM 357
1	x	0	x	5 V failure	Replace the FM 357/remove external connections
0	x	0	x	24 V failure	Check the 24 V supply/remove external connections/replace FM 357
Battery monitoring					
x	1	x	x	Battery failure	Scanning: On power-up Remedy: Replace battery
x	1/1	x	x	Battery warning, voltage below threshold	Scanning: Cyclical Remedy: Change battery as soon as possible
Power-up and software monitoring: The recommended remedy for this error group is to power up the FM 357 again with a SW update. If the error state still remains, replace the FM 357.					
1	1	1	1	RAM error in COM (power-up aborted)	
1	x	x	1/1	Error on COM power-up, parameterization by S7-300 missing.	
0	x	x	0	Power-up	Status display
0	x	x	1	CPU STOP	CPU RUN and POWER OFF/ON
0	x	x	2/1	COM watchdog	
0	x	x	3/1	NCK watchdog	
0	x	x	4/1	Internal NCK software error: Timer/memory/interface	Please contact the SIEMENS AG Hotline Tel. +49 9131 / 98 – 3836
0	x	x	5/1	Internal NCK software error: Processor errors	
0	x	x	6/1	Internal NCK software error: Plane run	
0	x	x	7/1	Internal NCK software error: Stack overflow, memory management error	
0	x	x	8/1	COM/NCK power-up error	

1) LED combination even if CPU is switched off or not switched on (no 5 V power supply on P bus).

Table 11-2 Summary of LED error displays, continued

SF	BAF	DC5V	DIAG	Significance	Note
0	x	x	9/1	COM/NCK power-up error	
1	x	x	10/1 to 17/1	Access error to local P bus	Hardware or software error on a module on the local P bus of the FM 357
1	1/1	x	1/1	RAM error	After POWER ON or RESET
Encoder monitoring					
1	x	x	2/1	Encoder supply failure	Check encoder and connection
Startup and checksum control, updates:					
1	x	x	1	No startup synchronization	Update request after 30 s
1/1	x	x	x	Update request	
1/1	x	x	2/1	COM checksum error	Perform update
1/1	x	x	3/1	NCK checksum error	Perform update
After update:					
1/1	x	x	x	Update request	
1/1	x	x	2/1	Checksum error on COM update	
1/1	x	x	3/1	Checksum error on NCK update	
1/1	x	x	4/1	Group error on NCK data unpacking	
0	x	x	1	FLASH deletion running	Status display
0	x	x	4/1	FLASH programming running	Status display
0	x	x	5/1	Update ended, o. k.	Status display, POWER OFF/ON

1) LED combination even if CPU is switched off or not switched on (no 5 V power supply on P bus).

11.2 Error messages and their effect

General

The following error messages are displayed to the user:

- NC_BEREIT (NC READY, see Section 4.7)
Ready signal NC, user DB, "NC signals", DBX25.4
- NC_FEMB
Error with machining stop, user DB, "NC signals", DBX26.7
- NC_FEOB
Error without machining stop, user DB, "NC signals", DBX26.6
- SYST_BEREIT
System ready, user DB, "NC signals", DBX7.0
- POS_FENR
Error number of positioning axis, user DB, "Axis signals", DBB33
- NC_FE
NC has error, user DB, "NC signals", DBX25.5

NC_BEREIT

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 tabulated error list 11-3, with the "Effect":
No Ready message (no NC-READY)

The "Ready" state can be restored only after elimination of the error and NC Restart (cold restart initiated manually via OP or by the "Parameterize FM 357" tool) or by switching the FM 357 power supply off and then on again.

NC_FEMB

Error message, see error list in Table 11-3, with the "Effect":

- NC Stop
- NC Start disable

The message can be acknowledged with NC Reset (RES), user DB, "NC signals", DBX12.7

NC_FEOB

Error message, see error list in Table 11-3, with the “Effect”:

Warning

The error can be acknowledged by

- CANCEL as PI service by means of FB 4
- CANCEL, error acknowledgement initiated manually via OP or “Parameterize FM 357”
- NC Reset (RES), user DB, “NC signals”, DBX12.7

SYST_BEREIT

The signal is cancelled

- in the case of faulty communication between CPU and FM 357
- when the module has not yet powered up
- when the FM 357 has a fault, diagnostic alarm

The error number is stored in GF_ERROR (basic function error) user DB, “NC signals”, DBW4 of FC 22 or FC 5. The communication ready state must be restored as described under NC_BEREIT (NC READY).

POS_FENR

Error number for axis positioning by the CPU. The error message is generated by the output parameter of FC 24.

Other error messages are implemented as output parameters of FB 2, FB 3 and FB 4.

NC_FE

An error with error number is active. The error number can be read out with FB 2 by means of variable “N_SALA_alarmNo”.

11.3 Error lists

General

This Section describes the errors of the FM 357, their cause, effect and remedy.

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
- General error: Number range 10 000 to 19 999
- Axis error: Number range 20 000 to 29 999

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 Tel +49 911 / 895 – 7000

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 11-3 below lists the following errors:

- Diagnostic errors
- General errors
- Axis errors

The error text may contain variables. These are identified by the % symbol and a number.

Example: %1 = slot number, %2 = block number, %3 = axis name

Table 11-3 Error list

Error no.	Error message, Error analysis and remedy	
Diagnostic errors		
2 000	Sign of life monitoring: CPU not active	
Cause	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. 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 also checks cyclically whether the counter state has changed.	
Effect	<ul style="list-style-type: none"> • No Ready message (no NC-READY) • NC Start disable • NC 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).	
Acknowledge-ment	NC 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 message (no NC-READY) • NC Start disable • NC Stop 	
Elimination	Determine the cause of the error on the CPU (loop or Stop in user program), and remedy.	
Acknowledge-ment	NC 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.	
Acknowledge-ment	Clear error with the CANCEL key.	
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 maintains the supply voltage for at least 30 min – the battery can be replaced within this time even then the control is switched off).	
Acknowledge-ment	Clear error with the CANCEL key.	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Diagnostic errors		
2 130	Encoder power supply (%1 V) has failed	
Cause	%1 = voltage The power supply (5 V / 24 V) of the encoders has failed.	
Effect	<ul style="list-style-type: none"> • No Ready message (no NC-READY) • NC Start disable • NC Stop • NC 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	
Acknowledgement	NC 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 message (no NC-READY) • NC Start disable • NC Stop • The NC 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. 	
Acknowledgement	Clear error with "NC Reset".	
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 	
Effect	Warning	
Elimination	After you load the default MD, you must enter/load the individual MD for the specific plant.	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Diagnostic errors		
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
	Acknowledge-ment	Clear error with the CANCEL key.
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 message (no NC-READY) • NC Start disable • NC Stop
	Elimination	Check the hardware and parameters
	Acknowledge-ment	NC 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 message (no NC-READY) • NC Start disable • NC Stop
	Elimination	Check the hardware and S7-300 configuration
	Acknowledge-ment	NC Restart
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
	Acknowledge-ment	Clear error with "NC Reset".

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
10 203	NC Start not possible with unreferenced axes	
Cause	NC 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.	
Acknowledgement	Clear error with "NC Reset".	
10 610	Axis %2 not stopped	
Cause	%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"> • NC Start disable • NC 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...	
Acknowledgement	Clear error with "NC Reset".	
10 621	Axis %2 rests on software limit switch %3	
Cause	%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	
Acknowledgement	Error remedy	
10 631	Axis %2 rests on working area limit %3	
Cause	%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	
Acknowledgement	Error remedy	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
10 650	Axis %2 Incorrect gantry machine data, error code %3	
	Cause	%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 message (no NC-READY) • NC Start disable • NC Stop
	Elimination	Correct machine data: "Leading axis", "Synchronized axis"
	Acknowledgement	Clear error with NC Restart.
10 651	Illegal gantry configuration %2	
	Cause	%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 message (no NC-READY) • NC Start disable • NC Stop
	Elimination	Correct machine data: "Leading axis", "Synchronized axis"
	Acknowledgement	Clear error with NC Restart.
10 652	Axis %2 Gantry warning threshold exceeded	
	Cause	%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 an NC Reset.
	Acknowledgement	Error remedy

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
10 653	Axis %2 Gantry error threshold exceeded	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	<ul style="list-style-type: none"> • Check axis (uneven mechanical movement?) • Parameter is set incorrectly (trip limit). An NC Restart must be executed if the parameter is changed. 	
Acknowledgement	Clear error with "NC Reset".	
10 654	Wait for synchronization start of gantry group %2	
Cause	%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	
Acknowledgement	The message disappears after the grouping has been synchronized.	
10 655	Synchronization of gantry group %2 in progress	
Cause	%2 = gantry group None	
Effect	Note	
Elimination	None	
Acknowledgement	The message disappears after the grouping has been synchronized.	
10 720	Block %3 axis %2 software limit switch %4	
Cause	%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	NC Stop	
Elimination	Correct the NC program	
Acknowledgement	Clear error with "NC Reset".	
10 730	Block %3 axis %2 working area limitation %4	
Cause	%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	NC Stop	
Elimination	<ul style="list-style-type: none"> • Correct the NC program • Change the working area limitation 	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
10 800	Block %3 axis %2 is not a geometry axis	
Cause	%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	NC Stop	
Elimination	Correct the NC program	
Acknowledge- ment	Clear error with "NC Reset".	
10 860	Block %2 Feedrate not programmed	
Cause	%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 .	
Acknowledge- ment	Clear error with "NC Reset".	
10 890	Block %2 Overflow of local block buffer when calculating splines	
Cause	%2 = block number, label The maximum permissible number of empty blocks (blocks without motion) has been exceeded.	
Effect	<ul style="list-style-type: none"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknowledge- ment	Clear error with "NC Reset".	
10 891	Block %2 Multiplicity of node is greater than its order	
Cause	%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"> • NC 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.	
Acknowledge- ment	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
10 913	Block %2 Negative feed profile is omitted	
Cause	%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	
Acknowledgement	Clear error with the CANCEL key.	
10 940	Block %2 Curve table %3: cannot be deleted/overwritten	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	All couplings which are using the curve table to be deleted must be deactivated.	
Acknowledgement	Clear error with "NC Reset".	
10 941	Block %2 Curve table %3: NC memory full	
Cause	%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"> • NC 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	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
10 942	Block %2 Curve table %3: Illegal instruction on defining	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program and restart.	
Acknow- ledgement	Clear error with "NC Reset".	
10 943	Block %2 Curve table %3: Direction change of lead value in the block not allowed	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program and restart.	
Acknow- ledgement	Clear error with "NC Reset".	
10 945	Block %2 Curve table %3: Illegal coupling of axes	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknow- ledgement	Clear error with "NC Reset".	
10 946	Block %2 Curve table %3: No contour defined	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknow- ledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
10 947	Block %2 Curve table %3: Discontinuous contour	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknow- ledgement	Clear error with "NC Reset".	
10 948	Block %2 Curve table %3: Position jump at edge of period	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknow- ledgement	Clear error with "NC Reset".	
10 949	Block %2 Curve table %3: No lead axis movement	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknow- ledgement	Clear error with "NC Reset".	
12 050	Block %2 Address %3 is not configured	
Cause	%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"> • NC Start disable • Stop NC block preparation 	
Elimination	Correct the NC program	
Acknow- ledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
12 060	Block %2 Same G group programmed repeatedly	
	Cause	%2 = block number, label 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. 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). G functions: 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
	Acknowledgement	Clear error with "NC Reset".
12 070	Block %2 Too many syntax-defining G functions	
	Cause	%2 = block number, label 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.
	Effect	NC block preparation stop
	Elimination	Correct the NC program
	Acknowledgement	Clear error with "NC Reset".
12 080	Block %2 Syntax error in text %3	
	Cause	%2 = block number, label; %3 = source text area 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. Example N10 IF GOTOF ... ; The condition for the jump is missing!
	Effect	NC block preparation stop
	Elimination	Correct the NC program
	Acknowledgement	Clear error with "NC Reset".

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
12 110	Block %2 Syntax cannot be interpreted	
Cause	%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	
Acknowledge- ment	Clear error with "NC Reset".	
12 120	Block %2 Write special G function in separate block	
Cause	%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.	
Acknowledge- ment	Clear error with "NC Reset".	
12 400	Block %2 Element of array %3 does not exist	
Cause	%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	
Acknowledge- ment	Clear error with "NC Reset".	
12 550	Block %2 Identifier %3 not defined or option does not exist	
Cause	%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)	
Acknowledge- ment	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
12 570	Block %2 Too many motion synchronous actions in %3	
Cause	%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.	
Acknowledge- ment	Clear error with "NC Reset".	
12 571	Block %2 Do not use %3 for motion synchronous actions	
Cause	%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	
Acknowledge- ment	Clear error with "NC Reset".	
12 572	Block %2 Use %3 only for motion synchronous actions	
Cause	%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	
Acknowledge- ment	Clear error with "NC Reset".	
12 580	Block %2 Invalid assignment to %3 for motion synchronous action	
Cause	%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. DO \$AA_IW[X]=10, for example, is not permitted.	
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	
Acknowledge- ment	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
12 581	Block %2 Invalid read of %3 while in motion synchronous action	
Cause	%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	
Acknowledgement	Clear error with "NC Reset".	
12 582	Block %2 Array index %3 invalid for motion synchronous action	
Cause	%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	
Acknowledgement	Clear error with "NC Reset".	
12 583	Block %2 Variable %3 no system variable	
Cause	%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.	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
12 587	Block %2 Motion synchronous action: Operation / function %3 not valid	
Cause	%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	
Acknowledgement	Clear error with "NC Reset".	
12 588	Block %2 Motion synchronous action: Address %3 illegal	
Cause	%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	
Acknowledgement	Clear error with "NC Reset".	
12 589	Block %2 Motion synchronous action: Variable %3 not allowed with modal ID	
Cause	%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.	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy								
General errors									
12 660	<p>Block %2 Motion synchronous action: Variable %3 reserved for motion synchronous actions and subroutines as actions</p> <table border="1"> <tr> <td data-bbox="331 472 475 636">Cause</td> <td data-bbox="475 472 1361 636">%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="331 636 475 680">Effect</td> <td data-bbox="475 636 1361 680">NC block preparation stop</td> </tr> <tr> <td data-bbox="331 680 475 725">Elimination</td> <td data-bbox="475 680 1361 725">Change the NC program</td> </tr> <tr> <td data-bbox="331 725 475 792">Acknowledge-ment</td> <td data-bbox="475 725 1361 792">Clear error with "NC Reset".</td> </tr> </table>	Cause	%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	Acknowledge-ment	Clear error with "NC Reset".
Cause	%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								
Acknowledge-ment	Clear error with "NC Reset".								
12 661	<p>Block %2 Subroutines as action in motion synchronous action %3: No further subroutine call possible</p> <table border="1"> <tr> <td data-bbox="331 866 475 949">Cause</td> <td data-bbox="475 866 1361 949">%2 = block number, label; %3 = name of subroutine as action It is not possible to call another subroutine as an action in a subroutine.</td> </tr> <tr> <td data-bbox="331 949 475 994">Effect</td> <td data-bbox="475 949 1361 994">NC block preparation stop</td> </tr> <tr> <td data-bbox="331 994 475 1039">Elimination</td> <td data-bbox="475 994 1361 1039">Change the NC program</td> </tr> <tr> <td data-bbox="331 1039 475 1106">Acknowledge-ment</td> <td data-bbox="475 1039 1361 1106">Clear error with "NC Reset".</td> </tr> </table>	Cause	%2 = block number, label; %3 = name of subroutine 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	Acknowledge-ment	Clear error with "NC Reset".
Cause	%2 = block number, label; %3 = name of subroutine 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								
Acknowledge-ment	Clear error with "NC Reset".								
14 000	<p>Block %2 Error at end of file</p> <table border="1"> <tr> <td data-bbox="331 1151 475 1285">Cause</td> <td data-bbox="475 1151 1361 1285">%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="331 1285 475 1368">Effect</td> <td data-bbox="475 1285 1361 1368"> <ul style="list-style-type: none"> • NC Start disable • NC block preparation stop </td> </tr> <tr> <td data-bbox="331 1368 475 1413">Elimination</td> <td data-bbox="475 1368 1361 1413">Correct the NC program</td> </tr> <tr> <td data-bbox="331 1413 475 1480">Acknowledge-ment</td> <td data-bbox="475 1413 1361 1480">Clear error with "NC Reset".</td> </tr> </table>	Cause	%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"> • NC Start disable • NC block preparation stop 	Elimination	Correct the NC program	Acknowledge-ment	Clear error with "NC Reset".
Cause	%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"> • NC Start disable • NC block preparation stop 								
Elimination	Correct the NC program								
Acknowledge-ment	Clear error with "NC Reset".								
14 011	<p>Block %2 Call program does not exist or not released for machining</p> <table border="1"> <tr> <td data-bbox="331 1525 475 1637">Cause</td> <td data-bbox="475 1525 1361 1637">%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="331 1637 475 1720">Effect</td> <td data-bbox="475 1637 1361 1720"> <ul style="list-style-type: none"> • NC Start disable • NC block preparation stop </td> </tr> <tr> <td data-bbox="331 1720 475 1803">Elimination</td> <td data-bbox="475 1720 1361 1803"> <ul style="list-style-type: none"> • Check whether NC program is stored in NC memory. • Check program enable </td> </tr> <tr> <td data-bbox="331 1803 475 1859">Acknowledge-ment</td> <td data-bbox="475 1803 1361 1859">Clear error with "NC Reset".</td> </tr> </table>	Cause	%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"> • NC 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 	Acknowledge-ment	Clear error with "NC Reset".
Cause	%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"> • NC 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 								
Acknowledge-ment	Clear error with "NC Reset".								

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
14 014	Selected program %3 or access permission not available	
Cause	%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	
Acknowledge	Clear error with the CANCEL key.	
14 040	Block %2 Error in end point of circle	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Check the circle geometry	
Acknowledge	Clear error with "NC Reset".	
14 080	Block %2 Jump destination not found	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Correct NC program (jump direction, jump destination)	
Acknowledge	Clear error with "NC Reset".	
14 092	Block %2 axis %3 has wrong axis type	
Cause	%2 = block number, label; %3 = axis name Certain keywords require a defined type of axis (see Section 10). Example: WAITP, G74	
Effect	<ul style="list-style-type: none"> • NC Start disable • NC block preparation stop 	
Elimination	Correct the NC program	
Acknowledge	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
14 095	Block %2 Circle programmed with zero radius	
	Cause	%2 = block number, label Circle radius has been programmed too small or as zero.
	Effect	<ul style="list-style-type: none"> • NC Start disable • NC block preparation stop
	Elimination	Check the circle geometry
	Acknowledge-ment	Clear error with "NC Reset".
14 750	Block %2 Too many auxiliary functions programmed	
	Cause	%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"> • NC Start disable • NC block preparation stop
	Elimination	Reduce the number of auxiliary functions to the value permitted per NC block.
	Acknowledge-ment	Clear error with "NC Reset".
14 751	Block %2 Motion synchronous actions are out of resources (code %3)	
	Cause	%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"> • NC Start disable • NC block preparation stop
	Elimination	Reduce the number of synchronous actions to the value permitted.
	Acknowledge-ment	Clear error with "NC Reset".
14 757	Block %2 Motion synchronous action uses wrong type	
	Cause	%2 = block number, label Programmed combination between action and type of motion synchronous action is illegal.
	Effect	<ul style="list-style-type: none"> • NC Start disable • NC block preparation stop
	Elimination	Correct the NC program
	Acknowledge-ment	Clear error with "NC Reset".
14 760	Block %2 Auxiliary function of a group programmed repeatedly	
	Cause	%2 = block number, label Only one auxiliary function is permitted within a group.
	Effect	<ul style="list-style-type: none"> • NC Start disable • NC block preparation stop
	Elimination	Program only one auxiliary function per auxiliary function group.
	Acknowledge-ment	Clear error with "NC Reset".

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
14 770	Block %2 Auxiliary function programmed incorrectly	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Correct the NC program	
Acknowledge- ment	Clear error with "NC Reset".	
14 790	Block %2 axis %3 currently controlled by CPU	
Cause	%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"> • NC 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). 	
Acknowledge- ment	Clear error with "NC Reset".	
15 460	Block %2 Syntax conflict with modal G function	
Cause	%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	
Acknowledge- ment	Clear error with "NC Reset".	
16 410	Block %2 axis %3 is not a geometry axis	
Cause	%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	
Acknowledge- ment	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
16 420	Block %2 axis %3 repeatedly programmed	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Delete axis addresses programmed more than once.	
Acknow- ledgement	Clear error with "NC Reset".	
16 776	Block %2 Curve table %3 does not exist for axis %4	
Cause	%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"> • NC Start disable • NC 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.	
Acknow- ledgement	Clear error with "NC Reset".	
16 777	Block %2 Lead value coupling: Lead axis %4 slave axis %3 not available	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	Release leading axis from the CPU.	
Acknow- ledgement	Clear error with "NC Reset".	
16 778	Block %2 Lead value coupling: Ring coupling for slave axis %3 and lead axis %4 not allowed	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	Correct coupling accordingly	
Acknow- ledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
16 779	Block %2 Lead value coupling: Too many couplings for axis %3, see active leading axis %4	
	Cause	%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"> • NC Start disable • NC Stop
	Elimination	Correct the NC program
	Acknowledgement	Clear error with "NC Reset".
16 794	Block %2 Coupling of axis/spindle %3 prohibits referencing	
	Cause	%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"> • NC 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.
	Acknowledgement	Clear error with "NC Reset".
16 830	Block %2 Invalid position for axis/spindle %3 programmed	
	Cause	%2 = block number, label; %3 = axis name A position outside the range 0 to 359.999 was programmed for a modulo axis.
	Effect	<ul style="list-style-type: none"> • NC Start disable • NC block preparation stop
	Elimination	Program a position within the range 0 to 359.999.
	Acknowledgement	Clear error with "NC Reset".
17 100	Block %2 Digital input no. %3 is not active	
	Cause	%2 = block number, label; %3 = no. of input An attempt was made to read a digital input of the FM 357 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"> • NC 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.
	Acknowledgement	Clear error with "NC Reset".

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
17 110	Block %2 Digital output no. %3 is not active	
Cause	%2 = block number, label; %3 = no. of output An attempt was made to read or set a digital output of the FM 357 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"> • NC 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.	
Acknowledge- ment	Clear error with "NC Reset".	
17 140	Block %2 Output no. %3 is assigned to function via machine data	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Use an unassigned output.	
Acknowledge- ment	Clear error with "NC Reset".	
17 150	Block %2 Maximum %3 FM outputs per block programmable	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Program fewer digital outputs in the block.	
Acknowledge- ment	Clear error with "NC Reset".	
17 190	Block %2 Illegal T number	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Check the tool number in the NC program.	
Acknowledge- ment	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
18 100	Block %2 Invalid argument passed to FXS[]	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Change NC program or parameter settings	
Acknowledge-ment	Clear error with "NC Reset".	
18 101	Block %2 Invalid argument passed to FXST[]	
Cause	%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"> • NC Start disable • NC block preparation stop 	
Elimination	Change NC program or parameter settings	
Acknowledge-ment	Clear error with "NC Reset".	
18 102	Block %2 Invalid argument passed to FXSW[]	
Cause	%2 = block number, label At the present time, only positive values including zero are valid.	
Effect	<ul style="list-style-type: none"> • NC Start disable • NC block preparation stop 	
Elimination	Change NC program or parameter settings	
Acknowledge-ment	Clear error with "NC Reset".	
18 200	Block %2 Curve table: Block search stop not allowed with definition CTABDEF	
Cause	%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"> • NC 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.	
Acknowledge-ment	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
General errors		
18 201	Block %2 Curve table: Table %3 does not exist	
Cause	%2 = block number, label; 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"> • NC 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.	
Acknowledgement	Clear error with "NC Reset".	
18 202	Block %2 Curve table: Instruction CTABEND illegal without CTABDEF	
Cause	%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.	
Effect	<ul style="list-style-type: none"> • NC 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.	
Acknowledgement	Clear error with "NC Reset".	
Axis errors		
20 000	Axis %2 Reference cam not reached	
Cause	%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"> • NC Start disable • NC 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. 	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Axis errors		
20 001	Axis %2 Cam signal missing	
Cause	%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"> • NC Start disable • NC 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".	
Acknow- ledgement	Clear error with "NC Reset".	
20 002	Axis %2 Zero reference mark not found	
Cause	%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"> • NC Start disable • NC 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).	
Acknow- ledgement	Clear error with "NC Reset".	
20 005	Axis %2 Reference point approach aborted	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	Check the cause of cancellation:	
Acknow- ledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Axis errors		
20 006	Axis %2 Reference point creep velocity not reached	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	Reduce the MD "reducing velocity".	
Acknowledge-ment	Clear error with "NC Reset".	
20 070	Axis 2% programmed end position is beyond software limit %3	
Cause	%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.	
Acknowledge-ment	Error remedy	
20 071	Axis %2 Programmed end position is beyond working area limit %3	
Cause	%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.	
Acknowledge-ment	Error remedy	
20 073	Axis %2 cannot be repositioned	
Cause	%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	
Acknowledge-ment	Clear error with the CANCEL key.	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Axis errors		
20 079	Axis %2 Oscillating path length %3 ≤ 0	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	Enter the correct target position (reversal point, end position)	
Acknow- ledgement	Clear error with "NC 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 message (no NC-READY) • NC Start disable • NC Stop 	
Elimination	<ul style="list-style-type: none"> • Check axis type • Does the approach block contain a programmed motion of the machine axis? 	
Acknow- ledgement	Clear error with "NC Reset".	

Table 11-3 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 message (no NC-READY) • NC Start disable • NC 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? 	
Acknowledgement	Clear error with "NC 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 message (no NC-READY) • NC Start disable • NC 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 NC Reset? • Has the CPU switched acknowledgement signals? 	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 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 message (no NC-READY) • NC Start disable • NC Stop 	
Elimination	<ul style="list-style-type: none"> • Check mechanical components, e.g. end stop broken off? • Monitoring window too small 	
Acknowledgement	Clear error with "NC 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 message (no NC-READY) • NC Start disable • NC Stop 	
Elimination	Has the CPU cancelled the acknowledgement bits even though the NC has not requested deselection?	
Acknowledgement	Clear error with "NC Reset".	
20 140	Motion synchronous action: Positioning axis cannot be traversed from synchronous action %2	
Cause	%2 = axis name An error has been detected in the positioning axis to be traversed from the synchronous action.	
Effect	<ul style="list-style-type: none"> • NC Start disable • NC Stop 	
Elimination	Change the NC program	
Acknowledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Axis errors		
20 141	Motion synchronous action: Illegal axis type	
Cause	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"> • NC Start disable • NC Stop 	
Elimination	Stop the axis first and deactivate the coupling. Then select the new state.	
Acknow- ledgement	Clear error with "NC Reset".	
20 145	Block %2 Motion synchronous action: Arithmetic error	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	Correct the relevant expression	
Acknow- ledgement	Clear error with "NC Reset".	
20 146	Block %2 Motion synchronous action: Nesting depth exceeded	
Cause	%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"> • NC Start disable • NC Stop 	
Elimination	Correct the relevant expression	
Acknow- ledgement	Clear error with "NC Reset".	
20 147	Block %2 Motion synchronous action: Command not executable	
Cause	%2 = block number A command in the synchronous action block cannot be executed, e.g. it is not possible to perform an NC Reset for the action in this block.	
Effect	<ul style="list-style-type: none"> • NC Start disable • NC Stop 	
Elimination	Change the synchronous action	
Acknow- ledgement	Clear error with "NC Reset".	

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Axis errors		
20 148	Block %2 Motion synchronous action: Internal error %3	
	Cause	%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 Tel +49 911 / 895 – 7000
	Effect	<ul style="list-style-type: none"> • NC Start disable • NC Stop
	Elimination	Change the synchronous action
	Acknow- ledgement	Clear error with “NC Reset”.
20 149	Block %2 Motion synchronous action: Illegal index	
	Cause	%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"> • NC Start disable • NC Stop
	Elimination	Use a valid index
	Acknow- ledgement	Clear error with “NC Reset”.
21 610	Axis %2 Encoder frequency limit exceeded	
	Cause	%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 message (no NC-READY) • NC Start disable • NC Stop
	Elimination	<ul style="list-style-type: none"> • Check MD “encoder frequency limit” . • Check axis velocity or encoder matching .
	Acknow- ledgement	Clear error with “NC Reset”.
21 612	Axis %2 Servo enable signal reset during traverse motion	
	Cause	%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"> • NC Stop • The NC switches to follow-up mode
	Elimination	Check the user program and/or the complete system.
	Acknow- ledgement	Clear error with “NC Start” or CANCEL key.

Table 11-3 Error list, continued

Error no.	Error message, Error analysis and remedy	
Axis errors		
21 614	Axis %2 Hardware limit switch %3 reached	
	Cause	%2 = axis name; %3 = string Axis has reached hardware limit switch.
	Effect	NC 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.
	Acknow- ledgement	Clear error with "NC Reset".
21 700	Block %3 axis %2 Touch probe already deflected, edge polarity not possible	
	Cause	%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"> • NC Start disable • NC Stop
	Elimination	<ul style="list-style-type: none"> • Check the probe • Check the start position for measurement • Check the program
	Acknow- ledgement	Clear error with "NC Reset".
21 702	Block %3 axis %2 Measurement aborted	
	Cause	%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
	Acknow- ledgement	Clear error with the CANCEL key.
21 703	Block %3 axis %2 Touch probe not deflected, edge polarity not possible	
	Cause	%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"> • NC Start disable • NC Stop
	Elimination	<ul style="list-style-type: none"> • Check the probe • Check the start position for measurement • Check the program
	Acknow- ledgement	Clear error with "NC Reset".

Table 11-3 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 message (no NC-READY) • NC Start disable • NC Stop • The NC switches to follow-up mode
	Elimination	Check the encoder
	Acknowledgement	NC 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 message (no NC-READY) • NC Start disable • NC Stop • The NC 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.
	Acknowledgement	Clear error with “NC 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 message (no NC-READY) • NC Start disable • NC Stop • The NC switches to follow-up mode
	Elimination	Check the parameters or drive
	Acknowledgement	Clear error with “NC Reset”.

Table 11-3 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 message (no NC-READY) • NC Start disable • NC Stop • The NC 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 	
Acknowledgement	Clear error with “NC 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 message (no NC-READY) • NC Start disable • NC Stop • The NC 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 	
Acknowledgement	Clear error with “NC Reset”.	

Table 11-3 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 message (no NC-READY) • NC Start disable • NC Stop • The NC 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 	
Acknowledge	Clear error with "NC 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 	
Acknowledge	Clear error with the CANCEL key.	

Table 11-3 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 message (no NC-READY) • NC Start disable • NC Stop • The NC 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 	
Acknowledge	Clear error with “NC Reset”.	
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 message (no NC-READY) • NC Start disable • NC Stop • The NC 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 	
Acknowledge	Clear error with “NC Reset”.	

Error list of all possible errors

Errors marked with an * affect functions that are not available in the FM 357.

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	System error %2 %3
001011	%3 System error %2
001012	System error %2 %3 %4
001013	System error %2
001014	System error %2
001015	Axis %2 system error %3
001016	Axis %2 system error %3
001017	Axis %2 system error %3
001018	Floating point arithmetic error in task %2 station %3 FPU state: %4
001019	Floating point arithmetic error at address %3 in task %2 FPU state: %4
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	Battery alarm
002110*	Temperature alarm
002120*	Fan alarm
002130	Encoder power supply (%1 V) has failed
002140	The current service switch position forces SRAM to be cleared at the next Power On (general reset active)
002190*	Hardware plug-in module for communication with the digitizer missing
003000	EMERGENCY STOP
003001	Internal EMERGENCY STOP
004000	Machine data %2 has gap in axis assignment
004001*	Axis %2 defined for more than one channel via machine data %3
004002	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
004010	Invalid identifier used in machine data %1 [%2]
004011	Invalid identifier used in machine data %2 [%3]
004020	Identifier %1 used several times in machine data %2
004021	Identifier %2 used several times in machine data %3
004030	Axis identifier missing in machine data %2 [%3]
004040	Axis identifier %2 not consistent with machine data %3
004050	NC code identifier %1 cannot be reconfigured to %2
004060	Standard machine data loaded
004062	Backup data loaded
004070	Normalizing machine data has been altered
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 of %4 memory.
004080*	Incorrect configuration of indexing axis in machine data %1
004100*	System cycle time corrected for digital drive

004101*	Position control cycle for digital drive reduced to %1 ms
004110	IPO cycle increased to %1 ms
004111	CPU cycle increased to %1 ms
004200	Geometry axis %2 must not be declared a rotary axis
004210*	Spindle %2 declaration as rotary axis missing
004215*	Spindle %2 declaration as modulo axis missing
004220*	Spindle %2 declared repeatedly
004225	Axis %2 declaration as rotary axis missing
004230*	Data alteration from external not possible in current channel state
004240	Runtime overflow for IPO cycle or position controller cycle IP %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 %1[%2] does not match hardware configuration
004282	Hardware of external NCK output 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
004350	Axis identifier %2 machine data %3 not consistent with machine data %4
004400	Machine data alteration will cause reorganization of buffered memory (loss of data !)
004502	Anachronism: %2(%3) -> %4
005000	Communication job not executed
006000	Memory reorganized using standard machine data
006010	Data module %2 not or not completely created, error code %3
006020	Machine data have been altered – now memory is reorganized
006030	Limit of user memory has been adapted
006401*	Tool not changed. No empty location for tool %2 Duplo no. %3 on magazine %4
006402*	Tool not changed. Magazine no. %2 not available
006403*	Tool not changed. Magazine location %2 on magazine no. %3 not available
006404*	Tool not changed. Tool %2 not available or missing
006405*	Command %2 has invalid CPU acknowledge parameter %3 – identifier %4
006406*	CPU acknowledge for command %2 is missing
006407*	Tool %2 cannot be placed in the magazine %3 in location %4. Invalid definition of magazine!
006410*	TO unit %1 tool %2 with Duplo no. %3 has reached tool monitor warning limit
006411*	Tool %2 with Duplo no. %3 has reached tool monitor warning limit
006412*	TO unit %1 tool %2 with Duplo no. %3 has reached tool monitor limit
006413*	Tool %2 with Duplo. no. %3 has reached tool monitor limit
006421*	Tool not moved. Empty location for tool %2 Duplo no. %3 on magazine %4 not available
006422*	Tool not moved. Magazine no. %2 not available
006423*	Tool not moved. Location %2 on magazine %3 not available
006424*	Tool not moved. Tool %2 not available or missing
006425*	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 assigned to spindle

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 is full
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
007100*	VDI area of %1 input and %2 output bytes defined by compile cycles exceeds %3 bytes. Maximum %3 bytes available
008000	Option "User interrupt programs" 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*	Block %2 Option "Interpolation of more than 4 axes" not set
008040	Machine data %1 reset, corresponding option is not set
008041	Axis %1: Machine data %2 deleted, corresponding option not sufficient.
008100	Block %2: Impossible due to embargo
010203	NC Start not possible with unreferenced axes
010207*	Error when selecting or deselecting the digitize function
010208	Continue program with NC Start
010209	Internal NC Stop after block search
010222*	Inter-channel communication not possible
010223	Channel %1: Command %2 already active
010225	Channel %1: Command %2 refused
010299	Function is not enabled
010600*	Block %2 Auxiliary function during thread cutting active
010601*	Block %2 Zero velocity at block end point during thread cutting
010602*	Block %2 Velocity limitation during thread cutting
010610	Axis %2 not stopped
010620	Block %3 axis %2 at software limit switch %4
010621	Axis %2 rests on software limit switch %3
010630	Block %2 axis %3 at working area limit %4
010631	Axis %2 rests at working area limit %3
010650	Axis %2 Incorrect gantry machine data, error code %3
010651	Illegal gantry configuration, error code %2
010652	Axis %2 Gantry warning threshold exceeded
010653	Axis %2 Gantry error threshold exceeded
010654	Waiting for synchronization start of gantry group %2
010655	Synchronization of gantry group %2 in progress
010700*	Block %2 NCK protected area %3 violated in automatic or MDI mode
010701*	Block %2 Channel-specific protected area %3 violated in automatic or MDI mode
010702*	NCK protected area %2 violated in manual mode
010703*	Channel-specific protection area %2 violated in manual mode
010704*	Block %2 Protected area is not guaranteed.
010706*	NCK protected area %2 reached with axis %3 in manual mode
010707*	Channel-specific protected area %2 reached with axis %3 in manual mode
010710*	Block %2 Conflict with centerless grinding
010720	Block %3 axis %2 software limit switch %4

010730	Block %3 axis %2 Working area limitation %4
010740*	Block %2 Too many empty blocks in WAB programming
010741*	Block %2 Direction reversal with WAB infeed motion
010742*	Block %2 WAB distance invalid or not programmed
010743*	Block %2 WAB programmed several times
010744*	Block %2 No valid WAB direction defined
010745*	Block %2 WAB end positioning not clear
010746*	Block %2 Block search stop for WAB
010747*	Block %2 Retraction direction defined for WAB
010750*	Block %2 Tool radius compensation activated without tool no.
010751*	Block %2 Danger of collision due to tool radius compensation
010752*	Block %2 Overflow of local block buffer with tool radius compensation
010753*	Block %2 Activate tool radius compensation in linear block only
010754*	Block %2 Deactivate tool radius compensation in linear block only
010755*	Block %2 Do not activate tool radius compensation via KONT at the current starting point
010756*	Block %2 Do not deactivate tool radius compensation via KONT at the programmed end point
010757*	Block %2 Do not change the compensation plane while tool radius compensation is active
010758*	Block %2 Curvature radius with variable compensation value too small
010759*	Block %2 Path is parallel to tool orientation
010760*	Block %2 Helical axis is not parallel to tool orientation
010761*	Block %2 Tool radius compensation for ellipse with more than one revolution not possible
010762*	Block %2 Too many empty blocks between two traversing blocks with active tool radius compensation
010763*	Block %2 Path component of the block in the compensation plane becomes zero
010764*	Block %2 Discontinuous path with active tool radius compensation
010765*	Block %2 3D tool radius compensation not possible
010766*	Illegal change of surface orientation between block %2 and block %3
010767*	Block %2 Processing with tilt angle unequal 0 not possible
010768*	Block %2 Illegal tool orientation with 3D-tool radius compensation
010769*	Block %2 Illegal surface normal vector with 3D-tool radius compensation
010770*	Block %2 Change of corner type due to change or orientation with active tool radius compensation
010771*	Block %2 Overflow of local block buffer due to orientation smoothing
010772*	Block %2 Illegal orientation change when activating or deactivating 3D face cutting
010773*	Illegal tool orientation in block %2 at inside corner with block %3
010774*	Illegal tool dimensions with face cutting in block %2
010775*	Illegal tool change with face cutting in block %2
010776*	Block %2 Axis %3 must be geo axis if cutter compensation is active
010777*	Block %2 Tool radius compensation: Too many blocks with suppression of compensation
010778*	Block %2 Preparation stop with active tool radius compensation
010800	Block %3 axis %2 it not a geometry axis
010805	Block %2 Repositioning after switch of geoaxes or transformation
010810*	Block %2 Master spindle not defined
010820	Rotary axis/spindle %2 not defined
010860	Block %2 Feedrate not programmed
010861	Block %2 Velocity of positioning axis %3 is zero
010862*	Block %2 Master spindle is axis of path
010870*	Block %2 Facing axis not defined

010880*	Block %2 Too many empty blocks between two traversing blocks when inserting chamfer or radius
010881*	Block %2 Overflow of local block buffer when inserting chamfer or radius
010882*	Block %2 Do not activate chamfer or radius without traversing
010890	Block %2 Overflow of local block buffer when calculating splines
010891	Block %2 Multiplicity of node is greater than its order
010900*	Block %2 No S value programmed for constant cutting speed
010910	Block %2 Excessive velocity of one path axis
010911*	Block %2 Transformation prohibits traversal of the pole
010912*	Block %2 Preparation and main run might not be synchronized
010913	Block %2 Negative feed profile is omitted
010914*	Movement not possible while transformation active – in channel %1 for block %2
010930*	Block %2 Interpolation type not allowed in stock removal contour
010931*	Block %2 Error in programmed stock removal contour
010932*	Block %2 Preparation of contour has been restarted
010933*	Block %2 Contour program contains too few contour blocks
010934*	Block %2 Array for contour segmentation is set too small
010940	Block %2 Curve table %3: cannot be deleted/overwritten
010941	Block %2 Curve table %3: NC memory full
010942	Block %2 Curve table %3: Illegal instruction on defining
010943	Block %2 Curve table %3: Direction change of lead value in the block not allowed
010944*	Block %2 Curve table %3: Illegal transformation
010945	Block %2 Curve table %3: Illegal coupling of axes
010946	Block %2 Curve table %3: No contour defined
010947	Block %2 Curve table %3: Discontinuous contour
010948	Block %2 Curve table %3: Position jump at edge of period
010949	Block %2 Curve table %3: No lead axis movement
012000	Block %2 Address %3 programmed repeatedly
012010	Block %2 Address %3 address type programmed too often
012020	Block %2 Combination of address modification not allowed
012030	Block %2 Invalid arguments or data types in %3
012040*	Block %2 Expression %3 is not of data type "AXIS"
012050	Block %2 DIN address %3 is not configured
012060	Block %2 Same G group programmed repeatedly
012070	Block %2 Too many syntax-defining G functions
012080	Block %2 Syntax error in text %3
012090	Block %2 Unexpected argument %3
012100	Block %2 Number of passes %3 not permissible
012110	Block %2 Syntax cannot be interpreted
012120	Block %2 Write special G function in separate block
012130*	Block %2 Tool orientation not permissible
012140	Block %2 Expression %3 not contained in this release
012150	Block %2 Operation %3 not compatible with data type
012160	Block %2 Range of values exceeded
012170	Block %2 Identifier %3 defined repeatedly
012180	Block %2 Illegal chaining of operators %3
012190*	Block %2 Variable of type ARRAY has too many dimensions
012200	Block %2 Symbol %3 cannot be created
012210*	Block %2 String %3 too long
012220*	Block %2 Binary constant %3 in string too long
012230*	Block %2 Hexadecimal constant %3 in string too long
012240*	Block %2 Tool orientation %3 defined repeatedly
012250*	Block %2 Do not nest macro %3

012260	Block %2 Too many initialization values given for %3
012261	Block %2 Initialization of %3 not allowed
012270*	Block %2 Macro identifier %3 already defined
012280*	Block %2 Maximum macro length %3 exceeded
012290	Block %2 Arithmetic variable %3 not defined
012300	Block %2 Call-by-reference argument missing on subroutine call %3
012310	Block %2 axis argument missing on procedure call %3
012320	Block %2 Argument %3 must be call-by-reference
012330	Block %2 Type of argument %3 incorrect
012340	Block %2 Number of arguments exceeded in %3
012350	Block %2 Argument %3 not accepted because AXIS argument is missing
012360	Block %2 Dimension of argument %3 incorrect
012370	Block %2 Range of values exceeded for %3
012380	Block %2 Maximum memory capacity exceeded
012390	Block %2 Type of initial value for %3 cannot be converted
012400	Block %2 Element of array %3 does not exist
012410	Block %2 Incorrect index type for %3
012420	Block %2 Identifier %3 too long
012430	Block %2 Invalid index
012440	Block %2 Maximum number of formal arguments exceeded
012450	Block %2 Label defined repeatedly
012460	Block %2 Maximum number of symbols exceeded with %3
012470	Block %2 Unknown G function %3 used
012480	Block %2 Subroutine %3 already defined
012490	Block %2 Access permission level %3 is not valid
012500	Block %2 Do not use %3 in this module
012510	Block %2 Too many machine data %3
012520	Block %2 Too many tool parameters %3
012530	Block %2 Invalid index for %3
012540	Block %2 is too long or too complex
012550	Block %2 Identifier %3 not defined or option does not exist
012560	Block %2 Programmed value %3 exceeds allowed limits
012570	Block %2 Too many motion synchronous actions in %3
012571	Block %2 Do not use %3 for motion synchronous actions
012572	Block %2 Use %3 only for motion synchronous actions
012580	Block %2 Invalid assignment to %3 for motion synchronous action
012581	Block %2 Invalid read of %3 while in motion synchronous action
012582	Block %2 Array index %3 invalid for motion synchronous action
012583	Block %2 Variable %3 no system variable
012584	Block %2 Variable %3 cannot be read synchronously with motion
012585	Block %2 Variable %3 cannot be changed synchronously with motion
012586	Block %2 Motion synchronous action: Type conflict in variable %3
012587	Block %2 Motion synchronous action: Operator / function %3 not valid
012588	Block %2 Motion synchronous action: Address %3 illegal
012589	Block %2 Motion synchronous action: Variable %3 not allowed with modal ID
012590*	Block %2 Global user data cannot be created
012600*	Block %2 Invalid checksum of line
012610	Block %2 Accessing single char with call-by-reference argument not allowed %3
012620	Block %2 Accessing this variable as single char not allowed
012630*	Block %2 Skip / label not allowed
012640*	Block %2 Invalid nesting of control structures
012641*	Block %2 Nesting level of control structures exceeds limit
012650*	Block %2 axis %3 name different in channel %4

012660	Block %2 Motion synchronous action: Variable %3 reserved for motion synchronous actions and subroutines as action
012661	Block %2 Technology cycle %3: No further subroutine call possible
014000	Block %2 Error at end of file
014001	Block %2 Error at end of block, line feed missing
014010	Block %2 Invalid default argument in subroutine call
014011	Block %2 Program %3 does not exist or not released for machining
014012	Block %2 Lowest subroutine level exceeded
014013	Block %2 Number of subroutine passes invalid
014014	Selected program %3 or access permission not available
014015	Channel %1: No access permission for file
014020	Block %2 Wrong number of arguments on function or procedure call
014021	Block %2 Wrong number of arguments on function or procedure call
014025	Block %2 Motion synchronous action: Illegal modal ID
014026*	Block %2 Motion synchronous action: Invalid polynomial in the FCTDEF command
014040	Block %2 Error in end point of circle
014045*	Block %2 Error in tangent circle programming
014048	Block %2 Incorrect number of revolutions programmed for circle
014050	Block %2 Nesting depth for arithmetic operations exceeded
014051	Block %2 Arithmetic error in part program
014060*	Block %2 Invalid skip level with differential block skip
014070	Block %2 Memory for variables not sufficient for subroutine call
014080	Block %2 Jump destination not found
014090	Block %2 Invalid D number
014091	Block %2 Invalid function, index %3
014092	Block %2 Axis %3 has wrong axis type
014093*	Block %2 Path interval zero or negative with polynomial interpolation
014094*	Block %2 Polynomial degree greater than 3 programmed for polynomial interpolation
014095	Block %2 Circle programmed with zero radius
014096	Block %2 Type conversion not possible
014097*	Block %2 String cannot be converted to AXIS type
014098	Block %2 Conversion error: Not a number
014099	Block %2 Result in string concatenation too long
014100*	Block %2 Orientation transformation not available
014101*	Block %2 Orientation transformation not active
014110*	Block %2 Do not mix use of Euler angles and orientation, vector components
014111*	Block %2 Do not mix use of Euler angles, orientation vector and transformation axes
014112*	Block %2 Programmed orientation path not possible
014113*	Block %2 Programmed lead angle too large
014114*	Block %2 Programmed tilt angle too large
014115*	Block %2 Illegal definition of part surface
014116*	Block %2 Absolute orientation programmed while ORIPATH is active
014120*	Block %2 Determination of plane not possible for programmed orientation
014130	Block %2 Too many initialization values given
014150*	Block %2 Illegal tool carrier number programmed or declared (MD)
014151*	Block %2 Illegal tool carrier rotation
014152*	Block %2 Tool carrier: Invalid orientation
014200	Block %2 Polar radius negative
014210	Block %2 Polar radius too large
014250	Block %2 Pole radius negative
014260	Block %2 Pole angle too large

014270	Block %2 Pole programmed incorrectly
014280	Block %2 Polar coordinates programmed incorrectly
014300*	Block %2 Overlaid handwheel motion activated incorrectly
014310*	Handwheel %1 configuration not correct or inactive
014400*	Block %2 Tool radius compensation active at transformation switchover
014401*	Block %2 Transformation not available
014402*	Block %2 Spline active at transformation change
014403	Block %2 Preparation might not be synchronized with main run
014404*	Block %2 Invalid argument in selection of transformation
014410	Block %2 Spline active at change of geoaxis
014411*	Block %2 Tool radius compensation active at change of geoaxis
014412*	Block %2 Transformation active at change of geoaxis
014413*	Block %2 Fine tool correction: Changeover geometry / channel axis not allowed
014414*	Block %2 Function GEOAX: Incorrect call
014420*	Block %2 Indexing axis %3 frame not allowed
014500	Block %2 Illegal DEF or PROC statement within part program
014510	Block %2 PROC statement missing on subroutine call
014520	Block %2 Illegal PROC statement in data definition section
014530	Block %2 EXTERN and PROC statement do not correspond
014600*	Block %2 Buffer %3 for sequential reload cannot be established
014601*	Block %2 Reload buffer cannot be erased
014602*	Block %2 Timeout for EXTCALL
014610*	Block %2 Compensation block not possible
014650	Block %2 SETINT uses invalid input to trigger ASUB
014660	Block %2 SETINT instruction uses invalid priority level
014700	Block %2 Timeout after command to interpreter
014701	Block %2 Number of available NC blocks reduced by %3
014710	Block %2 Error during phase %3 of INIT block generation
014720*	Block %2 Axes missing for centerless grinding
014730*	Block %2 Conflict at activation of centerless transformation
014740*	Block %2 Tool data missing for centerless grinding
014745*	Block %2 Centerless grinding not active
014750	Block %2 Too many auxiliary functions programmed
014751	Block %2 Motion synchronous actions are out of resources (code: %3)
014752	Block %2 Conflict with DELDTG and STOPREOF in synchronous action
014753	Block %2 Motion synchronous action uses illegal interpolation type
014754	Block %2 Motion synchronous action uses wrong feed type
014755	Block %2 Motion synchronous action needs traverse motion
014756	Block %2 Motion synchronous action uses wrong value
014757	Block %2 Motion synchronous action uses wrong type
014758*	Block %2 Programmed synchronous value is not available
014759	Block %2 Motion synchronous action uses wrong axis type
014760	Block %2 Auxiliary function of a group programmed repeatedly
014761*	Block %2 Motion synchronous action: DELDTG not allowed with active radius compensation
”014762	Block %2 Too many CPU variables programmed
014770	Block %2 Auxiliary function programmed incorrectly
014780	Block %2 Unreleased option used
014790	Block %2 Axis %3 currently controlled by CPU
014800	Block %2 Programmed path speed less or equal to zero
014810	Block %2 Negative axis speed for positioning axis %3
014811	Block %2 Acceleration value of axis/spindle %3 out of range
014812	Block %2 Axis %3 SOFTA not available

014820*	Block %2 Negative value for maximum spindle speed programmed with constant cutting speed
014821*	Block %2 Error in selection or disabling of GWPS
014822*	Block %2 Incorrect programming of GWPS
014823*	Block %2 Error on selection or disabling of tool monitoring
014824*	Block %2 Conflict with GWPS
014830	Block %2 Wrong feed type selected
014840*	Block %2 Value for constant cutting speed out of range
014900	Block %2 Use either center point or end point programming
014910	Block %2 Invalid angle of aperture for programmed circle
014920	Block %2 Intermediate point of circle incorrect
015000*	Block %2 Channel-sync instruction using illegal mark
015010*	Block %2 Program coordination instruction with invalid channel number
015020*	Block %2 Instruction CHANDATA cannot be executed. Channel %3 is not active
015021*	Block %2 Instruction CHANDATA uses invalid channel number
015100	Block %2 REORG abort caused by overflow of log file
015110	Block %2 REORG currently not possible
015150*	Block %2 Reload from external aborted
015160	Block %2 Wrong configuration of block buffer
015165	Block %2 Error by translating or interpreting CPU interrupt program %3
015170	Block %2 Program %3 could not be compiled
015175	Block %2 Program %3 interfaces could not be generated
015180*	Block %2 Program %3 cannot be executed as INI file
015185*	%2 Errors in INI file
015190	Block %2 No enough free memory for subroutine call
015300	Block %2 Invalid number-of-passed blocks during block search
015310*	Block %2 File requested during block search is not loaded
015320*	Block %2 Invalid block search command
015330*	Block %2 Invalid block number as target of block search
015340*	Block %2 Invalid label as target of block search
015350*	Block %2 Target of block search not found
015360*	Invalid target of block search (syntax error)
015370*	Target of block search not found
015400*	Block %2 Selected initial ini file does not exist
015410*	Block %2 Initialization file contains invalid M function
015420	Block %2 Instruction not accepted in current mode
015450	Block %2 Compiled program cannot be stored
015460	Block %2 Syntax conflict with modal G function
015500*	Block %2 Illegal angle of shear
015700*	Block %2 Illegal cycles alarm number
015800*	Block %2 Wrong starting condition for CONTPRON
015810*	Block %2 Wrong array dimension for CONTPRON
015900	Block %2 Touch probe not available
015910	Block %2 Touch probe not available
015950	Block %2 No traverse motion programmed
015960	Block %2 No traverse motion programmed
016000*	Block %2 Invalid value for lifting direction
016005*	Block %2 Invalid value for lifting distance
016010*	Block %2 Processing stop after rapid lift from contour
016020	Repositioning in block %2 is not possible
016100*	Block %2 spindle %3 not available in channel
016105*	Block %2 spindle %3 cannot be assigned
016110*	Block %2 spindle %3 for dwell time not in speed control mode
016120*	Block %2 Invalid index for online tool compensation

016130*	Block %2 Instruction not allowed with active FTOCON
016140*	Block %2 FTOCON not allowed
016150*	Block %2 Invalid spindle no. with PUTFTOCF
016200	Block %2 Spline and polynomial interpolation not available
016300*	Block %2 Invalid zero crossing of denominator polynomial within parameter range
016400	Block %2 Positioning axis %3 cannot participate in spline interpolation
016410	Block %2 Axis %3 is not a geometry axis
016420	Block %2 Axis %3 repeatedly programmed
016430	Block %2 Geometry axis %3 cannot traverse as positioning axis in rotated coordinate system
016500*	Block %2 Chamfer or radius negative
016510*	Block %2 Facing axis is not defined
016700	Block %2 axis %3 Invalid feed type
016710*	Block %2 axis %3 Master spindle not programmed
016715*	Block %2 axis %3 Master spindle not at standstill
016720*	Block %2 axis %3 Thread lead is zero
016730	Block %2 axis %3 Wrong parameter for thread cutting
016740	Block %2 Geometry axis must be programmed
016750*	Block %2 axis %3 SPCON not programmed
016751*	Block %2 spindle/axis %3 SPCOF
016755	Block %2 No wait needed
016760*	Block %2 axis %3 S value missing
016761	Block %2 axis/spindle %3 not programmable in channel
016762*	Block %2 spindle %3 Function of thread or drill is active
016763*	Block %2 axis %3 Programmed speed is illegal (zero or negative)
016770	Block %2 axis %3 Encoder missing
016776	Block %2 Curve table %3 does not exist for axis %4
016777	Block %2 Lead value coupling: Lead axis %4 slave axis %3 not available
016778	Block %2 Lead value coupling: Ring coupling for slave axis %3 and lead axis %4 not allowed
016779	Block %2 Lead value coupling: Too many couplings for axis %3, see active leading axis %4
016780	Block %2 Slave axis/spindle missing
016781	Block %2 Master axis/spindle missing
016782	Block %2 Slave axis/spindle %3 currently not available
016783	Block %2 Master axis/spindle %3 currently not available
016785	Block %2 Master and slave axis/spindle %3 are identical
016787	Block %2 Coupling parameter is not to be changed
016788	Block %2 Resulting coupling definition is cyclic
016789	Block %2 Axis/spindle used in another coupling definition
016790	Block %2 Coupling parameter is zero or missing
016791	Block %2 Coupling parameter neglected
016792	Block %2 Too many couplings for axis/spindle %3
016793*	Block %2 Coupling of axis %3 prohibits switchover of transformation
016794	Block %2 Coupling of axis/spindle %3 prohibits referencing
016795	Block %2 String cannot be interpreted
016796	Block %2 Coupling not defined
016797	Block %2 Coupling is active
016800	Block %2 Traverse instruction DC/CDC for axis %3 not allowed
016810	Block %2 Traverse instruction ACP for axis %3 not allowed
016820	Block %2 Traverse instruction ACN for axis %3 not allowed
016830	Block %2 Invalid position for axis/spindle %3 programmed
016903	Action %2 not allowed in current state

016904	Action %2 not allowed in current state
016905	Action %2 not allowed
016906	Action %2 is aborted because of an active alarm
016907	Action %2 only possible in Stop
016908	Action %2 only possible in Reset or at the block end
016909	Action %2 is not allowed in current mode
016911	Mode change is not allowed
016912	Action %2 only possible in Reset
016913	Mode change: Action %3 not allowed
016914	Mode change: Action %3 not allowed
016915	Action %2 in the current block not allowed
016916	Reposition: Action %2 not allowed in the current state
016918*	Action %2 needs reset in all channels
016919	Action %2 is not allowed because of an alarm
016920	Action %2 is already enabled
016921*	Machine data: Channel/mode group assignment not allowed or double
016922	Subprograms: Action %2 maximum stack level exceeded
016923	Action %2 not allowed in the current state
016924*	Caution: Program test will change the tool data. Make a tool/magazine data backup!
016925	Action %2 not allowed in the current state, action %3 active
016926*	Channel coordination: Action %2 not allowed in block %3, marker %4 already set
016927	Action %2 at active interrupt treatment not allowed
016928	Interrupt treatment: Action %2 not allowed in current state
016930	Predecessor and current block %2 must be separated by an executable block
016931	Subprograms: Action %2 maximum stack level exceeded
016932	Conflict on activation of user data type %2
017640*	Block %2 Spindle cannot be used as transformed axis %3
017000	Block %2 Maximum number of symbols exceeded
017001*	Block %2 No memory left for tool or magazine data
017010	Block %2 No memory left for symbol
017020	Block %2 1st array index out of range
017030	Block %2 2nd array index out of range
017040	Block %2 Illegal axis index
017050	Block %2 Illegal value
017060	Block %2 Requested data area too large
017070	Block %2 Data is write-protected
017080	Block %2 Value violates lower limit
017090	Block %2 Value violates upper limit
017100	Block %2 Digital input no. %3 is not active
017110	Block %2 Digital output no. %3 is not active
017120	Block %2 Analog input no. %3 is not active
017130	Block %2 Analog output no. %3 is not active
017140	Block %2 Output no. %3 is assigned to function via machine data
017150	Block %2 Maximum %3 FM outputs per block programmable
017160	Block %2 Tool is not selected
017170	Block %2 Too many symbols defined
017180	Block %2 Illegal D numbers
017190	Block %2 Illegal T number
017200	Block %2 Cannot delete an active tool
017210	Block %2 Access to variable not possible
017220	Block %2 Tool not available
017230*	Block %2 Duplo number already disposed

017240*	Block %2 Invalid definition of tool
017250*	Block %2 Illegal definition of magazine
017260*	Block %2 Illegal definition of magazine location
017270	Block %2 call-by-reference: Illegal variable
017500*	Block %2 Axis %3 is not an indexing axis
017501*	Block %2 Indexing axis %3 with Hirth tooth system is active
017502*	Block %2 Indexing axis %3 with Hirth tooth system Stop delayed
017503*	Block %2 Indexing axis %3 with Hirth tooth system and axis not referenced
017510*	Block %2 Invalid index for indexing axis %3
017600*	Block %2 Preset on transformed axis %3 not possible
017610*	Block %2 Positioning axis %3 cannot participate in transformation
017620*	Block %2 Fixpoint cannot be approached for transformed axis %3
017630*	Block %2 Referencing not possible for transformed axis %3
017800*	Block %2 Illegal fixed-stop end point programmed
017900	Block %2 axis %3 Use machine axis identifier
018000*	Block %2 Wrong definition of NCK-specific protection area %3, error code %4
018001*	Block %2 Wrong definition of channel-specific protection area %3, error code %4
018002*	Block %2 NCK protection area %3 cannot be activated, error code %4
018003*	Block %2 Channel-specific protection area %3 cannot be activated, error code %4
018004*	Block %2 Orientation of workpiece-related protection area %3 does not correspond to the orientation of tool-related protection area %4
018005*	Block %2 Serious error in definition of NCK-specific protection area %3
018006*	Block %2 Serious error in definition of channel-specific protection area %3
018100	Block %2 Invalid argument passed to FXS[]
018101	Block %2 Invalid argument passed to FXST[]
018102	Block %2 Invalid argument passed to FXSW[]
018200	Block %2 Curve table: Block search stop not allowed with definition CTABDEF
018201	Block %2 Curve table: Table %3 does not exist
018202	Block %2 Curve table: Instruction CTABEND illegal without CTABDEF
018300	Block %2 Frame: Fine shift not possible
020000	Axis %2 Reference cam not reached
020001	Axis %2 cam signal missing
020002	Axis %2 zero reference mark not found
020003	Axis %2 Encoder error
020004	Axis %2 Reference mark missing
020005	Axis %2 Reference point approach aborted
020006	Axis %2 Reference point creep velocity not reached
020007*	Axis %2 Reference point approach needs 2 encoders
020008*	Axis %2 Reference point approach needs second referenced encoder
020050*	Axis %2 Handwheel mode active
020051*	Axis %2 Handwheel mode not possible
020052	Axis %2 already active
020053*	Axis %2 DRF, FTOCON, external setting of offset not possible
020054*	Axis %2 Wrong index for indexing axis in JOG mode
020055*	Master spindle not available in JOG mode
020056*	Axis %2 No revolution feedrate possible. Axis/spindle %3 stationary
020057*	Block %2 Revolution velocity of axis/spindle %3 is less or equal zero.
020060	Axis %2 cannot move as geometry axis
020062	Axis %2 already active
020065*	Master spindle not defined for geometry axes in JOG mode
020070	Axis %2 Programmed end position is beyond software limit %3

020071	Axis %2 programmed end position is beyond working area limit %3
020072*	Axis %2 is not an indexing axis
020073	Axis %2 cannot be repositioned
020074*	Axis %2 Wrong index position
020075	Axis %2 Oscillation currently not possible
020076	Axis %2 Change of operation mode not possible during oscillation
020077	Axis %2 Programmed position is beyond software limit %3
020078	Axis %2 Programmed position is beyond working area limit %3
020079	Axis %2 Oscillating path length %3 ≤ 0
020080*	Axis %2 Handwheel not assigned for overlaid handwheel motion
020085*	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
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*	Contour tunnel monitoring
020140	Motion synchronous action: Traversing of command axis %2 not possible
020141	Motion synchronous action: Illegal axis type
020145	Block %2 Motion synchronous action: Arithmetic error
020146	Block %2 Motion synchronous action: Nesting depth exceeded
020147	Block %2 Motion synchronous action: Command not executable
020148	Block %2 Motion synchronous action: Internal error %3
020149	Block %2 Motion synchronous action: Illegal index
020150*	Tool management: CPU terminates the interrupted command
020160*	Tool management: CPU can terminate only incorrectly aborted commands
020170*	Illegal configuration \$AC_FIFO
020200*	Invalid spindle no. %2 with fine compensation of tool geometry
020201*	Spindle %2 No tool assigned
020203*	No tool selected
020204*	Instruction PUTFTOC not allowed during FTOCOF
020210*	Block %3 spindle %2 Wrong values for centerless grinding
020211*	Block %3 spindle %2 Support point beyond limits
021610	Axis %2 Encoder frequency limit exceeded
021612	Axis %2 Servo enable reset during traverse motion
021613*	Axis %1 switches active encoder
021614	Axis %2 Hardware limit switch %3 reached
021615	Axis %2 taken from traverse mode to follow-up mode

021616*	Block %2 Overlaid motion active at transformation switchover
021617*	Block %2 Transformation does not allow traversal of the pole
021618*	As from block %2 Transformation active: Overlaid motion too great
021619*	Channel %1, Block %2 Transformation active: Motion not possible
021650	Axis %2 Overlaid motion not allowed
021700	Block %3 axis %2 Touch probe already deflected, edge polarity not possible
021701	Block %3 axis %2 Measurement not possible
021702	Block %3 axis %2 Measurement aborted
021703	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 output of cam signals via timer
021760	Block %2 Too many auxiliary functions programmed
022000*	Block %3 spindle %2 Change of gear stage not possible
022010*	Block %3 spindle %2 Actual gear stage differs from requested gear stage
022040*	Block %3 spindle %2 is not referenced with zero marker
022045*	Block %2 spindle/axis %3 not available because it is active in channel %4
022050*	Block %3 spindle %2 Transition from speed control mode to position control mode not possible
022051*	Block %3 spindle %2 Reference mark not found
022052*	Block %3 spindle %2 Zero speed on block change not reached
022053*	Block %3 spindle %2 Reference mode not supported
022054*	Block %3 spindle %2 Improper punch signal
022055*	Block %3 spindle %2 Configured positioning speed is too high
022062	Axis %2 Reference point approach: Search speed for zero mark (MD) is not reached
022064	Axis %2 Reference point approach: Search speed for zero mark (MD) is too high
022065*	Tool management: Tool move not possible since there is no tool %2 with Duplo no. %3 in magazine %4
022066*	Tool management: Tool change not possible since tool %2 with Duplo no. %3 not in magazine %4
022067*	Tool management: Tool not changed because no tool available in tool group %2
022068*	Block %2 Tool management: No tool available in tool group %3
022100*	Block %3 spindle %2 Chuck speed exceeded
022101*	Block %3 spindle %2 Maximum speed for encoder resynchronization exceeded
022150*	Block %3 spindle %2 Maximum speed for position control exceeded
022200*	Spindle %2 Axis stopped during tapping
022250*	Spindle %2 Axis stopped during thread cutting
022260*	Spindle %2 Thread might be damaged
022270*	Block %2 spindle %3 Spindle speed too high for thread cutting
022320*	Block %2 PUTFTOCF data block could not be transferred
022321*	Axis %2 PRESET not allowed during traverse motion
022322*	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
025030	Axis %1 Actual velocity alarm
025031	Axis %1 Actual velocity warning
025040	Axis %1 Standstill monitoring
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
025200	Axis %1 Requested set of parameters invalid
025201	Axis %1 Drive fault
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
026050	Axis %1 Parameter block change from %2 to %3 not possible
026100	Axis %1 Drive %2 Sign of life of drive %2 missing
060000*	Block %2
061000*	Block %2
062000*	Block %2
063000*	Block %2
065000*	Block %2
066000*	Block %2
067000*	Block %2
068000*	Block %2
070000*	Compile cycle alarm
075000*	OEM alarm
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 found on drive bus has not been configured
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
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)



Technical Specifications

A

General

This section describes the technical data of the FM 357 multi-axis module.

- General technical data
- Dimensions and weight
- Load memory
- Encoder inputs
- Drive port
- Digital inputs

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 Programmable Controller; Hardware and Installation*.

UL/CSA Approvals

The following approvals have been granted for the S7-300:

UL Recognition Mark
Underwriters Laboratories (UL) in compliance with
UL Standard 508, File E 116536

CSA Certification Mark
Canadian Standard Association (CSA) in compliance with
Standard C 22.2 No. 142, File LR 48323

FM Approval

The following FM approval has been granted for the S7-300:
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 while the S7-300 is in operation.

In areas where there is a risk of explosion, always cut off power to the S7-300 before disconnecting plugs.



Warning

WARNING - NEVER DISCONNECT WHILE CIRCUIT IS LIVE
UNLESS LOCATION IS KNOWN TO BE NONHAZARDOUS

CE Mark

Our products comply with the requirements of EU Directive 89/336/EEC "Electromagnetic Compatibility" and the relevant harmonized European standards (EN).



The Declaration of Conformity in accordance with Article 10 of the EU Directive referenced above is contained in this manual (see Chapter B).

Field of application

SIMATIC products are designed for industrial applications.

Application	Requirement concerning	
	Noise emission	Noise immunity
Industry	EN 50081-2 : 1993	EN 50082-2 : 1995
Residential	Individual license	EN 50082-1 : 1992

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. 1150

User data memory

Non-volatile RAM, 512 KB

System cycles

Position control cycle: 6 ms; Interpolations: 18 ms

Encoder inputs

Position detection	<ul style="list-style-type: none"> • Incremental • Absolute (SSI)
Signal voltages	Inputs: 5 V, RS422-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. 1.25 Mbit/s with 10 m conductor length shielded • max. 156 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 to 5.25 V) • max. 35 m at max. 210 mA (tolerance 4.75 to 5.25 V) • max. 100 m at max. 300 mA (tolerance 20.4 to 28.8 V) • max. 300 m at max. 300 mA (tolerance 11 to 30 V)
Cable length with absolute encoder (SSI)	See data transfer rate

Drive port

Analog drive

Setpoint signal	
Rated voltage range	-10.5 to 10.5 V
Output current	-3 to 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

Output signals 5 V in conformity with RS 422 standard		
Differential output voltage	V_{OD}	min. 2 V ($R_L = 100 \Omega$)
Output voltage "1"	V_{OH}	3.7 V ($I_O = -20$ mA) 4.5 V ($I_O = -100$ μ A)
Output voltage "0"	V_{OL}	max. 1 V ($I_O = 20$ mA)
Load resistance	R_L	55 Ω
Output current	I_O	max. ± 60 mA
Pulse frequency	f_P	max. 625 kHz
Cable length		max. 50 m in hybrid configurations with analog axes 35 m with asymmetrical transfer 10 m

Digital inputs

Number of inputs	6
Supply voltage	DC 24 V (permissible range: 20.4 to 28.8 V)
Electrical isolation	
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 15 mA
Input delay (I0 to I5)	<ul style="list-style-type: none"> • 0 \rightarrow 1 signal: type 15 μs • 1 \rightarrow 0 signal: type 150 μs
Connecting a 2-conductor sensor	Possible



EC Declaration of Conformity

B

SIEMENS

EG-Konformitätserklärung

Nr. E002 V 21/03/97

Hersteller: SIEMENS AG

Anschrift: SIEMENS AG AUT 2
Frauenauracherstraße 80
91056 Erlangen

Produktbezeichnung: SINUMERIK 805, 805SM-P, 805SM-TW, 810, 810D,
820, 840C, 840CE, 840D, 840DE, FM NC
SIMATIC FM 353, FM 354, FM 357
SIROTEC RCM1D, RCM1P
SIMODRIVE 610, 611A, 611D, MCU, FM STEPDRIVE

Die bezeichneten Produkte stimmen mit den Vorschriften folgender Europäischer Richtlinie überein:

89/336/EWG-Richtlinie des Rates zur Angleichung der Rechtsvorschriften der Mitgliedsstaaten über die elektromagnetische Verträglichkeit (geändert durch 91/263/EWG, 92/31/EWG und 93/68/EWG)

Die Einhaltung dieser Richtlinie setzt einen EMV-gerechten Einbau der Produkte in die Gesamtanlage voraus.
Anlagenkonfigurationen, bei der die Einhaltung dieser Richtlinie nachgewiesen wurde, sowie angewandte Normen, siehe:

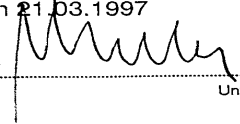
- Anhang A1 - A14 (Anlagenkonfigurationen)
- Anhang B1 - B7 (Komponenten)
- Anhang C (Normen)

SIEMENS

Erlangen, den 21.03.1997

R. Müller
Entwicklungsleitung

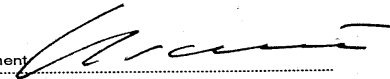
Name, Funktion



Unterschrift

K. Krause
Qualitätsmanagement

Name, Funktion



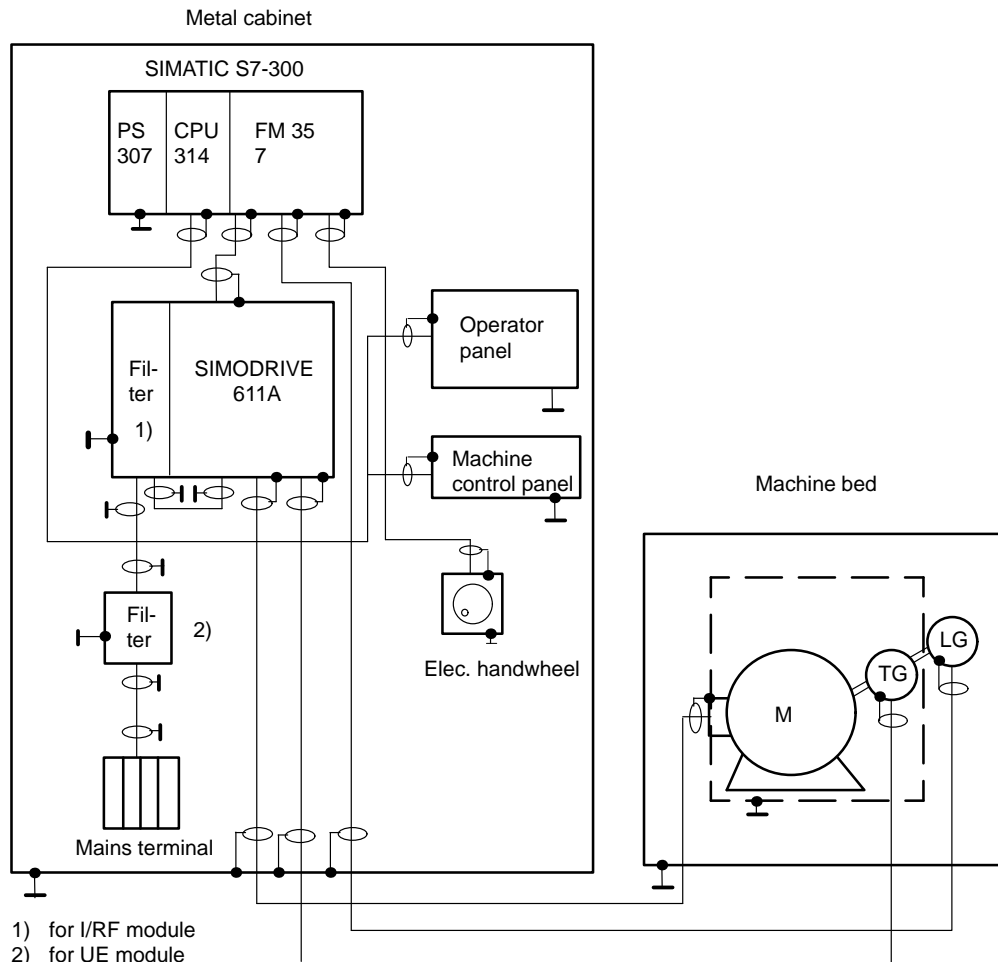
Unterschrift

Der Anhang ist Bestandteil dieser Erklärung.
Diese Erklärung bescheinigt die Übereinstimmung mit der genannten Richtlinie, ist jedoch keine Zusicherung von Eigenschaften im Sinne der Produkthaftung.
Die Sicherheitshinweise der mitgelieferten Produktdokumentation sind zu beachten.

Annex A to EC Declaration of Conformity No. E002 V 21/03/97

A8: Typical system configuration

SIMATIC FM 357 / SIMODRIVE 611A



- All components which are approved in accordance with the ordering document for a combined SIMATIC FM 357 / SIMODRIVE 611A plant meet the requirements of the 89/336/EEC directive when operated together.
- For conformity to standards, please see Annex C

Note

The system configuration sketch shows only the basic measures required for conformity of a typical system configuration with the 89/336/EEC Directive.

In addition, but particularly in cases where the system configuration deviates from the sketch, it is important to observe the installation instructions for proper EMC system design detailed in the product documentation and the EMC Installation Guidelines for SINUMERIK, SIROTEC, SIMODRIVE (Order No.: 6FC5 297-0AD30-0BP0).

Annex C to EC Declaration of Conformity No. E002 V 21/03/97

C: Compliance of the products with the 89/336/EEC directive has been verified by tests performed in accordance with the following basic technical specifications and the basic specifications listed therein:

Basic technical specification: EN 50081-2 Status 8/93

Basic specifications:

EN 55011 1)

Basic technical specification: EN 50082-2 Status 3/95

Basic specifications: Test subject:

ENV 50140	2)	High-frequency radiation
ENV 50141	3)	HF cable interference (amplitude-modulated)
ENV 50204		HF cable interference (pulse-modulated)
EN 61000-4-8	4)	Magnetic fields
EN 61000-4-2	5)	Static discharge
EN 61000-4-4	6)	High-speed transients (bursts)

Specifications also fulfilled:

cf. 1):	VDE 0875 Part 11
cf. 2):	VDE 0847 Part 3
cf. 3):	IEC 801-6
cf. 4):	VDE 0847 Part 4-8
	IEC 1000-4-8
cf. 5):	VDE 0847 Parts 4-2
	EN 60801 Part 2
	IEC 801-2
	VDE 0843 Part 2
cf. 6):	VDE 0843 Part 4
	VDE 0847 Part 4-4
	IEC 801-4



C

Index of Abbreviations

“A/ASBL” mode	“Automatic/Automatic Single Block” mode
“ITR” mode	“Incremental Travel Relative” mode
“J” mode	“Jog” mode
“MDI” mode	“ <u>M</u> anual <u>D</u> ata <u>I</u> nput” mode
“REF” mode	“Reference Point Approach” mode
ASCII	American Standard Code for Information Interchange
ASUB	Asynchronous Subroutine
AT	Advanced Technology
BD	Basic Data
BCD	Binary Coded Decimals
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
DSG	Dimension System Grid

DTE	Data Terminal Equipment
DW	Data Word
EN	Enable (input parameter in ladder diagram)
ENO	Enable Output (output parameter in ladder diagram)
EMC	ElectroMagnetic Compatibility
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
FEPRM	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
K bus	Communication bus

LAD	Ladder Diagram
LED	Light Emitting Diode
MCS	Machine Coordinate System
MDI	<u>M</u> anual <u>D</u> ata <u>I</u> nput
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 Control
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
RFG	Controller enable
ROV	Rapid Override
RPA	R Parameter Active
RPS	Reference Point Switch (cam)
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
S7-300	Programmable logic controller in medium performance range
TC	Tool Change
TEA	Testing Data Active: Refers to machine data
TF	Technology Function
TO	Tool Offset
TO	Tool Offset
TOA	Tool Offset Active
TRC	Tool Radius Compensation
UE	Unregulated supply
UP	User Program
VGA	Video Graphics Array
WCS	Workpiece Coordinate System
ZOA	Zero Offset Active



List of Indices

A

- Absolute dimension G90, 10-14
- Absolute dimension, rotary axes, 10-15
- Absolute encoder (SSI), 4-19
- Absolute encoders (SSI), 9-12, 9-47
 - Parameters, 9-12
- Acceleration, 5-14, 9-26, 9-27, 9-28
 - Brisk acceleration, 9-26
 - Drive acceleration, 9-28
 - Soft acceleration, 9-27
- Acceleration pattern, 5-14, 9-26
- Acceleration patterns, 10-52
- Acceleration rate, 9-24
- Actual velocity monitoring, 9-34
- Alignment, 9-39
 - For absolute encoders, 9-47
 - Parameters, 9-47
- Analog drives, signals, 4-11
- Application examples, 6-62
- Arithmetic parameters
 - Compare operations, 10-71
 - Operators/Arithmetic functions, 10-70
- Asynchronous subroutine (ASUB), 9-66, 10-83
- Automatic, 9-63
- Automatic single block, 9-63
- Auxiliary functions, 6-59
- Axis errors, 11-34
- Axis motions
 - Circular interpolation, 10-36
 - Linear interpolation with feed, 10-34
 - Linear interpolation with rapid traverse, 10-34
 - Positioning motions, 10-35
 - Programming of feedrates, 10-30
- Axis movements, 10-30
- Axis name, 9-4
 - Geometry axis, 9-4
 - Machine axis, 9-4
 - Special axis, 9-4
- Axis number, 9-4
- Axis type, 9-5
- Axis types, 6-60, 10-11
 - Machine axes, 10-12
- Axis velocity, 5-14, 9-25

B

- Backlash compensation, 9-17
 - Parameters, 5-13, 9-18
- Battery compartment, 4-31
- Battery type, 4-32
- Blocks, 6-1
 - Application examples, 6-62
 - FB 1 – basic function, startup section, 6-5
 - FB 2 – read NC variable, 6-16
 - FB 3 – write NC variable, 6-22
 - FB 4 – select program, acknowledge error, 6-27
 - FC 22 – basic functions and operating modes, 6-7
 - FC 24 – positioning of linear and rotary axes, 6-12
 - FC 5 – basic function, diagnostic alarm, 6-31
 - FC 9 – start of asynchronous subroutines, 6-33

C

- CE- marking, ii
- CE-Mark, A-2
- Checkback signals, 6-54
- Circular interpolation, 10-36
- Clamping monitor, 9-33
- Coarse target range, 9-31, 10-47
- Configuration, 9-3
 - Parameters, 9-7
- Connecting cable, 4-5
 - Measuring system cable, 4-5, 4-24
 - MPI cable, 4-5
 - Setpoint cable, 4-5
- Connection data, A-3
- Connection of drive unit, 4-15
- Continuous-path mode, 10-49
- Control signals, 6-48
- Coordinate systems, 10-10
- COROS devices (operator panels), 8-3
- Coupled motion, 9-69
- Creep acceleration, 5-14, 9-28

Creep velocity, 5-14, 9-28
CSA-Approval, A-1
Current control loop, 9-23
Curve table, 9-80, 10-108
 Non periodic, 10-109
 Parameters, 5-18, 9-80, 9-81
 Periodic, 10-109
Cycle time, 9-3

D

Diagnostic errors, 11-10
Digital inputs
 On local P bus, 9-53
 On-board inputs, 4-26, 9-52, A-4
Digital outputs, On local P bus, 9-53
Dimensions, 10-20
Dimensions of FM 357, A-3
Direction reversal actual value, 9-17
Distributed I/Os, 1-4
Drift compensation, 9-22
Drift limit value, 9-22
Drive, 9-6
Drive interface, assignments, 4-10
Dwell time, 10-54

E

EMC Guidelines, 4-1
EMERGENCY STOP, 6-51, 9-97
 Parameters, 5-14, 9-97
 Sequence, 9-97
Encoder, 4-19
 Absolute encoder, 4-19
 Connecting encoders, 4-23
 Incremental encoder, 4-19
Encoder inputs, A-3
Encoder supply, 4-21
Encoders, 9-8
 Absolute encoders, 9-12
 Incremental encoders, 9-10
 Parameters, 9-9
 Selection, 9-8
Error list
 Axis errors, 11-34
 Diagnostic errors, 11-10
 General errors, 11-13
Error lists, 11-9, 11-48
Error messages, Display by LEDs, 11-3
Error messages and their effect, 11-7
External master value, 9-6

F

Feed interpolation, 10-31
Field of application, 1-1, A-2
Fine target range, 9-31, 10-47
Firmware, 3-4
Firmware update, 3-4
 Central configuration, 3-4
 Distributed configuration, 3-5
FM 357 power-up, 7-3
FM STEPDRIVE, connection, 4-16
FM-Approval, A-2
Following error monitoring, 9-32
Front connector, 4-5
Front-panel elements, 1-9
 LED displays, 1-9

G

Gantry, 9-72
 Initial start-up, 9-77
 Interface signals, 9-73
 Parameters, 5-19, 9-72
General errors, 11-13
Geometry axis, 9-4, 10-12

H

H functions, 10-66
 Output, 9-50

I

Incremental dimension G91, 10-14
Incremental encoder, 4-19
Incremental encoders, 9-10, 9-41
 Parameters, 9-11
Incremental travel relative, 9-62
Installation of the FM 357, 3-3
Interface signals
 Axis signals, 6-44, 6-52, 6-55, 6-57, 6-58
 NC signals, 6-37, 6-48, 6-54, 6-57, 6-58, 6-59

-
- Interfaces, 1-9, 4-9, 4-19, 4-25
 - Drive -interface, 1-9, 4-9
 - I/O device -interface, 1-9
 - I/O-interface, 4-25
 - Measuring system interface, 1-9
 - Measuring system-interface, 4-19
 - Memory submodule interface, 1-9
 - Power supply connection, 1-9, 4-6
 - SIMATIC bus link- interface, 1-9

 - J**
 - Jerk, 5-14, 9-27
 - Jerk filter, 9-16
 - Parameters, 5-13
 - Jog , 9-62

 - K**
 - Kv factor, 9-19

 - L**
 - Limit signals (software cams), Parameters, 5-17
 - Limit switching signals (software cams), 9-56
 - Generation, 9-59
 - Output, 9-61
 - Parameters, 9-56
 - Linear axes, 9-5
 - Linear interpolation with feed, 10-34
 - Linear interpolation with rapid traverse, 10-34
 - List-based parameterization, 5-20
 - Local P bus, 1-4

 - M**
 - M functions, 10-64
 - Options, 9-49
 - Machine axis, 9-4, 10-12
 - Machine data (parameters), 5-9
 - Value ranges, 5-11
 - Man/machine interface, 8-1, 8-3
 - Master axis, 9-78
 - Master value coupling, 9-78, 10-108
 - Curve table, 9-80, 10-108
 - Master axis, 9-78
 - Parameters, 9-78, 9-79, 9-84
 - Slave axis, 9-78
 - System variables, 10-111
 - Master value link, Parameters, 5-18
 - Maximum velocity, 9-24
 - MDI, 9-63
 - Measurement, 9-86
 - Measurement (programming), 10-56
 - Axial (MEAS, MEAW), 10-58
 - Block-specific (MEAS, MEAW), 10-56
 - Modulo rotary axes, 9-5
 - Monitoring functions, 9-30
 - Actual velocity, 9-34
 - Clamping operation, 9-33
 - Coarse target range, 9-31
 - Encoders, 9-35
 - Fine target range, 9-31
 - Following error, 9-32
 - Hardware limit switch, 9-37
 - Monitoring time, 9-31
 - Software limit switch, 9-37
 - Speed setpoint, 9-33
 - Motion control, 2-1
 - Motion coupling, 10-54
 - Motion synchronous actions
 - Principle, 10-103
 - Structure, 10-87
 - Multi-tier configuration, 1-4

 - N**
 - NC program execution, 9-64
 - NC programming, 10-1
 - Block format, 10-6
 - Program structure, 10-3
 - Special characters, 10-9
 - Statements, 10-4
 - NC-READY output, 4-27
 - NC-VAR selector, 6-4

O

- Offset compensation, 9-21
- OP 17 menu tree, 8-4
- Operating modes, 9-62
- Optimization, 7-6
- Oscillation, 10-104
- Overlaid motion in synchronized actions, 9-84
 - Parameters, 5-18, 9-85
- Override, 6-50
- Override coding, 9-4

P

- Parameterization, 5-1
 - List-based parameterization, 5-20
 - Machine data (parameters), 5-9
 - Menus, 5-23
 - Parameterization Wizard, 5-10
 - User data, 5-21
- Parameterization data, 5-7
 - Offline editing, 5-8
 - Online editing, 5-7
- Parameterization Wizard, 5-10
- Path acceleration, 5-14, 9-29
- Path action, 9-29, 10-46
 - Acceleration patterns, 10-52
 - Continuous-path mode, 10-49
 - Exact stop, 10-47
 - Programmable acceleration, 10-53
 - Target range, 10-47
- Path axes, 10-12
- Path jerk, 5-14, 9-29
- Path override, 6-50
- Plane selection, 10-21
- Position control, 9-15
- Position control gain, 9-19
 - Parameters, 9-19
- Position of interfaces, 1-8
- Positioning axes, 10-12
- Positioning velocity, 5-14, 9-24
- Power supply, 4-6
- Program jumps, 10-78
- Programmable acceleration, 10-53
- Programming
 - NC programs, 10-1
 - User program, 6-1
- Programming a polar coordinate, 10-17
- Programming of feedrates, 10-30

R

- R parameters (arithmetic parameters), 10-69
- Rapid traverse override, 5-14, 9-25
- Read data block, 6-57
- Reference point approach, 9-62
- Reference point switch, 9-41
- Referencing, 9-39
 - Incremental encoders, 9-41
 - Parameters, 5-15, 9-43
 - Stepper motor without encoder, 9-46
 - With RPS, 9-41
 - Without RPS, 9-42
- Removal and replacement of the FM 357, 3-6
- Rotary axes, 9-5
- Rotation monitoring, 9-36

S

- Safety rules, 4-1
 - EMERGENCY -STOP- devices, 4-1
- Service data, 7-7
- Servo drive, 9-6
- Setting an actual value, 10-29
- Signal connections on stepper motor interface, 4-14
- SIMATIC Manager, 5-4
- SIMODRIVE 611 connection, 4-15
- Single-tier configuration, 1-4
- Slave axis, 9-78
- Software cams, 9-56
 - Parameters, 5-17, 9-56
- Spatially distributed arrangement, 1-4
- Special axis, 9-4, 10-12
- Speed feedforward control, 5-17, 9-23
- Speed feedforward control (FFWON, FFWOF), 10-112
- Speed setpoint monitor, 9-33
- Spline, 10-40
- Standard function blocks, 6-2
- Start-up, 7-6
- Start-up switch, 7-3
- Startup switch, 1-8
- Statements, 10-4
 - Overview, 10-113
- Stepper drive, 9-6
 - Signals, 4-12

Stepper motor, 9-6, 9-14
 Parameters, 5-12, 9-14
 With/without encoder, 9-6
 Stop preprocessor, 10-62
 Subroutine system, 10-80
 Synchronized actions, 10-87
 Synchronized axes, 10-12
 Synchronous actions
 Operators, 10-97
 System variables, 10-98
 System of measurement, 9-3
 System overview, 1-5
 Components, 1-5
 Data handling, 1-7
 System variable, 10-72

T

T function, 10-67
 Options, 9-50
 Technical data, 6-66
 Testing, 7-6
 Axis, 7-10
 Time constant , 9-23
 Tool offset values, 10-67
 Trace, 7-8
 Travel direction reversal, 9-20
 Parameters, 9-20
 Travel to fixed stop, 9-88, 10-60
 Clamping torque, 10-61
 Clock pulse diagrams, 9-94
 Monitoring window, 10-61
 Parameters, 5-19, 9-89
 Traversing characteristics, Positioning axes,
 10-51
 Troubleshooting , 7-7

U

UL-Approval, A-1
 User data, 5-21
 User data blocks, 6-36
 Axis signals, 6-44
 NC signals, 6-37
 User handling procedures for controlling axes,
 6-60

V

VDI output, 9-6
 Velocities, 9-24
 Velocity assignment, 9-20, 9-21
 Servo drive, 9-20
 Parameters, 9-20
 Stepper motor, 9-21
 Parameters, 9-21

W

Weight, A-3
 Weighting factor, 9-23
 Wiring diagram of an FM 357, 4-3
 Wiring of FM 357, 4-1
 Wiring of front connector, 4-28
 Working area limitations, 10-62
 Write data block, 6-58

Z

Zero offset, 10-22
 Zero offsets
 Programmable, 10-24
 Settable, 10-22

