SIEMENS

SIMOTION

Motion Control
TO Axis Electric / Hydraulic, External Encoder

Function Manual

Preface

Fundamental safety instructions 1

Part I Axis - Overview 2

Axis fundamentals 3

Configuring an axis 4

Support for SINAMICS Safety Integrated Functions 5

Part II Hydraulic Functionality 6

Fundamentals of hydraulic functionality 7

Part III Programming/ Reference 8

Part IV External Encoder - Description 9

External Encoder Fundamentals 10

Programming External Encoders/Reference 11

Valid as of Version 5.4

07/2021
A5E33428123B

Legal information

Warning notice system

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

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

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

⚠ **CAUTION**
indicates that minor personal injury can result if proper precautions are not taken.

NOTICE
indicates that property damage can result if proper precautions are not taken.

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

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

⚠ **WARNING**
Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

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

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E33428123B
Ⓟ 05/2021 Subject to change

Copyright © Siemens AG 2016 - 2021.
All rights reserved

Preface

Content

This document is part of the **System and Function Descriptions** documentation package.

Scope of validity

This manual is valid for SIMOTION SCOUT product version V5.4:

- SIMOTION SCOUT V5.4 (engineering system of the SIMOTION product family)
- SIMOTION Kernel V5.4, V5.3, V5.2, V5.1, V4.5, V4.4, V4.3, V4.2, V4.1, V4.0
- SIMOTION technology packages CAM, CAM_ext and TControl in the version adapted to the respective kernel.

Sections in this manual

The following is a list of sections included in this manual along with a description of the information presented in each section.

Part I Axis

- **Overview**
  This section provides the user with an overview of the Axis technology object.
- **Axis Fundamentals**
  This section explains the basic setting options and functions of the Axis Technology Object.
- **Configuring an Axis**
  This section explains the configuration procedure with reference to various tasks.
- **Support for SINAMICS Safety Integrated Functions**
  This section describes the SIMOTION-side support for safety functions for SINAMICS drives.

Part II Hydraulic Functionality

- **Overview**
  This section provides the user with an overview of the hydraulic functionality of the Axis technology object.
- **Fundamentals of Hydraulic Functionality**
  This section explains the basic setting options and functions concerning the hydraulic functionality of the Axis technology object.

Part III Programming / Reference

- This section explains the commands and functions in greater detail.

Part IV External Encoder

- **Description**
  This section provides the user with an overview of the External Encoder technology object.
- **External Encoder Fundamentals**
  This section explains the basic setting options and functions of the External Encoder technology object.
- **Configuring an External Encoder (online help only)**
  This section explains the configuration procedure with reference to various tasks.
- **Programming External Encoders Reference**
  This section explains the commands and functions in greater detail.

Index

- Keyword index for locating information

SIMOTION Documentation

An overview of the SIMOTION documentation can be found in the SIMOTION Documentation Overview document.

This documentation is included as electronic documentation in the scope of delivery of SIMOTION SCOUT. It comprises ten documentation packages.

The following documentation packages are available for SIMOTION product version V5.4:

- SIMOTION Engineering System Handling
- SIMOTION System and Function Descriptions
- SIMOTION Service and Diagnostics
- SIMOTION IT
- SIMOTION Programming
- SIMOTION Programming - References
- SIMOTION C
- SIMOTION P
- SIMOTION D
- SIMOTION Supplementary Documentation

Hotline and Internet addresses

SIMOTION at a glance

We have compiled an overview page from our range of information about SIMOTION with the most important information on frequently asked topics - which can be opened with only one click.

Whether beginner or experienced SIMOTION user – the most important downloads, manuals, tutorials, FAQs, application examples, etc. can be found at

https://support.industry.siemens.com/cs/ww/en/view/109480700

Additional information

Click the following link to find information on the following topics:

- Documentation overview
- Additional links to download documents
- Using documentation online (find and search manuals/information)

https://support.industry.siemens.com/cs/ww/en/view/109479653

My Documentation Manager

Click the following link for information on how to compile documentation individually on the basis of Siemens content and how to adapt it for the purpose of your own machine documentation:

https://support.industry.siemens.com/My/ww/en/documentation

Training

Click the following link for information on SITRAIN - Siemens training courses for automation products, systems and solutions:

http://www.siemens.com/sitrain

FAQs

Frequently Asked Questions can be found in SIMOTION Utilities & Applications, which are included in the scope of delivery of SIMOTION SCOUT, and in the Service&Support pages in **Product Support**:

https://support.industry.siemens.com/cs/de/en/ps/14505/faq

Technical support

Country-specific telephone numbers for technical support are provided on the Internet under **Contact**:

https://support.industry.siemens.com/cs/ww/en/sc/2090

Table of contents

Preface ... **3**

1 Fundamental safety instructions ... **15**

1.1 General safety instructions ... 15
1.1.1 Malfunctions of the machine as a result of incorrect or changed parameter settings ... 15
1.2 Security information ... 15
1.3 Note regarding the general data protection regulation ... 16
1.4 Danger to life due to software manipulation when using removable storage media ... 16

2 Part I Axis - Overview ... **17**

2.1 General information about axes ... 17

3 Axis fundamentals ... **21**

3.1 Axis technologies ... 21
3.1.1 Overview of axis technologies ... 21
3.1.2 Axis-drive relationship ... 25
3.2 Axis types ... 26
3.2.1 Overview of axis types ... 26
3.2.2 Setting for the electric axis type ... 27
3.2.3 Setting for the hydraulic axis type ... 29
3.2.4 Setting for the virtual axis type ... 30
3.2.5 TypeOfAxis configuration data ... 31
3.3 Units and accuracies ... 33
3.4 Axis settings / drive assignment ... 36
3.4.1 Overview of how to create an axis ... 36
3.4.2 Real and virtual axes ... 36
3.4.3 Setting as a real axis with analog drive link ... 38
3.4.4 Setting as a real axis with digital drive coupling ... 39
3.4.4.1 Symbolic assignment and adaptation (as of V4.2) ... 39
3.4.4.2 Adapted data (SINAMICS S120 as of V2.6.2) ... 42
3.4.4.3 Communication and control word / status word ... 45
3.4.4.4 Specific removal of drive enables in the event of a fault ... 49
3.4.4.5 Brake control ... 50
3.4.4.6 Specify control bits via the application (without TO check) [expert] ... 51
3.4.4.7 Status diagram - PROFIdrive General State Diagram ... 53
3.4.4.8 SIMOTION system behavior with a drive-autonomous OFF3 ramp ... 54
3.4.4.9 Technologies and telegram types ... 55
3.4.4.10 Coupling of digital drives ... 56
3.4.5 Setting as a real axis with stepper drive C2xx (V3.2 and higher) ... 58
3.4.6 Stepper drives on IM174 and stepper drives with PROFIBUS interface ... 58
3.4.7 Setting as a real axis with encoder signal simulation (V4.0 and higher) ... 59
3.4.8 Setting a non-exclusive drive assignment (as of V4.1 SP1) ... 59
3.4.9 Setting as a real axis without drive (axis simulation) ... 60

3.5 Encoders and encoder parameters ... 62
3.5.1 Overview of encoders and encoder parameters ... 62
3.5.2 Encoder for position ... 62
3.5.3 Encoder for velocity ... 64
3.5.4 Encoder assignment and terminology ... 64
3.5.5 Encoder list ... 67
3.5.6 Onboard encoder interface on SIMOTION C2xx ... 67
3.5.7 Encoder interface using the PROFIdrive message frame ... 68
3.5.8 Encoder interface as a direct value in the I/O area ... 72
3.5.9 Encoder value via system variable ... 77
3.5.10 Non-exclusive encoder assignment (as of V4.1 SP1) ... 78
3.5.11 Diagnostic features ... 78

3.6 Input limits, technological limiting functions ... 79

3.7 Setting for axis and encoder mechanics ... 79
3.7.1 Overview of setting options for axis and encoder mechanics ... 79
3.7.2 Inversion of actual position value ... 80
3.7.3 Boundary conditions for mechanical settings for modulo axes (long-term stability / long-term accuracy) ... 81

3.8 Defaults ... 84

3.9 Homing ... 84
3.9.1 Overview of homing ... 84
3.9.2 Terminology ... 85
3.9.3 Homing types ... 86
3.9.4 Active homing ... 87
3.9.5 Passive homing/on-the-fly homing ... 93
3.9.6 Direct homing/setting the home position ... 94
3.9.7 Relative direct homing/relative setting of home position (V3.2 and higher) ... 94
3.9.8 States that require a new homing procedure for incremental encoders ... 95
3.9.9 Absolute encoder homing / absolute encoder adjustment ... 95
3.9.10 Homing mark monitoring ... 97
3.9.11 Homing output monitoring ... 98
3.9.12 Displaying actual value change during homing (V4.0 and higher) ... 98
3.9.13 Traversing with a non-homed axis ... 98
3.9.14 Correcting the actual position/set position without homing ... 98
3.9.15 Differential position measurement (V3.2 and higher) ... 99

3.10 Monitoring/limiting functions ... 100
3.10.1 Overview of monitoring/limiting functions (block diagram) ... 100
3.10.2 Dynamic monitoring of following errors ... 100
3.10.3 Positioning and standstill monitoring ... 102
3.10.4 Standstill signal ... 104
3.10.5 Manipulated variable monitoring ... 105
3.10.6 Manipulated variable limiting (backstop) (V3.1 and higher) ... 106
3.10.7 Hardware limit monitoring ... 107
3.10.8 Software limit monitoring ... 108
3.10.9 Encoder limit frequency monitoring ... 109
3.10.10 Velocity error monitoring ... 110
3.10.11 Measuring system differential/slip monitoring ... 110

3.11 Positioning axis with position control ... 110
3.11.1 Overview of positioning axis with position control ... 110

3.11.2 Position control ... 111
3.11.3 Dynamic Servo Control (DSC) ... 118
3.11.4 DSC with spline (as of V4.4) [Expert] ... 122
3.11.5 Fine interpolation ... 126
3.11.6 Dynamic controller data ... 127
3.11.7 Setpoint superimposition ... 128
3.11.8 Dynamic response adaptation ... 129
3.11.9 Actual value measurement / actual value system ... 130
3.11.10 Preparation of manipulated variables for electric axis ... 136
3.11.11 Manipulated variable superimposition ... 137
3.11.12 Manipulated variable filtering (as of V4.1 SP1) ... 137
3.11.13 Drift/offset compensation ... 138
3.11.14 Static friction compensation ... 138
3.11.15 Backlash on reversal compensation ... 139
3.11.16 Traversing of the positioning axis without position control ... 142
3.11.17 Stepper drives ... 143
3.11.18 Encoder signal output (V4.0 or later) ... 145

3.12 Commissioning the position controller of positioning axes ... 147
3.12.1 Overview of commissioning the position controller of positioning axes ... 147
3.12.2 Configuration data ... 148
3.12.3 Example of commissioning a proportional-action controller with precontrol ... 149

3.13 Command variable calculation ... 153
3.13.1 Velocity profiles ... 153
3.13.2 Defining accelerations and decelerations ... 156
3.13.3 Override ... 157
3.13.4 Default settings for dynamic response parameters ... 158
3.13.5 Dynamic limiting functions ... 160
3.13.6 Stopping with preassigned braking ramp ... 163
3.13.7 Traversing the axis via velocity specifications ... 163
3.13.8 Positioning ... 164
3.13.9 Positioning with blending ... 164
3.13.10 Superimposed positioning ... 166
3.13.11 Traversing with specific motion profiles ... 167
3.13.12 Traversing according to motion vectors (V3.2 and higher) ... 167
3.13.13 Jerk limitation for local stop response (V3.2 or higher) ... 168

3.14 Superimposed motion ... 169

3.15 Torque limiting via torque reduction ... 171
3.15.1 Overview of torque limiting via torque reduction ... 171
3.15.2 Conversion of torque/force ... 176

3.16 Travel to fixed endstop ... 176

3.17 Technology data ... 179

3.18 Torque limiting B+/B- (V3.2 and higher) ... 182

3.19 Additive set torque (V3.2 and higher) ... 185

3.20 Force/pressure control ... 187
3.20.1 Overview of force/pressure control ... 187
3.20.2 Configuration of the actual force/pressure value sensors ... 189
3.20.3 Controller for force/pressure control ... 190

3.20.4 Monitoring functions / limiting functions / emergency strategies with active force/ pressure control ... 192
3.20.5 Activating the force/pressure control ... 193
3.20.6 Force/pressure setpoint specification ... 195
3.20.7 Commissioning procedure for force/pressure control ... 195
3.20.8 Force/pressure control with velocity limiting ... 195

3.21 Force/pressure limiting ... 196
3.21.1 Overview of force/pressure limiting ... 196
3.21.2 Positioning with active force/pressure limiting (V3.2 and higher) ... 198
3.21.3 Increase limiting for pressure profiles and pressure limiting (V3.2 and higher) ... 199

3.22 Data sets ... 200
3.22.1 Data set overview ... 200
3.22.2 Data set switchover / encoder switchover ... 200

3.23 Traversing with user-defined motion and force/pressure profiles ... 203
3.23.1 Overview of traversing with user-defined motion and force/pressure profiles ... 203
3.23.2 Profile reference ... 205
3.23.3 Profile types ... 205
3.23.4 Behavior at the end of the profile (V3.2 and higher) ... 208

3.24 Motion commands ... 208
3.24.1 Motion execution/interpolator ... 208
3.24.2 Command groups ... 210
3.24.3 Changing motion commands into the interpolator ... 216
3.24.4 Motion transitions ... 216
3.24.5 Conditions for command advance ... 217
3.24.6 State model/axis status ... 218

3.25 Data exchange between Axis technology object and DCC ... 221

3.26 Drive communication based on DPV1 services ... 222

3.27 Links/interconnections to other technology objects ... 223

4 Configuring an axis ... 225

4.1 Overview of axis configuration ... 225

4.2 Linking digital drives ... 225
4.2.1 Overview of linking digital drives ... 225
4.2.2 Configuring PROFIBUS DP in HW Config to optimize run-time ... 226

4.3 Linking analog drives to SIMOTION ... 226

4.4 Axis with stepper motor connection ... 228

4.5 Using the expert list for an axis ... 229

4.6 Automatic controller setting ... 230
4.6.1 Overview of automatic controller setting (as of V4.1 SP1) ... 230
4.6.2 Automatic speed controller setting (as of V4.1 SP1) ... 231
4.6.3 Automatic position controller setting (as of V4.1 SP1) ... 235

4.7 SIMOTION measuring functions ... 238

4.8 Axis control panel ... 243

5 Support for SINAMICS Safety Integrated Functions 245

5.1 Overview - Support of SINAMICS Safety Integrated Functions on the Axis technology object 245

5.2 Communication 248

5.3 Main procedure for safety configuration with SIMOTION 249

5.4 Safety channel and safety channel types (when symbolic assignment is deactivated) 251

5.5 Safety licenses 253

5.6 Standard (DSDB) as of SIMOTION V4.4 253
5.6.1 Display of SINAMICS Safety Integrated Functions in SIMOTION 253
5.6.2 Axis configuration 254
5.6.3 Safety status information (part of the DSDB) 254

5.7 Compatibility mode (SIDB) as of SIMOTION V4.1 256
5.7.1 Safety Information Data Block (SIDB) 257
5.7.2 Safety status information (part of the SIDB) 259
5.7.3 Display of SINAMICS Safety Integrated Functions in SIMOTION 261
5.7.4 Axis configuration 261

5.8 Behavior and user responses 262
5.8.1 Overview 262
5.8.2 Overview of safety functions 263
5.8.3 Safe Torque Off (STO)As of SIMOTION V4.1 264
5.8.4 Safe Stop 1 (SS1)As of SIMOTION V4.1 264
5.8.5 Safe Stop 2 (SS2)As of SIMOTION V4.1 266
5.8.6 Safe Operating Stop (SOS)As of SIMOTION V4.1 267
5.8.7 Safely-Limited Speed (SLS)As of SIMOTION V4.1 267
5.8.8 Safe Direction (SDI)As of SIMOTION V4.3 269
5.8.9 Safe Position (SP) 269
5.8.10 Safely-Limited Position (SLP)As of SIMOTION V4.4 270
5.8.11 Safe Cam (SCA) 271
5.8.12 SLA - Safely-Limited Acceleration (as of SIMOTION V5.2) 272

5.9 Safety brake test as of SIMOTION V4.4 272

5.10 Pulse enable evaluation (when standard settings are deactivated) 273

5.11 Messages and alarms 274

5.12 SINAMICS Safety Message Buffer 276

5.13 Isochronous PROFIBUS operation 277

5.14 Restrictions after switching from compatibility mode (SIDB) to standard (DSDB) 278

6 Part II Hydraulic Functionality 281

6.1 Hydraulic functionality overview 281

7 Fundamentals of hydraulic functionality 283

7.1 Axis settings / drive assignment 283
7.1.1 Overview of axis settings / drive assignment 283
7.1.2 Setting as a real axis with hydraulic functionality 283
7.1.3 Setting as a real axis with Q valve only 285
7.1.4 Setting as a real axis with Q valve + P valve/F output 287

7.1.5 Setting an axis as a real axis with P valve only (V3.2 and higher) ... 289
7.1.6 Setting an axis as a real hydraulic axis without a valve (axis simulation) ... 290
7.2 Input limits, technological limiting functions ... 291
7.3 Settings for axis and encoder mechanics ... 291
7.4 Defaults ... 291
7.5 Homing ... 291
7.6 Differential pressure measurement (V3.2 and higher) ... 291
7.7 Differential position measurement (V3.2 and higher) ... 293
7.8 Monitoring/limiting functions ... 293
7.9 Motion profiles ... 293
7.10 Hydraulic axis with position control/velocity control ... 293
7.10.1 Position control for setting a positioning axis with hydraulic functionality ... 293
7.10.2 Velocity controller when setting speed-controlled axis with hydraulic functionality ... 295
7.10.3 Preparation of manipulated variables for axis with hydraulic functionality ... 297
7.10.4 Manipulated variable filtering (as of V4.1 SP1) ... 298
7.10.5 Compensations that are active only on the axis with hydraulic functionality ... 298
7.10.6 Consideration of valve characteristic when specifying a hydraulic axis ... 300
7.10.7 Access to the same final controlling element from multiple axes ... 303
7.11 Travel to fixed endstop ... 304
7.12 Force/pressure control with hydraulic axes with Q valve only ... 304
7.13 Force/pressure limiting with hydraulic axes with Q valve only ... 304
7.14 Force/pressure limiting with hydraulic axes with P valve ... 305
7.15 Force/pressure control with hydraulic speed-controlled axes with Q valve only ... 305
7.16 Force/pressure limiting with hydraulic speed-controlled axes with Q valve only (V4.0 and higher) ... 306
7.17 Velocity limiting with hydraulic axes ... 307

8 Part III Programming/Reference ... 309

8.1 Overview of commands ... 309
8.1.1 Overview of commands ... 309
8.1.2 Commands for specification and control of motion on the axis ... 309
8.1.3 Command properties ... 312
8.2 Enables, stop and continue commands, resets ... 313
8.2.1 Setting and canceling the axis enables ... 313
8.2.2 Enabling force/pressure control depending on switchover conditions ... 321
8.2.3 Stopping motions with _stopEmergency() ... 322
8.2.4 Stopping motions with _stop() ... 323
8.2.5 Stopping position-controlled axes in speed-controlled mode (V3.1 and higher) ... 324
8.2.6 Resuming motions ... 325
8.2.7 Reset axis ... 325
8.2.8 Resetting an axis error ... 327
8.2.9 Canceling/deleting an axis command (as of V4.1 SP1) ... 327
8.3 Commands for axis motions ... 327

8.3.1 Homing ... 327
8.3.2 Moving ... 329
8.3.3 Positioning ... 330
8.3.4 Starting a time-related velocity profile ... 331
8.3.5 Starting a position-related velocity profile ... 332
8.3.6 Positioning with user-definable position profile ... 333
8.3.7 Enabling and disabling torque limiting ... 333
8.3.8 Enable force/pressure limiting with position-related force/pressure limiting profile ... 334
8.3.9 Enabling force/pressure limiting with time-related motion profile ... 335
8.3.10 Starting the time-related force/pressure profile ... 335
8.3.11 Starting the position-related force/pressure profile ... 336
8.3.12 Enabling/disabling force/pressure limiting ... 337
8.3.13 Enabling/disabling velocity limiting ... 338
8.3.14 Enabling velocity limiting with position-related velocity limiting profile ... 338
8.3.15 Enabling velocity limiting with time-related velocity limiting profile ... 339
8.3.16 Travel to fixed endstop ... 339

8.4 Commands for defining the coordinate system ... 339
8.4.1 Resetting the set and actual positions ... 339

8.5 Simulation commands ... 341
8.5.1 Enabling/disabling program simulation ... 341

8.6 Information functions / command buffers ... 343
8.6.1 Overview of information functions/command buffer ... 343
8.6.2 Reading the execution status of a motion command ... 343
8.6.3 Reading current phase of motion ... 344
8.6.4 Saving the Command ID ... 345
8.6.5 Canceling saving of Command ID ... 345
8.6.6 Reading the status of a specific error on the axis ... 345
8.6.7 Reading out pending alarms (V4.0 and higher) ... 345
8.6.8 Reading the status of the motion buffer on the axis ... 345
8.6.9 Clearing the motion buffer on the axis ... 346
8.6.10 Activating data sets ... 346
8.6.11 Writing a data set ... 347
8.6.12 Reading a data set ... 347
8.6.13 Writing the force/pressure-specific data of a data set ... 347
8.6.14 Reading the force/pressure-specific data of a data set ... 348
8.6.15 Writing the data of the data set (hydraulic functionality only) ... 348
8.6.16 Reading the force/pressure-specific data of the data set (hydraulic functionality only) ... 348
8.6.17 Command for calculating a braking distance ... 349

8.7 Assigning automatic controller optimization ... 350

8.8 Technological alarms ... 353
8.8.1 Alarm reactions ... 353
8.8.2 Adjustable response to RELEASE_DISABLE ... 357
8.8.3 Toleration of the failure of an encoder not involved in closed-loop control (V4.0 and higher) ... 357

9 Part IV External Encoder - Description ... 359

9.1 External encoder overview ... 359

10 External Encoder Fundamentals ... 361

10.1 Actual values for the external encoder technology object ... 361

10.2 Encoder mounting type ... 362
10.3 Encoder for position ... 362
10.4 Encoder for velocity ... 364
10.5 Encoder assignment and terminology ... 364
10.6 Encoder list ... 367
10.7 Onboard encoder interface on SIMOTION C2xx ... 367
10.8 Encoder interface using the PROFIdrive message frame ... 368
10.9 Encoder interface as a direct value in the I/O area ... 372
10.10 Actual value system ... 377
10.11 Overflow bit for modulo counting ... 377
10.12 Actual value smoothing ... 377
10.13 Actual value extrapolation ... 378
10.14 Standstill signal ... 378
10.15 Monitoring functions ... 378
10.16 Synchronization / Homing ... 379
10.16.1 Overview of synchronization / homing ... 379
10.16.2 Synchronization/homing with incremental encoders ... 379
10.16.3 Synchronization/homing with absolute encoders ... 380
11 Programming External Encoders/Reference ... 383
11.1 Commands ... 383
11.2 Technological alarms ... 384
11.2.1 Possible alarm reactions ... 384
Index ... 385

Fundamental safety instructions 1

1.1 General safety instructions

⚠ WARNING

Danger to life if the safety instructions and residual risks are not observed

The non-observance of the safety instructions and residual risks stated in the associated hardware documentation can result in accidents with severe injuries or death.

- Observe the safety instructions given in the hardware documentation.
- Consider the residual risks for the risk evaluation.

1.1.1 Malfunctions of the machine as a result of incorrect or changed parameter settings

⚠ WARNING

Malfunctions of the machine as a result of incorrect or changed parameter settings

As a result of incorrect or changed parameterization, machines can malfunction, which in turn can lead to injuries or death.

- Protect the parameterization against unauthorized access.
- Handle possible malfunctions by taking suitable measures, e.g. emergency stop or emergency off.

1.2 Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial security measures that may be implemented, please visit (https://www.siemens.com/industrialsecurity).

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customers' exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed visit (https://www.siemens.com/industrialsecurity).

1.3 Note regarding the general data protection regulation

Siemens observes the principles of data protection, in particular the principle of data minimization (privacy by design). For this product this means:

The product does not process / store any personal data, only technical functional data (e.g. time stamp). If the user links this data with other data (e.g. shift plans) or stores personal data on the same medium (e.g. hard disk) and thus establishes a personal reference, the user must ensure compliance with data protection regulations.

1.4 Danger to life due to software manipulation when using removable storage media

⚠ WARNING

Danger to life due to software manipulation when using removable storage media

The storage of files on removable storage media involves a high risk of infection, e.g. via viruses or malware. Incorrect parameter assignment can cause machines to malfunction, which can lead to injuries or death.

- Protect the files on removable storage media against harmful software through appropriate protective measures, e.g. virus scanners.

Part I Axis - Overview

2

2.1 General information about axes

Overview of available axis types

The instantiable **Axis technology object (TO)** is downloaded to the SIMOTION device with a technology package, thus providing the function for activating and monitoring an actuator (drive, motor, valve, etc.). As of V3.2, the **Axis technology object (TO)** is included in the Cam, Path, and Cam_ext technology packages.

The functionality is set by configuration and parameter assignment / programming.

The Axis technology object can be applied to an axis with an electric drive (axis) or an axis with a hydraulic final controlling element / valve (hydraulic functionality).

Any number of axes may be generated, based on the CPU processing power.

When programming in SIMOTION SCOUT (e.g. with MCC), an Axis technology object can be accessed via system functions or system variables. For example, to traverse an axis at a specified velocity to a certain position, you specify the velocity and position via system functions. All other functions (e.g. monitoring of limit values) are specified via the configuration data and system variables of the Axis technology object.

The following axis technologies are distinguished during the configuration phase:

- **Speed-controlled axis**
  Motion control is performed using a speed specification without position control.
  Choosing this axis technology provides the minimum functionality for an axis.
  The speed-controlled axis is referred to by the data type **driveAxis** in reference lists and when programming.
- **Positioning axis**
  Motions are position-controlled.
  The position axis is referred to by the data type **posAxis** in reference lists and when programming.

- **Synchronous axis**
  The synchronous axis creates a grouping of the following axis and synchronous object. Functions such as master value coupling, synchronization and desynchronization, and gearing and camming are provided via the synchronous object. The synchronous object can be interconnected with different master values.
  See the Technology Objects Synchronous Operation, Cam manual for information about using the synchronous axis.
  The following axis is referred to by the data type **followingAxis** and the synchronous object by **followingObjectType** in reference lists and when programming.
- **Path axis**
  The path axis type can be interconnected with a path object.
  The path object can be used to calculate and traverse a linear, circular or polynomial path in the 2D/3D coordinate system for at least two path axes and up to three path axes. A synchronous axis can be traversed in parallel to this.
  The Technology Object Path Object manual describes how to use the path axis with the path object.
  The path axis is referred to by the data type **_pathAxis** and the path object by the data type **_pathObjectType** in reference lists and when programming.

All axis types can also be configured as **virtual axes**, i.e. they do not control a real drive but are used as auxiliary axes for calculations, e.g. as a virtual master axis for several slave axes (line shaft).

Axis operation in SIMOTION

- Any necessary settings can be made by means of configuration data on the axis.
- States are indicated, and standard values and settings can be read and entered by means of system variables on the axis.
- Axis motion sequences are specified by means of motion commands on the axis. The user program can be used to check the motion status at any time and to control specific aspects of the motion. Motions can be aborted, overridden, appended, or superimposed.
- Errors and technological alarms are indicated using alarms on the axis.

Advanced functions

Advanced functions on the axis include path interpolation, synchronous operation, measuring inputs, and output cams. For more information refer to the manuals: Technology Object Path Object, Technology Object Synchronous Operation, and Technology Objects for Output Cams and Measuring Inputs.

Functional interface to the drive

The functional interface to the drive is the speed setpoint interface.

The functional interface to a hydraulic valve is the analog flow rate setpoint and, if available, the analog force limiting or pressure limit value.

It is possible to connect both analog drives and digital drives as well as stepper drives.

Standardized protocols are used for setpoint specification for digital drives and for acknowledgments of encoder information.

Note

A SIMOTION axis can only execute functions that are supported by the connected drive. For further information, refer to the relevant product descriptions for the drives.

Automatic configuration as of V4.2 (Use symbolic assignment setting)

The only settings required during configuration are the preferred technology and functionality for the axis and encoder and the drive to be used by the axis. The PROFIdrive telegrams required for communication are determined and set by the system when the online connection is established.

Automatic adaptation uses add relevant drive data to the TO configuration without having to make these settings manually.

This functionality is available as of with SIMOTION V4.2 in conjunction with SINAMICS V2.6.2.

Programming commands / functions for the Axis technology object

The MCC and ST programming languages are available for programming axes.

Axis functions can also be addressed via the PLCopen blocks of the SIMOTION Function Library (up to V3.2) and the SCOUT command library (as of V4.0). This can also be accomplished in the LAD and FBD programming languages.

See the *SIMOTION MCC Motion Control Chart*, *SIMOTION ST Structured Text* and *PLCopen* Programming Manuals.

Axis fundamentals

3

3.1 Axis technologies

3.1.1 Overview of axis technologies

The Axis technology object can be configured as a speed-controlled axis, positioning axis, synchronous axis, or path axis. The various axis technologies differ according to the functionality provided on the axis.

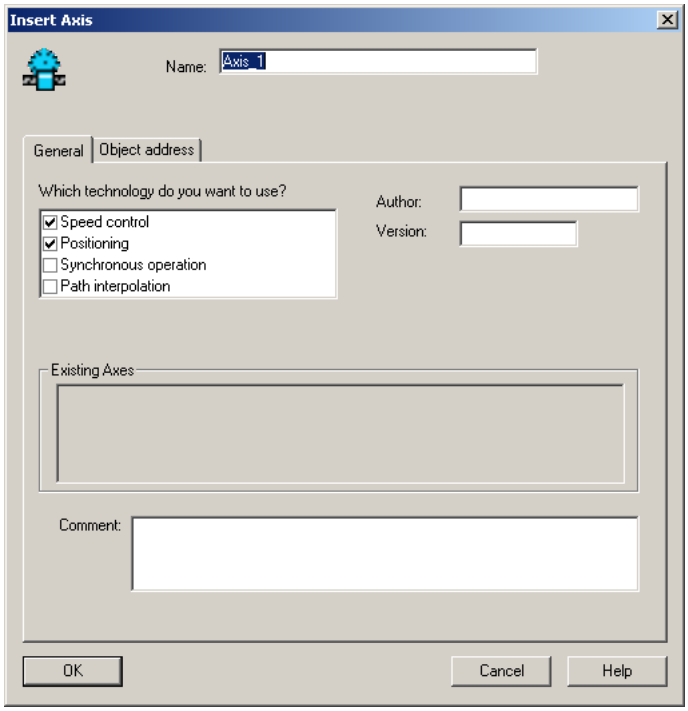


Figure 3-1 Axis technology setting in SIMOTION SCOUT

Possible axis functions in relation to the axis type

Table 3-1 Axis functions - Overview

Function	Speed-controlled axis	Positioning axis	Following axis	Path axis
Speed or velocity specification	X	X	X	X
Travel with torque limitation	X	X	X	X
Traversing according to motion vectors	X	X	X	X

Function	Speed-controlled axis	Positioning axis	Following axis	Path axis
Positioning	-	X	X	X
Travel to fixed stop	-	X	X	X
Homing	-	X	X	X
Advanced functions				
Measuring input	-	X	X	X
Output cam	-	X	X	X
Cam track	-	X	X	X
Gearing	-	-	X	X
Camming	-	-	X	X
Path interpolation	-	-	-	X

Operating modes

- Setpoint operation
  Setpoint mode is the "normal" mode of the axis, in which motion commands are accepted and executed.
- Follow-up mode
  The setpoint is corrected to the actual value in follow-up mode. The actual position value and actual speed value will be updated. The axis can then also be tracked if it is moved by external effects.
  Motion commands are not accepted/executed.
- Simulation mode
  The axis and the position controller are active in simulation mode. Simulation mode is used to test the programmed sequences in the control and the interaction between various axes, for example, on the basis of trace recordings without moving the axis.
  Simulation mode is only useful for real axes.
  Two variants of simulation mode are possible:
  - Program simulation mode
    The setpoints are calculated according to the programming, but are not sent to the position controller. The position controller setpoints remain set at the values they had prior to switching to simulation mode.
    See also Enabling/disabling program simulation (Page 341).
  - Axis simulation mode
    Unlike program simulation, it is possible to switch a real axis to axis simulation status even if the drive is not connected. The position controller remains active and the drive is simulated.
    See also Setting as a real axis without drive (axis simulation) (Page 60).

The operating modes are set via commands.

Speed-controlled axis

The speed-controlled axis type is used when the position of the axis is of no importance. Only the speed or velocity of an axis is specified, controlled, and monitored.

The speed is monitored if an encoder is configured on the axis; otherwise, the speed is not monitored.

The position of the speed-controlled axis is not monitored or controlled.

The speed of rotation is indicated as a unit of velocity, such as rpm.

Operating modes

- Speed controlled / speed specification (setpoint mode)
- Simulation
- Follow-up mode

Functions

- Speed or velocity specified via
  - Programmable value setting
  - Free velocity profile (time-related)
- Travel with torque limitation
- Force/pressure control, force/pressure limiting (for hydraulic functions only)

Positioning axis

The positioning axis type specifies, controls, and monitors the position of the axis.

The axis is moved to a programmed target position, which can be specified as a relative or an absolute value. The direction of motion or rotation can be specified for modulo axes.

For position-controlled positioning axes, position control is possible in the CPU or in the drive, provided the drive supports the dynamic servo control (DSC) method.

The positioning axis in SIMOTION does not have a velocity controller. The speed controller for an electric axis is in the drive.

The positioning axis can be interconnected with a path object for path-synchronous motion as a master axis with synchronous axis or as an axis.

Operating modes

As for speed-controlled axis, plus

- Position control

Functions

- Speed or velocity specified via
  - Programmable value setting
  - Free velocity profile (time- or position-related)
- Travel with torque limitation
- Position specified via
  - Programmable value setting
  - Free velocity profile (time-related)
- Travel to fixed stop

- Homing
- Force/pressure control, force/pressure limiting
- Synchronization of encoder values
- Positioning axis with path-synchronous motion (see path interpolation)

Synchronous axis

The synchronous axis type determines the axis setpoint from a master value according to conversion rules.

The synchronous object and following axis are separate objects, but together they form a synchronous axis.

The "Axis" and "Synchronous Operation" technology objects have a reciprocal effect on each other in terms of their operating states and the effectiveness of the commands used.

For example, errors in the "Axis" technology object will directly affect the synchronous operation functionality. When an axis stop response is triggered, the synchronous motion is stopped as well.

Operating modes

As for positioning axis

Functions

In addition to the positioning axis functions, other functions are available via the synchronous object:

- Gearing
- Camming
- Velocity gearing
- Dynamic synchronization/desynchronization

Other functions associated with the Synchronous Axis technology object can be found in the SIMOTION Technology Objects for Synchronous Operation, Cam Manual.

Path axis

The path axis type is used to travel along a path together with at least one additional path axis on the path object.

Via the path object, a path can be generated for at least two and up to three path axes.

The setpoints generated for the axis on the path object are limited on the axis to the maximum dynamic values of the axis.

The "Path Axis" and "Path Object" technology objects have a reciprocal effect on each other in terms of their operating states and the effectiveness of the commands used.

For example, errors in the "Path Axis" technology object will directly affect the generation of motion on the path object. When a stop response is triggered on the axis, the path motion is stopped as well.

Operating modes

As for positioning axis

Functions

In addition to the positioning axis functions, other functions are available via the path object:

- Linear path interpolation
- Circular path interpolation
- Polynomial path interpolation

The path axis contains the functionality of the synchronous axis.

Other functions associated with the Path Interpolation technology object can be found in the *SIMOTION Technology Object Path Object* Manual.

See also

Setting and canceling the axis enables (Page 313)

3.1.2 Axis-drive relationship

Technology objects represent the applicable real objects (e.g. an axis) in the controller.
The Axis technology object provides the user with a technological view of the drive and encoder (actuator and sensor), provides technological functions for them and performs the specific hardware connection.

The Axis technology object includes comprehensive functionality, e.g. communication with the drive, actual value conditioning, position control and positioning functions. It executes control and motion commands and indicates statuses and actual values.

The drive contains the speed and current control for the motor.

Technological limiting functions and values for the axis and encoder mechanics (e.g. leadscrew pitch and gear) are set at the axis.
The user can then only work with technological parameters.

The Axis technology object communicates with an actuator (drive or hydraulic valve) via a field bus system (PROFIBUS or PROFINET via PROFIdrive protocol) or via a direct setpoint interface (analog ±10 V or pulse/direction).

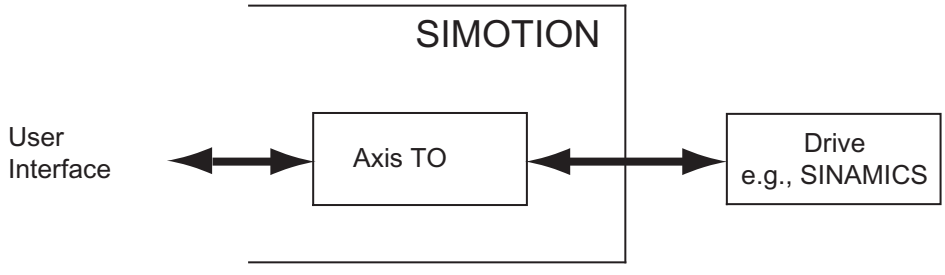


Figure 3-2 Axis-drive relationship

Functional interface to the drive

Various functional interfaces to the drive are available.

Analog drives, hydraulic valves, or stepper drives can be operated at the direct setpoint interface. See also Setting as a real axis with analog drive link (Page 38), Linking analog drives to SIMOTION (Page 226).

Setpoint specifications and feedback signals (incl. encoder information) for drives connected to a field bus are provided via standardized protocols (standard telegrams in accordance with the PROFIdrive profile). See also Coupling of digital drives (Page 56).

See also

Axis with stepper motor connection (Page 228)

Overview of axis settings / drive assignment (Page 283)

Encoder interface using the PROFIdrive message frame (Page 68)

3.2 Axis types

3.2.1 Overview of axis types

There are different axis types that differ according to their axis mechanism. The axis type also determines the units used to calculate axis variables such as position, velocity, etc.

- **Linear axes**
  Linear axes have coordinates that are specified in a unit of length. The position profile is continuous within the traversing range. Motion instructions are specified in units of length, for example in mm.
- **Rotary axes**
  Rotary axes have coordinates that are specified in a unit of rotation. The position profile is continuous within the traversing range. Motion instructions are specified in units of rotation, for example in degrees or rad.
- **Setting a linear axis or rotary axis as a modulo axis**
  Modulo axes have an unlimited traversing range and their position is mapped to a repeating modulo traversing range. The modulo range is defined by the start value and the modulo length.
  If the position value or axis position overshoots the modulo length, the position is reset to the modulo start value. If the position value undershoots the modulo start position, it is reset to the modulo start value plus the modulo length.
  Like rotary axes, linear axes can also be defined as modulo axes (modulo linear axis, modulo rotary axis).

Axis type setting

The following settings are available for the **axis type**:

- **Linear**
- **Rotary**

and

- **Electrical**
- **Hydraulic**
- **Virtual**

The **electrical**, **hydraulic**, or **virtual** setting affects the subsequent menu content.

See also

Actual value measurement / actual value system (Page 130)

3.2.2 Setting for the electric axis type

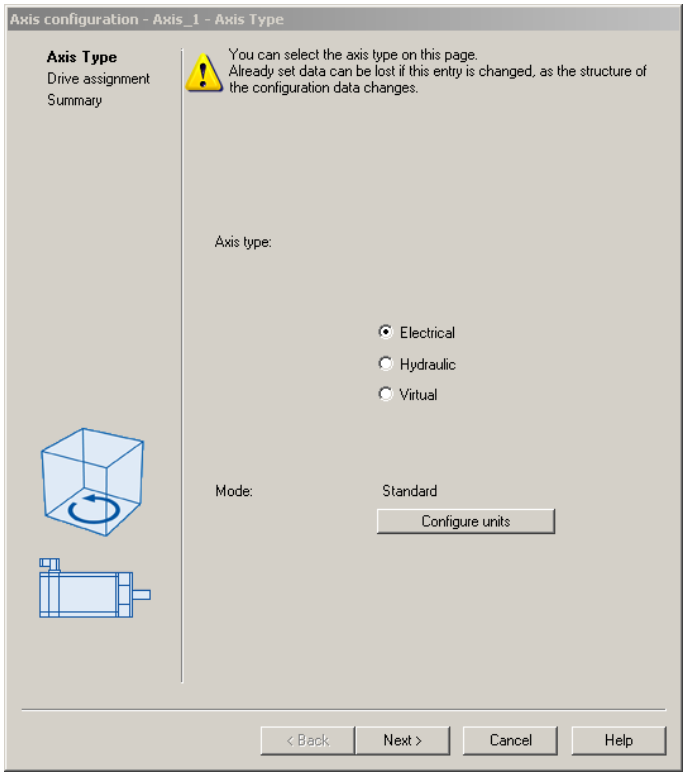


Figure 3-3 Axis type setting for an electric drive axis in SIMOTION SCOUT

Note

There is no force/pressure control for the electric drive axis. In this case, the mode is preset to Standard and cannot be changed.

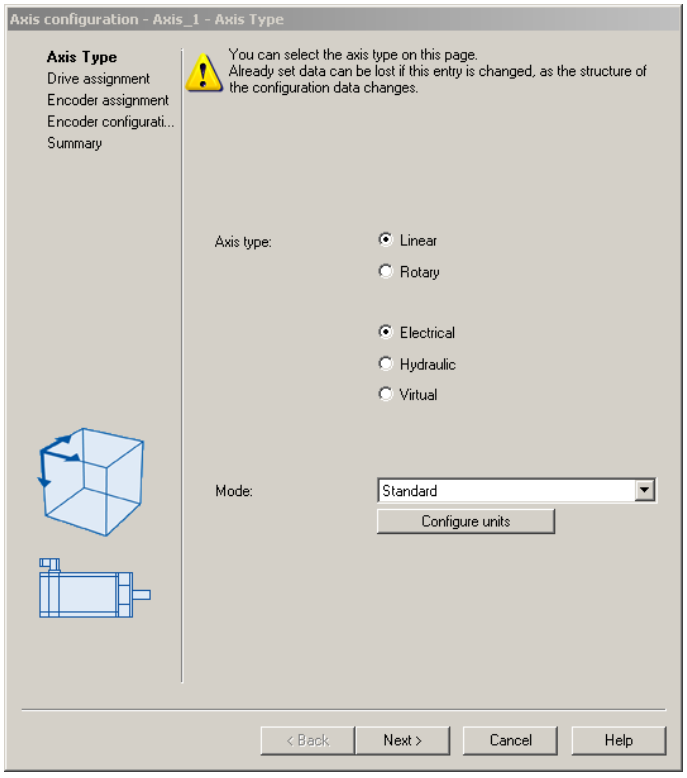


Figure 3-4 Axis type setting for an electric position or following axis in SIMOTION SCOUT

Table 3-2 Axis type options

Axis type	Description
Linear	Setting as linear axis
Rotary	Setting as rotary axis

Table 3-3 Mode options

Mode	Description
Standard	Position control
Standard + pressure	Position control and pressure control/pressure limitation
Standard + force	Position control and force control/force limitation

See also

Overview of force/pressure control (Page 187)

Overview of force/pressure limiting (Page 196)

Overview of axis types (Page 26)

TypeOfAxis configuration data (Page 31)

3.2.3 Setting for the hydraulic axis type

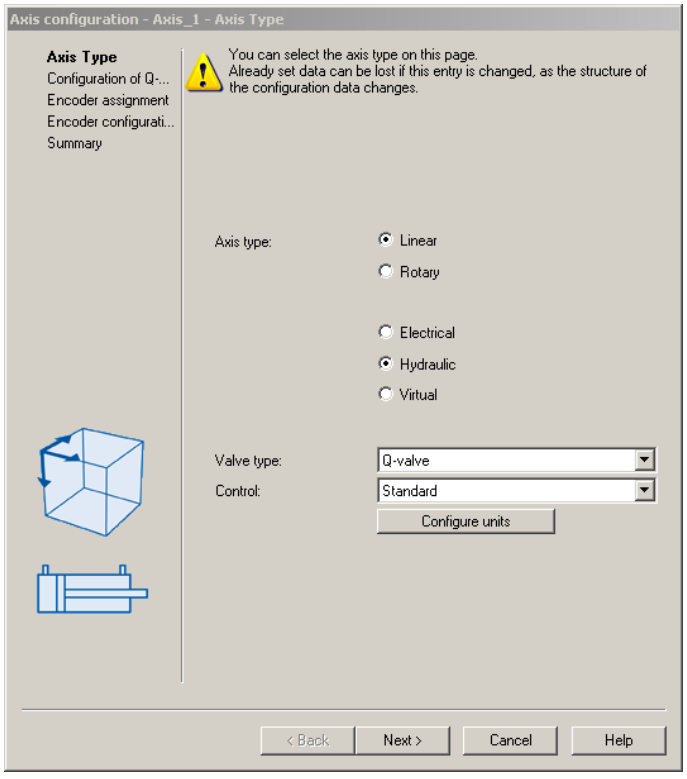


Figure 3-5 Axis type setting for a hydraulic position or following axis in SIMOTION SCOUT

Table 3-4 Valve type options

Valve type	Description
Q-valve	Axis with Q-valve (volumetric flow control)
P-valve [1]	Axis with P-valve (force/pressure control)
P+Q-valve	Axis with P+Q-valve

[1] Additional choice for drive axis

Table 3-5 Closed-loop control options

Closed-loop control	Description
Standard	Position control only
Standard + pressure	Position control and pressure control
Standard + force	Position control and force control

See also

Overview of force/pressure control (Page 187)
Overview of force/pressure limiting (Page 196)
Hydraulic functionality overview (Page 281)
TypeOfAxis configuration data (Page 31)
Setting as a real axis with Q valve only (Page 285)
Setting as a real axis with Q valve + P valve/F output (Page 287)
Setting an axis as a real axis with P valve only (V3.2 and higher) (Page 289)

3.2.4 Setting for the virtual axis type

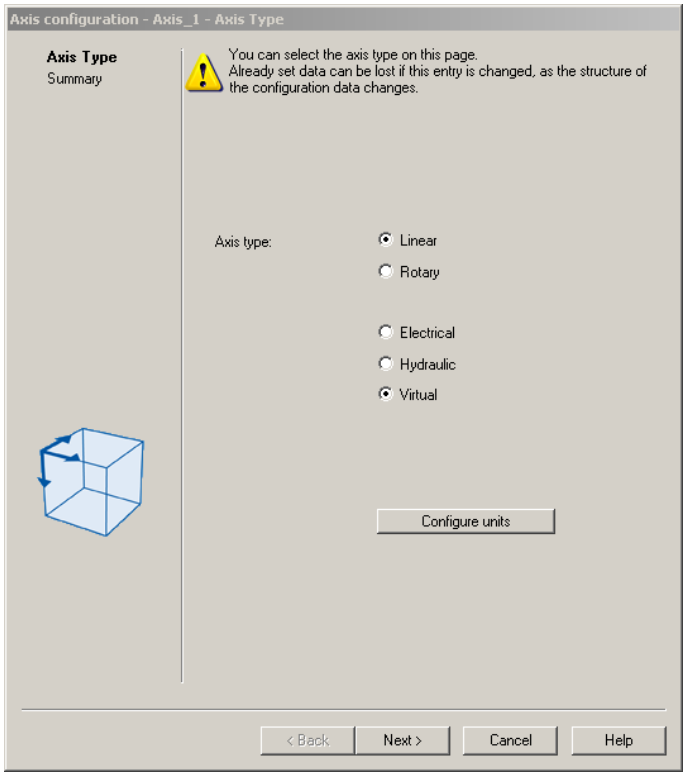


Figure 3-6 Axis type setting for a virtual position or following axis in SIMOTION SCOUT

3.2.5 TypeOfAxis configuration data

The axis type is entered according to the axis type parameter assignments under the **TypeOfAxis** configuration data in the axis wizard. The following table shows which axis wizard parameterization corresponds to which value under **TypeOfAxis**.

Assignment of **TypeOfAxis** based on the current axis wizard configuration:

	Axis type					Closed-loop control			Valve type d		
TypeOfAxis	Linear c	Rotary	Electrical	Hydraulic	Virtual	Standard	Standard+force	Standard+pressure	P valve	Q valve	P+Q-valve
VirtualAxis	x				x						
		x			x						
Real_Axis	x		x			x					
		x	x			x					
Real_Axis_With_Signal_Output	x										
		x	x								
Real_Axis_With_Force_Control [a]	x		x				x				
		x	x				x				
	x		x					x			
		x	x					x			
Real_QFAxis	x			x		x				x	
		x		x		x				x	
Real_QFAxis_With_Open_Loop_Force_Control	x			x			x				x
		x		x			x				x
	x			x				x			x
		x		x				x			x
Real_QFAxis_With_Closed_Loop_Force_Control	x			x			x			x	
		x		x			x			x	
	x			x				x		x	
		x		x				x		x	
Real_QFAxis_With_Open_Loop_Force_Control_only[b]		x		x			x		x		
		x		x				x	x		

a not possible for speed-controlled axis
b speed-controlled axis only
c not possible for speed-controlled axis
d hydraulic functionality only

3.3 Units and accuracies

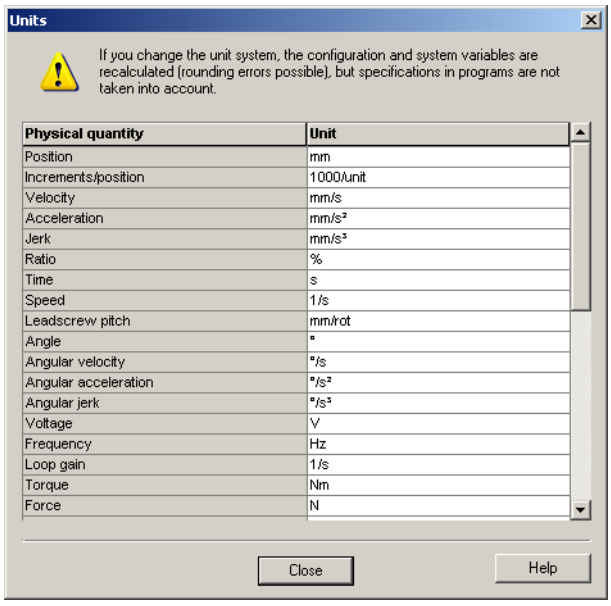


Figure 3-7 Setting the units and resolution in SIMOTION SCOUT

With the SIMOTION technology objects, such as the Axis technology object, physical variables such as position, velocity, acceleration, time, force, and torque are represented in the SI or US system of units (metric or imperial). The unit can be defined for all variables on the relevant technology object during configuration, for example, for:

- Unit of length
  - mm
  - m
  - km
  - inch
- Unit of force
  - N
  - kN
  - tfm
    ton force, metric unit
  - tfs
    ton force (short) or ton force (US), US unit

The defined units represent system variables and configuration data.

Changing the unit settings converts the current values of the system variables and configuration data from the engineering system to the new units.

Note

Values in the command parameters are interpreted in the defined unit on the technology object.

Numerical values in user programs (e.g. in the motion commands) are **not** converted to the new units when the unit settings are changed!

In compound variables, such as controller gains, the units may differ from the units of the individual variables, for example, force in [kN] and force/time in [N]/[sec].

The **internal computational accuracy and internal representation of the position** can also be defined in the unit configuration. It defines, among other things, the accuracy with which specifications in system variables, configuration data, and command parameters are accepted, processed, and displayed by the system.

Note

The positioning accuracy equates to one computational increment (increments/position), i.e. positions can only be defined at integer increment values. Intermediate values arising during interpolation and as a result of closed-loop control may also exist between the integer increments.

The display resolution and the resolution when entering parameter values are independent of the increments.

This definition refers to a specific basic unit of the position, depending on the axis type.

- Linear axis: Increments/mm
- Rotary axis / speed-controlled axis: Increments/degree

The control performs internal calculations in increments with reference to these basic units. Prior to processing, values are converted to the internal representation.

Example 1

The following configuration has been defined:

- Linear axis
- Position unit: m
- Increments/position: 1000/mm

Calculation of setpoint accuracy during positioning:

Position: 1,000/mm corresponds to 0.001 mm = 10^{-6} m

Example 2

The following configuration has been defined:

- Linear axis
- Position unit: mm
- Increments/position: 1000/mm

- Leadscrew pitch: 10.3334 mm
- Modulo length: 20.3335 mm

Determination of the effective leadscrew pitch and modulo length:

Position accuracy: 0.001 mm

- Effective leadscrew pitch on the TO: 10.333 mm
- Effective modulo length: 20.333 mm

If the leadscrew pitch and the modulo length have to be mapped precisely, the positioning accuracy on the TO has to be increased (incr/mm).

Fetch the units setting

The units setting can be fetched using the associated specially defined system variables. The listing of the variables is shown below:

- <speedaxis>.units : StructUnitsSpeedAxis
- <posaxis>.unitsPosAxis : StructUnitsPosAxis
- <externalencoder>.units : StructUnitsExternalEncoder
- <controller>.units : StructUnitsController
- <sensor>.units : StructUnitsSensor
- <synchronousoperations>.units : StructUnitsSynchronousOperations
- <temperaturechannel>.units : StructUnitsTemperatureChannel
- <pathobject>.units : StructUnitsPathObject
- <additionobject>.units : StructUnitsAdditionObject
- <fixedgear>.units : StructUnitsFixedGear
- <measuringinputs>.units : StructUnitsMeasuringInputs
- <outputcam>.units : StructUnitsOutputCam
- <camtrack>.units : StructUnitsCamTrack

Note

Fetching is possible only when the units have valid values, which is the case only in online mode.

For further information, see the *TP System Variables* List Manual.

3.4 Axis settings / drive assignment

3.4.1 Overview of how to create an axis

New axes are created using an axis wizard in which the axis parameters (configuration data and system variables) are queried or are configured automatically. You can specify other selected parameters via the axis parameter assignment dialog boxes (under Axis Object in the project navigator).

3.4.2 Real and virtual axes

Setting options

In SIMOTION, you can define the axes as:

- **Real axis**
  This axis features motion control, as well as drive and encoder interfaces.
- **Real axis with force/pressure control**
  This axis features motion control, drive and encoder interfaces, and an interface for force/pressure measurement and closed-loop control.
  For force/pressure control, the input for the force/pressure measurement must also be configured.
- **Virtual axis**
  This axis features reference variable generation, but does not have closed-loop control or a drive or encoder interface. The setpoints and actual values are always identical. A virtual axis is generally used as an auxiliary axis, in order to generate the setpoints for several real axes as the master axis, for example.
  The controller-specific enables are set by default.

See also Overview of axis types (Page 26).

Axes with no drive assignment (as of V4.2)

A drive is assigned to the axis in the drive assignment dialog during axis configuration. It is also possible, however, to leave the drive assignment open. The real axis can be programmed with all functions and the system remains operational after axes with no assigned drive have been downloaded. No consistency error is output.

One advantage of axes with no assigned drive is that you can set up and test the configuration in advance while there is still no drive or if you do not have all the necessary drive know-how. The axis can be connected to a drive later.

Runtime model and processing

The integrated motion control functionality uses a deterministic real-time runtime model for motion control purposes. This includes, in particular:

- Isochronous system levels for
  - Bus task
    PROFIBUS/PROFINET communication for linking digital drives, data exchange with the I/O
  - Servo_fast (optional)
    Position control and monitoring of axes, drive communication, and I/O processing synchronized with a fast PROFINET cycle clock
  - Servo
    Position control and monitoring of axes, drive communication, and I/O processing
  - IPO_fast
    Motion control for the axes is calculated in the fast PROFINET bus system
  - IPO
    Interpolator = reference variable calculation/motion profile calculation of the axes
    The interpolator cycle clock is set during configuration of the execution system of the device. There are two interpolator levels in the system, IPO and IPO2.
- Adjustable transformation ratios between bus task, servo, and IPO for appropriate load distribution and optimum system utilization

The processing cycle clock (axis-specific interpolator cycle clock) of the axis technology object can be set to IPO or IPO2. This makes it possible to place an interpolator for axes that do not require a high time resolution to calculate reference variables in a cyclical system task with a longer cycle time, thereby requiring less processing power.

For fast responses in motion control, in exceptional cases, the processing cycle clock can even be set to servo or servo_fast, see also Motion execution/interpolator (Page 208).

The processing cycle clock is set in the axis dialog configuration.

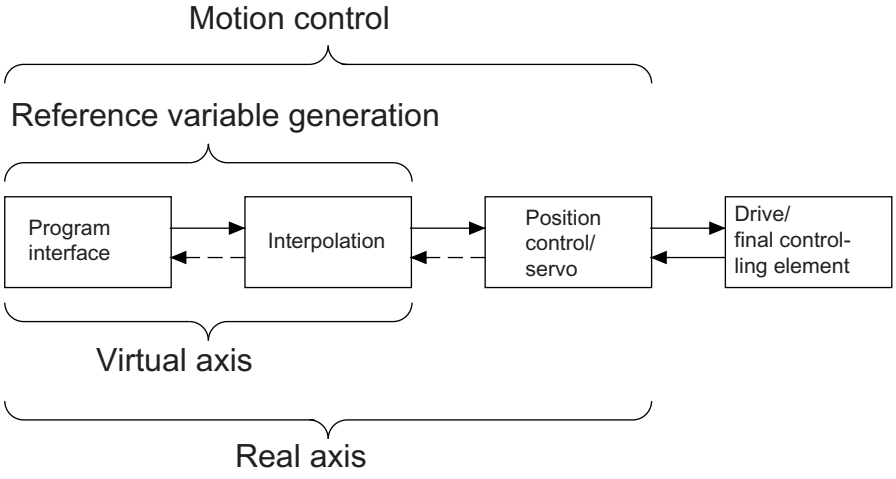


Figure 3-8 Difference between real and virtual axis (position axis example)

For real axes, the interface to the drives/final controlling elements is set.

For the hydraulic functionality, the analog output is set for manipulated variable value Q (flow rate) and, if applicable, for manipulated variable value F (force/pressure limiting). This makes it possible to connect valves with an analog manipulated variable.

If DSC (Dynamic Servo Control) is set for digital drives with a PROFIdrive interface, a position controller is executed in the drive (e.g. in the closed-loop speed control cycle clock).

If **symbolic assignment** is used (default setting as of V4.2), DSC is selected automatically.

3.4.3 Setting as a real axis with analog drive link

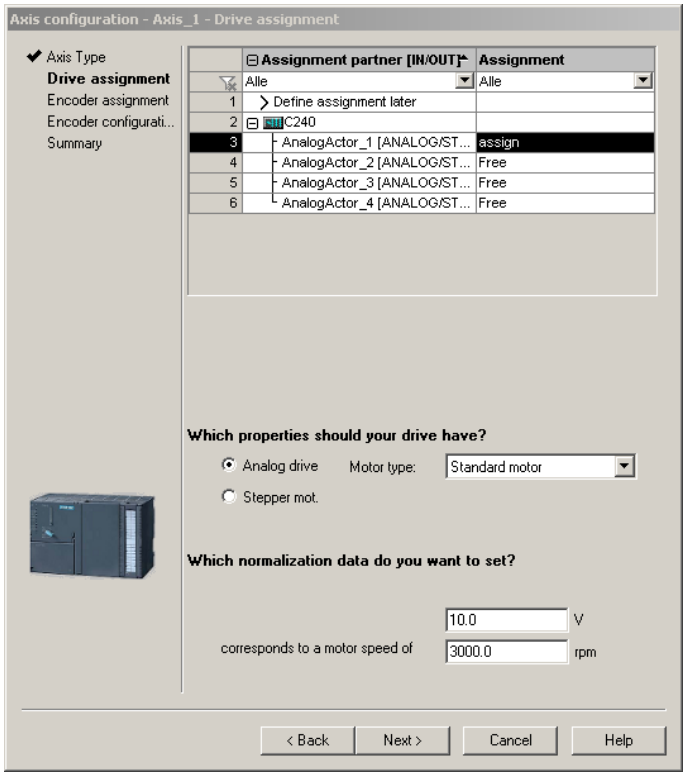


Figure 3-9 Setting for analog drive link to C2xx

For interconnection of the axis-specific inputs/outputs (encoder, analog output, enables), see the C2xx operating instructions.

In addition to the option of operating analog axes on the onboard inputs of the C2xx, the PROFIBUS ADI4 and IM174 modules are available for use in all platforms as interfaces for analog drive links. From the SIMOTION perspective, these modules behave like digital drive links; see Coupling of digital drives (Page 56).

The enable signal to the drive is activated with **_enableAxis()** (**enableMode**=ALL); the enable is displayed by the system variables **actormonitoring.driveState** = ACTIVE and **actormonitoring.power** = ACTIVE.

See also

Coupling of digital drives (Page 56)

3.4.4 Setting as a real axis with digital drive coupling

3.4.4.1 Symbolic assignment and adaptation (as of V4.2)

As of V4.2, SIMOTION supports the automatic adaptation of drive data as well as the configuration of technology objects (TOs) with symbolic assignment.
This makes it easier to configure the technical relationships between the SIMOTION technology objects (TOs) and the SINAMICS drive objects (DOs).

Adaptation and symbolic assignment enable:

- The required axis telegrams, as well as the addresses used, to be automatically specified by the engineering system
- Telegrams to be expanded and interconnections in the drive to be set up automatically dependent upon the selected TO technology (e.g. SINAMICS Safety Integrated)
- Axis configuration and drive configuration to be completed as two separate procedures
- Drive and encoder data, as well as reference variables, maximum variables, torque limits, and the selectivity associated with torque reduction on the SINAMICS S120 to be adapted automatically on system ramp-up

For more information, please refer to the sections of the **Motion Control Basic Functions** manual which deal with *symbolic assignment (as of V4.2)*.

Requirement

The following components support adaptation and symbolic assignment:

- SIMOTION C, P and D devices, with firmware as of SIMOTION V4.2
- SINAMICS S110 drives, firmware V4.x with SINAMICS as of V4.3
- SINAMICS S120 drives, firmware V2.x with SINAMICS as of V2.6.2
- SINAMICS S120 drives, firmware V4.x with SINAMICS as of V4.3

The SINAMICS versions indicated relate to:

- SINAMICS control units connected via PROFIBUS or PROFINET
- SINAMICS Integrated on a SIMOTION D
- Controller extensions CX32/CX32-2

Axis configuration: Drive assignment

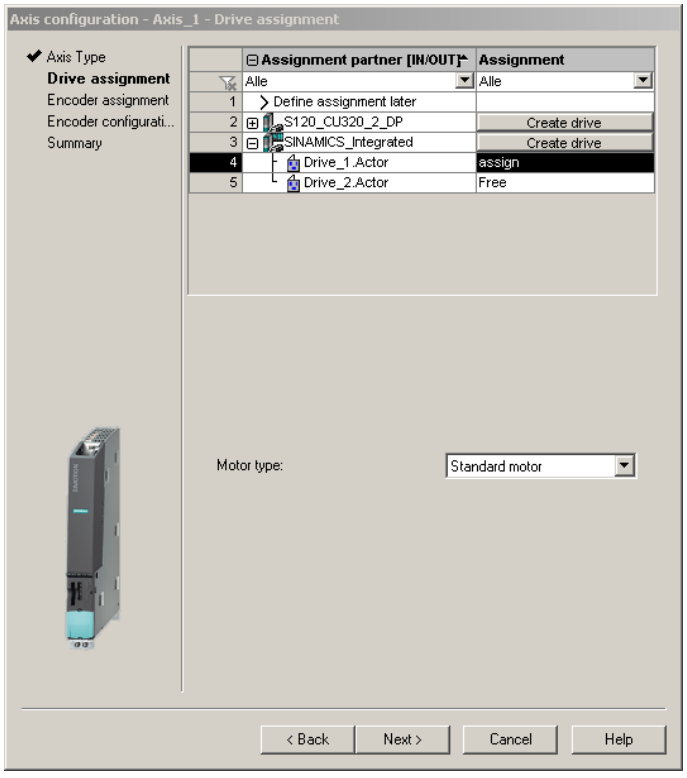


Figure 3-10 Drive assignment dialog

The following settings can be made in the **Drive assignment** dialog for the axis configuration:

- Assign drive
  All of the drives configured in SCOUT or HW Config are displayed as assignment partners. To interconnect a drive, click a drive object in the table in the **Drive assignment** dialog.
  The hardware addresses and telegrams are determined automatically by the system.
- Assign later
  Enables the axes to be configured independently on the SIMOTION side and the drives to be configured independently on the SINAMICS side.
  This means, for example, that:
  - The drives can be configured by a drive expert and tested and optimized using the drive control panel
  - The PLC and Motion Control functions can be configured by a programmer using technology objects (e.g. Axis technology object)
  - The technology objects can then be symbolically assigned to the drive objects at a later point in time via an assignment dialog
- Create drive
  From the Assign dialog, a new drive can be created on an existing drive unit directly (e.g. S120 CU320-2 or SINAMICS Integrated) and the axis assigned. As a result, it is not necessary to configure a drive before creating an axis.

The TO axis, TO external encoder, TO output cam, TO cam track, and TO measuring input support symbolic assignment. Furthermore, the onboard I/O of a SIMOTION D, a SINAMICS S120 control unit, and even selected terminal modules can be interconnected symbolically (e.g. for hardware-limit-switch inputs on the technology object).

The list of addresses in the **All addresses** view provides an overview of the assignments of all the Axis technology object interfaces. It is also possible to change the assignments from this view via the Assign dialog (**...** button).

Note

The previous methods of drive and axis configuration continue to be available. Symbolic assignment must be deactivated in such cases.

Symbolic assignment is used by default for newly created projects.

If projects < V4.2 are converted upwards, symbolic assignment is deactivated by default and must be activated if required.

Symbolic assignment can be activated/deactivated in SIMOTION SCOUT via the **Project > Use symbolic assignment** menu.

Adaptation

In addition to the benefits offered by symbolic assignment, configuration is also made easier as of SIMOTION V4.2 with the addition of automatic adaptation of SINAMICS S120 data.

Drive and encoder data as well as reference variables, maximum values, torque limits and granularity for SINAMICS S120 torque reduction, are taken over automatically for the configuration data of the SIMOTION Axis and External encoder technology objects. These data no longer have to be entered in SIMOTION.

Events and commands after which an adaptation is carried out:

- Switching the controller on (ramp-up)
- Download
- STOP-RUN transition, if no adaptation has been made yet
- Restarting a TO
- **_activateTO()** command
- **_enableAxisInterface()** command if the status "not adapted" is present at the interface with non-exclusive drive assignment
- **_adaptAxisConfigData()** and **_adaptExternalEncoderConfigData()** user commands
- Drive data set changeover (and encoder changeover does not cause adaptation)

The adaptation of relevant drive data is activated automatically as of SIMOTION V4.2 in conjunction with SINAMICS S120 as of V2.6.2. If projects < V4.2 are converted upwards, automatic adaptation is deactivated by default. You can activate automatic adaptation if required via the expert list in the configuration data element **TypeOfAxis.DriveControlConfig.dataAdaption**. You can activate automatic adaptation of encoder data in the configuration data element **TypeOfAxis.NumberOfEncoders.Encoder_n.dataAdaption**.

The configured data initially contains default values for the values to be adapted. Adaptation may result in a difference between the configured values and the adapted values. In the case of a *Copy current data to RAM*, the TO would, therefore, be displayed as inconsistent in the project navigator.

When executing the functions *Copy current data to RAM* and *Copy RAM to ROM*, the difference will be detected and an upload of the adapted data into the SCOUT project will be executed. However, this can be deselected in the dialog box.

3.4.4.2 Adapted data (SINAMICS S120 as of V2.6.2)

The adapted drive and encoder data is listed in the following.

Adapted drive data (motor and actuator)

Table 3-6 Reference and maximum velocity

Motor type	Parameter	Configuration data element
Round-frame motor	p2000	**driveData.nominalSpeed**
Linear motor	p2000	**linearMotorDriveData.nominalSpeed**
Round-frame motor	p1082	**driveData.maxSpeed**
Linear motor	p1082	**linearMotorDriveData.maxSpeed**

Table 3-7 Resolution of torque reduction

Motor type	Parameter	Configuration data element
Round-frame motor	p1544	**driveData.torqueReductionGranularity**
	p1520	**driveData.maxTorque** (and p1520 = |p1521|)
Linear motor	p1544	**linearMotorDriveData.forceReductionGranularity**
	p1520	**linearMotorDriveData.maxForce** (and p1520 = |p1521|)

Table 3-8 Resolution of the reference and maximum values for B+, B-, M_{add} (technology data block)

Motor type	Parameter	Configuration data element
Round-frame motor	p2003	**driveData.nominalTorque**
Linear motor	p2003	**linearMotorDriveData.nominalForce**

Table 3-9 Adaptation of the parameters for specific DO types and availability in the drive

Parameter	Semantics	DO Servo	DO Vector	DO TM41	DO DCM	DO Encoder
p2000	Reference speed	x	x	x	x	x
p2003	Reference torque	x	x	x	x	-
p1082	Maximum speed	x	x	-	-	-

Parameter	Semantics	DO Servo	DO Vector	DO TM41	DO DCM	DO Encoder
p1520	Maximum torque	x	x	-	-	-
p1544	Torque reduction granularity (only 100 telegrams)	x	-	-	-	-
p0979	Encoder information	x	x	x	x (only telegram 1)	x

The following conditions must be observed:

- When assigning an axis on a SINAMICS drive, the following is set implicitly:
  - **Adaptation**
    driveControlConfig.dataAdaption=YES
  - **speedReference** (speed, velocity) to nominalspeed, ~velocity
    driveDate.speedreference=NOMINAL_VALUE
    linearMotorDriveData.velocityReference=NOMINAL_VALUE
  - **torqueReference**, **forceReference** (torque, force) to nominaltorque, ~force
    driveDate.torqueReference=NOMINAL_VALUE
    linearMotorDriveData.forceReference=NOMINAL_VALUE
- Adaptation is not set generally for an interconnection to a SINAMICS DO, but specifically depending on the DO type.
  This means that the properties of the DO type are known when setting the adaptation.
- Parameter p1544:**driveData.torqueReductionGranularity** or **linearMotorDriveData.forceReductionGranularity** is only adapted when a 1xx telegram is set.
  Otherwise no adaptation, since it is not relevant or with DO vector, DO TM 41 and DO DCM also not available.
- If parameter p1082 (maximum speed) is not available on a DO, then the maximum speed is set equal to the reference speed in SIMOTION (**driveData.maxSpeed** = driveData.nominalSpeed or **linearMotorDriveData.maxSpeed** = linearMotorDriveData.nominalSpeed).
  This is the case, for example, for DO TM41 or TO DCM.
- If parameter p1520 (maximum torque) is not available on a DO (e.g. DO DCM), then the maximum torque/force is set equal to the reference torque/force on the TO (**driveData.maxTorque** = driveData.nominalTorque or **linearMotorDriveData.maxForce** = linearMotorDriveData.nominalForce). The torque values are not adapted when interconnecting to DO TM41.

Adapted encoder data

The encoder type set in the axis configuration, incremental or absolute, is checked during the adaptation.

The measuring system setting, linear or rotary, is adapted.

Table 3-10 Incremental rotary encoder

Encoder characteristic data	Parameter	Configuration data element
Increments/revolution	p0979[2]	**incEncoder.incResolution**
Fine resolution	p0979[3]	**incEncoder.incResolutionMultiplierCyclic**

Table 3-11 Incremental linear encoder

Encoder characteristic data	Parameter	Configuration data element
Grid spacing	p0979[2]	**resolution.distance**
Fine resolution	p0979[3]	**resolution.multiplierCyclic**

Table 3-12 Absolute rotary encoder

Encoder characteristic data	Parameter	Configuration data element
Increments/revolution	p0979[2]	**absEncoder.absResolution**
Fine resolution	p0979[3]	**absEncoder.absResolutionMultiplierCyclic**
Fine resolution for absolute value in Gn_XIST2	p0979[4]	**absEncoder.absResolutionMultiplierAbsolute**
Data width of absolute value without fine resolution	p0979[2] and p0979[5] *)	**absEncoder.absDataLength**

*) The data width of the absolute value (without fine resolution) is a result of the sum of the bits for representing the number of encoder pulses and the bits for representing the maximum number of revolutions that can be registered by the encoder according to the type plate.

Example:
4,096 pulses/rev (= 12 bit) and maximum of 4096 revolutions that can be registered results in a 12 + 12 = 24-bit data width of the absolute value.

Table 3-13 Absolute linear encoder

Encoder characteristic data	Parameter	Configuration data element
Grid spacing	p0979[2]	**resolution.distance** (**resolution.multiplierCyclic** has no significance for a linear absolute encoder)
Fine resolution	p0979[3]	**absEncoder.absResolutionMultiplierCyclic**

Encoder characteristic data	Parameter	Configuration data element
Fine resolution for absolute value in Gn_XIST2	p0979[4]	**absEncoder.absResolutionMultiplierAbsolute**
Data width of absolute value without fine resolution	p0979[2] and p0979[5] *)	**absEncoder.absDataLength**

*) The data width of the absolute value (without fine resolution) is a result of the sum of the bits for representing the number of encoder pulses and the bits for representing the maximum number of revolutions that can be registered by the encoder according to the type plate.

Example:
4,096 pulses/rev (= 12 bit) and maximum of 4096 revolutions that can be registered results in a 12 + 12 = 24-bit data width of the absolute value.

3.4.4.3 Communication and control word / status word

Communication with the drive via PROFIdrive

The communication with digital drives using PROFIBUS/PROFINET is made in accordance with the PROFIdrive V4 specification and the application classes 1 to 4 (class 4 both with and without DSC).

The ADI4 and IM 174 modules can be used to connect axes with analog interface to the system. The system uses the standard 3 telegram in accordance with the PROFIdrive profiles for the ADI4 and IM174 modules. Consequently, in this case, the axis TO sees a real axis with digital drive coupling.

For axes with digital drive coupling using the PROFIdrive telegram, the maximum velocity is twice the reference velocity.

Control word

Drive enables are set in conformity with PROFIdrive Profile V3.1 via control word STW1.

Control bits in control word STW1

The Axis technology object takes over control of the control word with the **_enableAxis()** and **_disableAxis()** commands.

The system uses **_enableAxis()** or **_disableAxis()** to set the control bits in STW1 bit 0 - STW1 bit 6. Also refer to the following table for the semantics of the STW1 bit 0 – STW1 bit 6 control word bits.

_enableAxis() (**enableMode**=DRIVE) is used to set Bit4 - Bit6 in STW1, while **_enableAxis()** (**enableMode**:=POWER) is used to set Bit0 - Bit3 in STW1.

enableMode=ALL is used to set the DRIVE and POWER bits.

Table 3-14 Meaning of the control word bits STW1 bit 0 – STW1 bit 6

STW1	Bit = 0 meaning	Bit = 1 meaning	Notes
Bit 0	OFF, Brake on ramp-function gen. (OFF 1)	ON	Brake along the ramp generator deceleration ramp followed by pulse suppression
Bit 1	Coast stop (OFF2)	No coast stop (no OFF2)	Pulse suppression
Bit 2	Quick stop (OFF3)	No quick stop (no OFF3)	Brake along the OFF3 deceleration ramp followed by pulse suppression
Bit 3	Disable operation	Enable operation	-
Bit 4	Disable ramp generator	Enable ramp generator	-
Bit 5	Freeze ramp generator	Unfreeze ramp generator	-
Bit 6	Disable setpoint	Enable setpoint	-

You can find a detailed description of the bits in status and control words in the SINAMICS List Manuals.

The control word from the drive protocol is displayed in the system variable **driveData.STW** (STW1 corresponds to STW[0]).

The states are also displayed in the SCOUT at **Drives - Drive_x - Diagnosis - Control/status words**.

Stop modes with PROFIdrive – Profile Drive Technology

- STW1 bit 0 = 0 (OFF1): **Ramp stop**
  - The drive travels to zero velocity with the deceleration that can be defined on the drive.
  - The stopping process can be interrupted and the drive reactivated.
  - After stopping, a pulse suppression is carried out and the status changes to ready to start.
- STW1 bit 1 = 0 (OFF2): **Coast stop**
  - The drive immediately goes to pulse suppression and the status changes to switch-on disable.
    The drive coasts to a standstill.
- STW1 bit 2 = 0 (OFF3): **Quick stop**
  - The drive travels to zero velocity with the torque limit that can be defined on the drive.
  - The stopping process cannot be interrupted.
  - After stopping, a pulse suppression is carried out and the status changes to switch-on disable.

Status word

Status word ZSW1 contains status information of the drive in accordance with PROFIdrive profile V3.1.

Table 3-15 Meaning of the status word bits ZSW1 bit0 – ZSW1 bit 15

ZSW1	Bit = 0 meaning	Bit = 1 meaning	Notes / parameters
Bit 0	Not ready to power up	Ready to start Power supply on, electronics initialized, line contactor released if applicable, pulses inhibited.	BO: r0899.0
Bit 1	Not ready for operation Reason: No ON command present	Ready for operation Voltage at Line Module, i.e. line contactor closed (if used), field is being built up.	BO: r0899.1
Bit 2	Operation inhibited	Operation enabled Enable electronics and pulses, then ramp up to active setpoint	BO: r0899.2
Bit 3	No fault active No active fault in the fault buffer.	Fault active The drive is faulty and, therefore, out of service. The drive switches to "switching on inhibited" once the fault has been acknowledged and the cause has been remedied. The active faults are stored in the fault buffer.	BO: r2139.3
Bit 4	Coasting down active (OFF2) An OFF2 command is active.	No OFF2 active	BO: r0899.4
Bit 5	Quick stop active (OFF3) An OFF3 command is active.	No OFF3 active	BO: r0899.5
Bit 6	No "Power-on inhibit" Switching on is possible	Power-on inhibit A restart is only possible by means of OFF1 and then ON.	BO: r0899.6
Bit 7	No alarm active No active alarm in the alarm buffer.	Alarm active The drive is operational again. No acknowledgment necessary. The active alarms are stored in the alarm buffer.	BO: r2139.7
Bit 8	Setpoint/actual speed monitoring not within tolerance band	Setpoint/actual speed monitoring within tolerance band Actual value within a tolerance band; dynamic overshoot or undershoot for t < t_{max} permissible, e.g. $n = n_{set}\pm$ $f = f_{set}\pm$, etc., t_{max} can be parameterized	BO: r2197.7

ZSW1	Bit = 0 meaning	Bit = 1 meaning	Notes / parameters
Bit 9	Local operation Control only possible on device	Control requested The PLC is requested to assume control. Condition for applications with isochronous mode: Drive synchronized with PLC system.	BO: r0899.9
Bit 10	f or n comparison value not reached.	f or n comparison value reached or exceeded.	BO: r2199.1 **Note:** The message is parameterized as follows: p2141 Threshold value p2142 Hysteresis
Bit 11	I, M or P limit reached or exceeded	I, M or P limit not reached	BO: r1407.7
Bit 12	Holding brake closed	Holding brake opened	BO: r0899.12
Bit 13	Motor overtemperature alarm active	Motor overtemperature alarm not active	BO: r2135.14
Bit 14	Actual speed < 0	Actual speed > = 0	BO: r2197.3
Bit 15	Alarm, converter thermal overload The overtemperature alarm for the converter is active.	No alarm active	BO: r2135.15

The status word from the drive protocol is displayed in the system variable **driveData.ZSW** (ZSW1 corresponds to ZSW[0]).

See also

Drive communication based on DPV1 services (Page 222)

Alarm reactions (Page 353)

3.4.4.4 Specific removal of drive enables in the event of a fault

In the event of a fault (local alarm reaction RELEASE_DISABLE), all drive enables are removed by default.
A setting can also be made to remove drive enables individually.
To access the **Functions - Response to alarm** dialog, select the *Axis - Configuration* menu and click the *Change ...* button under Functions.

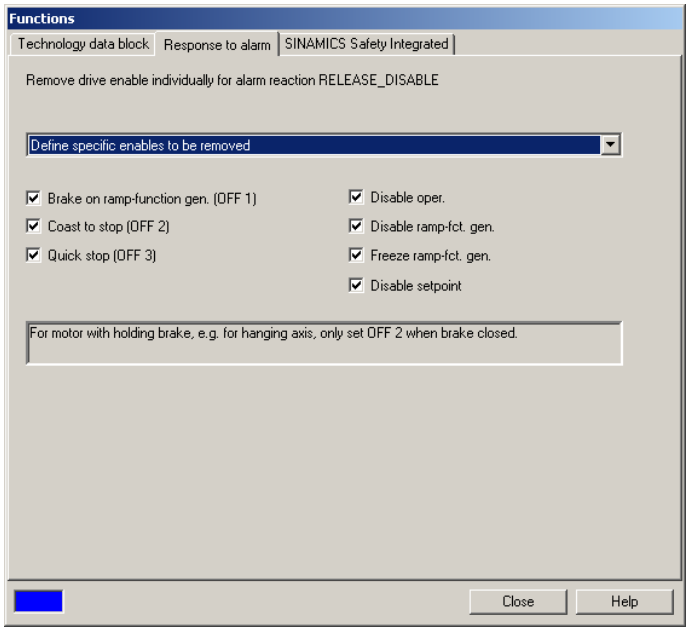


Figure 3-11 Settings for the drive in SCOUT

The bits selected in the screen form will be reset in the control word for the RELEASE_DISABLE local alarm response.

For the local RELEASE_DISABLE alarm response, all enables to the drive will be removed for system setting. Before the enables in STW1 bit 0 - STW1 bit3 are removed, status word ZSW1 bit 10 = 0 (n = 0 in the drive) will be awaited in accordance with the general state diagram in SIMOTION (see figure below).
This system setting can be changed in the **TypeOfAxis.driveControlConfig.releaseDisableMode** configuration data as well as in the dialog box.

Note

At a minimum, the canceling of one enable from bit 0 to bit 6 must be set.

If the value 0 is entered in **releaseDisableMode** in bit 0 to bit 6, all enables are canceled.

See also

Alarm reactions (Page 353)

3.4.4.5 Brake control

The brake control is required for operating "hanging axes", for example. The integral brake control of the drives is used for this purpose.
The brake control must be taken into account when enabling the drive and revoking the enable signal.

System support (from SIMOTION V4.4)

The configuration data element **TypeOfAxis.DriveControlConfig.disableModeSpecification** can be used to switch off the drive via AUS3 by default if OFF1_AND_OFF3_MODE is set. OFF2 is therefore only set for the drive when e.g. the brake is closed.
This setting is also the standard setting for newly created axes.

When the enable signals are revoked (via **_disableAxis()** or the RELEASE_DISABLE alarm response), the system behaves as follows if OFF1_AND_OFF3_MODE is set:

1. OFF1 and OFF3 are canceled first when the axis is switched off.
   - The OFF3 ramp is therefore active.
   - STW1 bit0 is already 0 (power-on inhibit canceled)
2. After the drive has transitioned to S1 or S2 as per the PROFIdrive General State Diagram, the remaining enable signals for switching off are canceled.

This functionality is independent of the brake control configuration in the drive and is active even when the enable signals are canceled via the axis control panel.

Directly and exclusively setting OFF2 is always effective.

Extended brake control with SINAMICS

In the case of extended brake control with SINAMICS, the ready signal "Operation enabled" is sent to SIMOTION via status word 1 (system variable **driveData.ZSW[0]** Bit 2) after the brake opening time has expired (p1216). This means the step enabling condition following **_enableAxis()** is not met until the brake opening time has expired.

Please note that when using the extended brake control, the feedback signal "Brake released" is not sent to the ZSW[0] Bit2 (operation enabled), and instead only the brake opening time is taken into account.

When implementing a brake control in the drive, the enable signals must be specifically revoked in the case of **_disableAxis()** and the RELEASE_DISABLE local alarm response (see above), e.g.:

- First, revoke OFF3 (STW1 bit 2)
- When the brake is closed, disconnect the power (revoke STW1 bit 1)

For information on defining the reaction to technological alarms, refer also to Chapter Adjustable response to RELEASE_DISABLE (Page 357).

3.4.4.6 Specify control bits via the application (without TO check) [expert]

As of V3.2, it is also possible to specify Bit0 - Bit6 in STW1 specifically via **_enableAxis()** (enableMode:=BY_STW_BIT) and **_disableAxis()** (disableMode:=BY_STW_BIT). Each bit to be set/canceled is then specified in the STWBitSet parameter. For this setting, the axis TO checks also for single bit assignment whether the specifications of the PROFIdrive state machine are observed.

If the axis is switched on from the S1 state, in accordance with PROFIdrive, bit 0 of the initial state 0 for STW1 is required (pulse suppression and ready-to-start status). If STW1 bit 0 in state S1 is not set to 0 (as result of enables being canceled using single-bit assignment or the cancelation of some enables with the RELEASE_DISABLE alarm reaction), STW1 bit 0 must be set to 0 by means of single-bit assignment in the **_disableAxis()** command or **_disableAxis()** with disableMode:=ALL. If this is not the case, a switch on of the axis will be rejected with alarm 20005: Type 1 reason 0x0100h (control signals to the PROFIdrive state machine specified incorrectly).

The status in **actorMonitoring.power** is displayed as specified in STW1 in versions up to and including V3.1. As of V3.2, it is displayed as per Bit0 - Bit2 in ZSW1.

The status indication in **actorMonitoring.driveState** is made in accordance with the specifications in STW1 bit 4 - STW1 bit 6, and will not be derived from the drive status.

The control word and the status word from the drive protocol are indicated in the **drivedata.stw** and **drivedata.zsw** system variables.

n-actual from the drive protocol is indicated in the **actorData.actualspeed** system variable (as of V4.0).

Specify control bits in STW1 via the application (without checking technology object) [expert]

As of V4.1 SP2, the **_disableAxis()** (**disableMode**:=STATE_MACHINE_CONTROL_BY_APPLICATION) setting can be used to directly specify the control bits in STW1 bit 0 - STW1 bit 6 using the **_setAxisSTW()** command without the TO testing correctness in accordance with the PROFIdrive profile state machine.

This makes it possible to switch a drive on and off that does not behave conform to the PROFIdrive profile state machine. A traversing of the drive using the TO or a motion variable generation using the TO for the drive is not possible in this status.

Another possible use is, for example, the stopping of large drives with closed brake in the magnetized state. If the speed controller enable is removed, after the re-setting of the speed controller enable, it is possible to continue motion again without switching the drive on again and so needing to pass through the S1 to S4 states.

The position controller must be switched to 'tracking' (**servoControlMode**=INACTIVE). This setting places the **control** indicator variable to INACTIVE, i.e. no movement commands can be performed. For independent drive transitions, e.g. S4->S5, no 20005 alarm will be generated in this state. The STATE_MACHINE_CONTROL_BY_APPLICATION mode is switched off with the **_enableAxis()** / **_disableAxis()** commands, with another setting of the mode and with **_resetAxis()**.

In the STATE_MACHINE_CONTROL_BY_APPLICATION mode,

- The **actorMonitoring.driveState** and **actorMonitoring.power** system variables indicate INACTIVE.
- The setting/resetting of the STW1 bit 0 - STW1 bit 6 bits is permitted directly using the **_setAxisSTW()** command.
- A fault message for the drive can be reset only using the TO control with **_resetAxisError()**. The control bit STW1 bit 7 (reset fault memory / fault acknowledge) remains in TO administration, a handling of STW1 bit 7 using **_setAxisSTW()** is not possible.
  This means a fault is reset uniformly using the TO, whereby the acknowledgement can be made using SCOUT/HMI or the user program.
- The TO axis no longer performs any monitoring for drive failure / independent disable of the drive. Other monitoring functions remain active, e.g. encoder monitoring, sign-of-life monitoring, drive error 20005: Reason 1 will be signaled and entered in the diagnostic buffer.
- All alarms are performed with RELEASE_DISABLE response as OPEN_POSITION_CONTROL response.

When the STATE_MACHINE_CONTROL_BY_APPLICATION mode is exited and so further handling of the state machine continues using the TO,

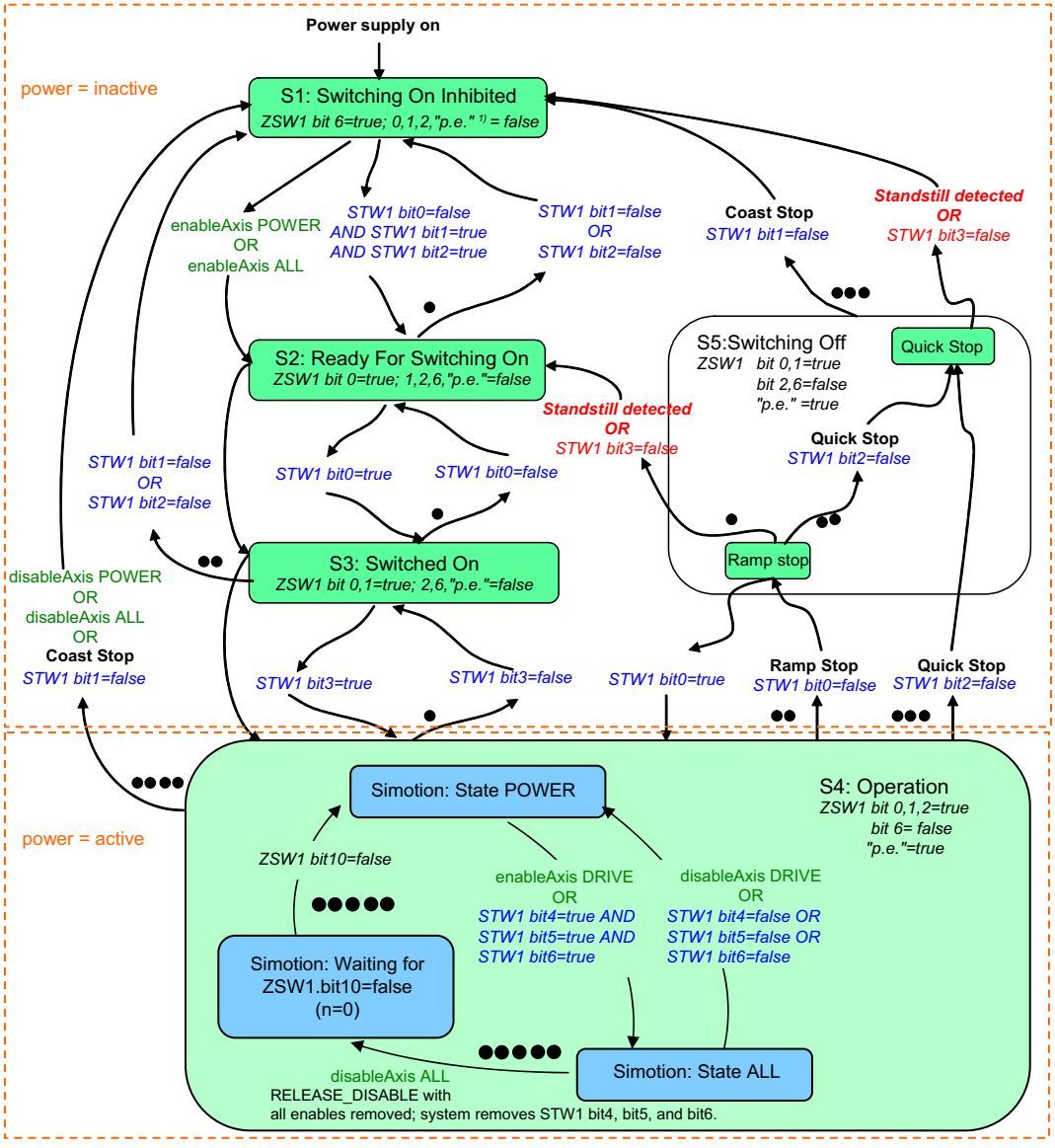
- The TO uses the control signals that were transferred last to the drive.
- When the mode is disabled, the current status of the state machine will be assumed. If the axis is activated in state S4, e.g. with **_enableAxis()**, no reactivation of the drive is possible. Consequently, the drive should be in a defined PROFIdrive S1-S4 state with regard to the drive, otherwise the TO will initiate the technological alarm 20005: type 1 reason 0x04 and the enables for the drive removed.

The status is indicated with the **actorMonitoring.stateMachineControl**=APPLICATION system variables.

See also

Alarm reactions (Page 353)

3.4.4.7 Status diagram - PROFIdrive General State Diagram



STW1 bit x,y = These control word bits must be set by the controller.
ZSW1 bit x,y = These status word bits indicate the current status.
1) Abbreviation: "p.e." = "pulses enabled" optional, not supported by SIMOTION
Standstill detected: an internal result of a Stop operation
●●●…: The number of points at the illustrated transitions determines the priority.
The more points, the higher the priority.
- With SIMOTION V3.2 and higher, two variations are possible for the parameter assignment of the _enableAxis() command:
POWER/DRIVE/ALL: The StateMachine is controlled by the Axis technology object.
BY_STW_BIT: The StateMachine is controlled by the user program.
- The drivestate system variable is active in all states S1-S4 if STW1 bits 4 to 6 are set to true.

Figure 3-12 General state diagram in SIMOTION

3.4.4.8 SIMOTION system behavior with a drive-autonomous OFF3 ramp

When an OFF3 ramp is triggered (by the SS1 or SS2 safety functions, for example), the behavior with regard to the shutting down of the SINAMICS drives during the OFF3 ramp varies dependent upon certain settings on the Axis technology object:

A) Behavior as of SIMOTION V4.2:

As of SIMOTION V4.2, drive-autonomous motion (triggered by SS1, for example), can be clearly identified even if the *Safety Integrated Extended Functions* have not been set; in other words, even if **TypeOfAxis.TechnologicalData.driveSafetyExtendedFunctionsEnabled**=NO.
This requires that the configuration data item **TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation** is set to YES (system setting when new axes are set up).
This means that drive enables can be canceled during the drive-autonomous state.

If **TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation** is set to COMPATIBILITY_MODE (for axes from projects converted upwards), the behavior from V4.1 applies. If DSDB (Page 253) is used, this setting is not allowed (**TypeOfAxis.TechnologicalData.numberOfDriveSafetyDataBlock**>0).

B) Behavior in SIMOTION V4.1:

If a drive-autonomous motion is triggered, it is not possible to cancel the drive enables during the autonomous state unless the Extended Safety Functions are set (**TypeOfAxis.TechnologicalData.driveSafetyExtendedFunctionsEnabled**=NO).

A drive-autonomous motion can be triggered by selecting SS1, for example. The drive enables cannot be removed during the drive-autonomous motion - neither through the user command **_disableAxis()** nor through an error on the axis with the local reaction RELEASE_DISABLE.

This is to be taken into account when designing software and hardware limit switches among other things.

If the Extended Functions are set (that is to say if **TypeOfAxis.TechnologicalData.driveSafetyExtendedFunctionsEnabled**=YES and **TypeOfAxis.DriveControlConfig.pulsesEnabled** are correctly configured), it is possible to cancel the drive enables during the autonomous state (e.g. by using the **_disableAxis()** command and the RELEASE_DISABLE error response).

For **TypeOfAxis.DriveControlConfig.pulsesEnabled** configuration, see Pulse enable evaluation (when standard settings are deactivated) (Page 273).

This needs to be taken into account, for example, in cases where a control has already been configured for a V4.1 CPU and an additional control is then added via PROFIsafe or TM54F to the OFF3 ramp via a terminal or to the Safety Integrated Basic Functions via terminals (mixed operation).

To avoid following error limit 50102 during the OFF3 ramp in this case, the following procedure is recommended:

1. Cyclic evaluation of the system variables
   - driveData.driveSafetyExtendedFunctionsInfoData.state.1=TRUE
   - driveData.driveSafetyExtendedFunctionsInfoData.state.2=TRUE
   - driveData.ZSW[0].5=FALSE
2. If a condition is fulfilled switch to follow-up mode.
   (e.g. with the command **_move()** at velocity = 0 and movingMode=SPEED_CONTROLLED)

See also

Safe Stop 1 (SS1)As of SIMOTION V4.1 (Page 264)

Safe Stop 2 (SS2)As of SIMOTION V4.1 (Page 266)

3.4.4.9 Technologies and telegram types

Table 3-16 Technologies and telegram types supported for real axis with digital drive link

Drive	Technology	Telegram type
SIMODRIVE 611U universal SIMODRIVE 611U universal HR	All	1 to 6, 101, 102, 103, 105, 106 [1]
SIMODRIVE POSMO CA/CD	All	1 to 6, 101, 102, 103, 105, 106
SIMODRIVE POSMO SI	All	1, 2, 3, 5, 101, 102, 105
SIMODRIVE sensor isochronous	External encoder	81
SIMODRIVE sensor (PROFIBUS)	External encoder	81
SIMODRIVE sensor (PROFINET)	External encoder	81, 83
MASTERDRIVES MC	All	1 to 6 [2]
MASTERDRIVES VC	Speed-controlled axis	1, 2
MICROMASTER 4xx	Speed-controlled axis	1
SINAMICS S120	All	1 to 6, 83, 102, 103, 105, 106, 125 [3], 126 [3]
SINAMICS Integrated (SIMOTION D)	All	1 to 6, 83, 102, 103, 105, 106, 125 [3], 126 [3]
SINUMERIK ADI4, SIMATIC IM174	All	3

[1] For further details, see also the SIMOTION SCOUT Configuration Manual and the accompanying 611U product description.

[2] For further details, see also the SIMOTION SCOUT Configuration Manual and the accompanying MD MC product description.

[3] As of SIMOTION V4.4 with SINAMICS V4.7

Table 3-17 Telegram types

Telegram type	Brief description
Standard telegrams	
1	n-set interface, 16-bit, without encoder
2	n-set interface, 32-bit, without encoder

Telegram type	Brief description
3	n-set interface, 32-bit, with encoder 1
4	n-set interface, 32-bit, with encoder 1 and encoder 2
5	n-set interface, 32-bit, with DSC and encoder 1
6	n-set interface, 32-bit, with DSC, encoder 1 and encoder 2
81	Encoder telegram with 1 encoder channel
83	n-act. interface, 32-bit, extended encoder telegram with 1 encoder channel
Siemens telegrams	
101	n-set interface
102	n-set interface with encoder 1 and torque limiting
103	n-set interface with encoder 1, encoder 2 and torque limiting
105	n-set interface with DSC, encoder 1 and torque limiting
106	n-set interface with DSC, encoder 1, encoder 2 and torque limiting
125	n-set interface with DSC with spline, encoder 1 and torque limiting
126	n-set interface with DSC with spline, encoder 1, encoder 2 and torque limiting

Information on the activation of technology data blocks can be found in Chapter Technology Data.

See also

Technology data (Page 179)

3.4.4.10 Coupling of digital drives

Separation of reference value and maximum value (as of V4.0 through < V4.2)

With V4.2 and higher, the reference/maximum velocity, reference/maximum torque, and reference/maximum force are adapted by the SINAMICS drive during runtime by default. The following is not relevant in this context.

For digital drive coupling, the reference value for sending the speed or velocity to the drive can be configured independently of the maximum value. Alternatively, the maximum speed/velocity can still be used as the reference value. This can be defined accordingly in the **driveData.speedReference** or **linearMotorDriveData.speedReference** configuration data element.

When a new axis is created and coupled with a digital drive, the normalization speed/velocity is pre-selected as the reference value by default in the axis wizard. The maximum speed/velocity can be entered independently.

With SIMODRIVE and MASTERDRIVE, this data must be entered manually. You must make sure that the reference values are the same on the drive and the controller.

In all cases, you still have the option of setting the maximum velocity as the reference velocity.

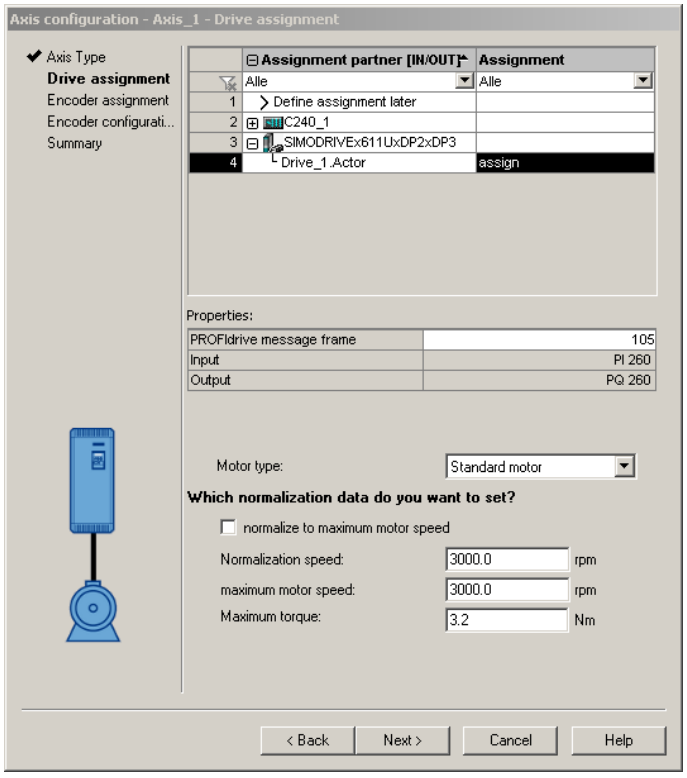


Figure 3-13 Setting the normalization speed and maximum motor speed

In the axis wizard, the drive settings are displayed as default settings.

For a coupling with SINAMICS, you can use the **Data transfer from the drive** button to transfer the **normalization speed**/velocity and the **maximum speed**/velocity from the offline configuration of the drive. If the relevant drive parameters are modified online, they must be uploaded and saved to the project prior to the data transfer.

SINAMICS Safety Integrated Extended Functions

See Overview - Support of SINAMICS Safety Integrated Functions on the Axis technology object (Page 245).

Drive communication based on DPV1 services

Drive communication based on DPV1 services facilitates acyclic communication with the drive, e.g. making it possible to read and write drive parameters.

For more information, see Drive communication based on DPV1 services (Page 222).

Time-of-day synchronization SIMOTION - SINAMICS

As of V4.2, SIMOTION features automatic time-of-day synchronization.

The time of day on the SINAMICS drive system is synchronized with the master SIMOTION system and the time ratio between SINAMICS messages and SIMOTION messages is correct.

For firmware versions < V4.2, see Drive communication based on DPV1 services (Page 222).

The following control units support time synchronization:

- SINAMICS Integrated of the SIMOTION D
- CX32/CX32-2 controller extensions
- SINAMICS S120 CU310, CU310-2, CU320, CU320-2 Control Units connected via PROFIBUS or PROFINET

Requirement: SIMOTION V4.4 and higher

See also

Overview - Support of SINAMICS Safety Integrated Functions on the Axis technology object (Page 245)

Adjustable response to RELEASE_DISABLE (Page 357)

Drive communication based on DPV1 services (Page 222)

Overview of torque limiting via torque reduction (Page 171)

3.4.5 Setting as a real axis with stepper drive C2xx (V3.2 and higher)

Possible drive types for stepper motors:

- Stepper motor with encoder
- Stepper motor without encoder
  In this case, stepper motor is entered as the encoder mode and the encoder side is not used.

When you select stepper motor without encoder on drive 1, encoder input 1 is assigned automatically since this data is traced back to it within the system. Drive 1 cannot be selected if the encoder input is no longer free.

For stepper motors, you can enter the number of steps and the stepping rate. The speed of rotation is then calculated and displayed.

3.4.6 Stepper drives on IM174 and stepper drives with PROFIBUS interface

The IM174 module is available as the interface for stepper drives for all SIMOTION platforms. From the SIMOTION perspective, stepper drives linked via IM174 behave like digital drive links.

Alternatively, stepper drives can be linked with a PROFIBUS interface if they support the PROFIdrive profile.

You can find more information in the *Distributed I/Os IM 174 PROFIBUS Module* Manual.

See also

Coupling of digital drives (Page 56)

3.4.7 Setting as a real axis with encoder signal simulation (V4.0 and higher)

With this setting, no drive/actuator is connected via the Axis technology object. An angle signal is output directly to a third-party control via the SINAMICS TM41 peripheral module (encoder simulation).

The output angle signal behaves just like the signal from an incremental encoder.

Enable signal

The enable signal to the drive is activated with **_enableAxis()** (**enableMode**=ALL); the enable is displayed by the system variables **actormonitoring.driveState** = ACTIVE and **actormonitoring.power** = ACTIVE.

See also

Encoder signal output (V4.0 or later) (Page 145)

3.4.8 Setting a non-exclusive drive assignment (as of V4.1 SP1)

The **typeOfAxis.setpointDriverInfo.InterfaceAllocation** setting is used to assign the drive/actuator of the axis to the drive, either exclusively (in versions up to and including V4.0) or non-exclusively (as of V4.1 SP1). For non-exclusive assignment, you must specify whether this drive interface should be activated when the technology object is ramped up.

With non-exclusive drive assignment, you can interconnect several Axis technology objects with one drive.

The **_enableAxisInterface()** and **_disableAxisInterface()** commands or the actor=YES function parameter are used to activate and deactivate the actuator interface during operation.

The Activated/Deactivated drive interface status is displayed in the **actorMonitoring.output** system variable.

See also

Non-exclusive encoder assignment (as of V4.1 SP1) (Page 78)

3.4.9 Setting as a real axis without drive (axis simulation)

It is possible to switch a real axis to simulation status even if the drive is not connected. The axis simulation status is displayed in the configuration dialog for the axis under drive and encoder assignment.

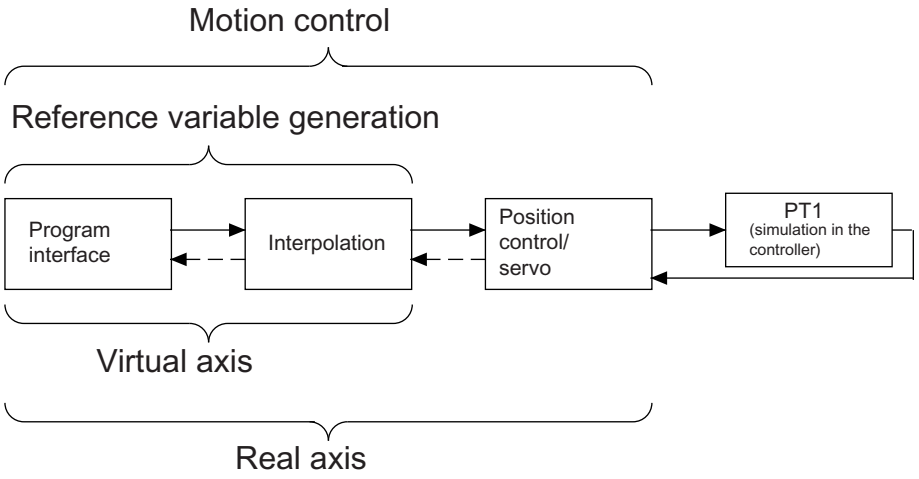


Figure 3-14 Axis simulation functional scheme

The dynamic behavior of the axis is simulated with the total time constant via an internal PT1 element.

The axis is thus available for the user program in the basic functions (e.g. setting of enables, execution of motion commands), but the manipulated variable is not output to the connected or missing drive.

There is no need for precise dynamic modeling of signals.

The simulation of an axis with an "analog output via I/O peripheral" drive assignment is not supported.

The complete axis configuration is kept when setting the simulation status and is still available without any changes after switching back in the expert list to the normal mode status.

Table 3-18 Settings in the configuration data

TypeOfAxis.SetPointDriverInfo.mode	= SIMULATION
TypeOfAxis.SetPointDriverInfo.outputNumberOnDevice	= 0
TypeOfAxis.NumberOfEncoders.Encoder_1.encoderIdentification	= SIMULATION
TypeOfAxis.NumberOfEncoders.Encoder_1.DriverInfo.encoderNumberOnDevice	= 0
TypeOfAxis.NumberOfDataSet.DataSet_x.ProcessModel.T1 (as of V4.0)	Time constant T1
TypeOfAxis.NumberOfDataSet.DataSet_x.ProcessModel.T2 (as of V4.0)	Time constant T2

Comments

- All real axis types can be set to SIMULATION.
- The process response/axis response is simulated from **processModel.T1** and **processModel.T2** via the PT1 element with the total time constant (T1+T2).

- Homing to the encoder zero mark and the external zero mark is not possible in Axis simulation mode.
- Output cam functions are executed.
- Only measuring input via time stamp is possible.
- If an axis with the DSC setting is set to SIMULATION, the position controller is calculated in the position control cycle clock in the control/simulation driver, i.e. it may be necessary to cancel the gain and adjust the following error monitoring.
- Unlike program simulation, the drive does not need to be present and the position controller remains active with simulated drive behavior.
- The simulation mode is provided for actuators and sensors. Technology data (such as Technology Data Block TDB, Safety Information Data Block SIDB, Drive Safety Data Block DSDB, etc.) of an axis cannot be simulated. Deactivate them before the axis is switched to simulation mode.
- The **_setAxisSTW()** command is not active in simulation mode.

As of SIMOTION V4.2

Axes with symbolic drive allocation

If symbolic allocation is set and if an axis allocated to a SINAMICS drive is to be removed from axis simulation, then this axis must be reallocated to the drive again.

Axes without drive assignment

If the axis is not yet assigned (see also Real and virtual axes (Page 36)), the logical addresses for the actuator and encoder must be converted.
To do this, set the following configuration data in the expert list to a free and valid address: The address must not be equal to 65535 and the same one must be used for all four items of configuration data, e.g. 15000.

Configuration data:

- **TypeOfAxis.SetPointDriverInfo.logAdressIn**
- **TypeOfAxis.SetPointDriverInfo.logAdressOut**
- **TypeOfAxis.NumberOfEncoders.Encoder_n.DriverInfo.logAdressIn**
- **TypeOfAxis.NumberOfEncoders.Encoder_n.DriverInfo.logAdressOut**

When deactivating the simulation, set each of the values for this configuration data back to 65535.

Note

A script is available to switch the axis simulation on and off for axes with and without symbolic drive assignment. This script can be integrated into a project via the ProjectGenerator (accessible from SCOUT). You can find the script in *SIMOTION Utilities & Applications*, which is part of the scope of delivery of SIMOTION SCOUT.

See also

Enabling/disabling program simulation (Page 341)

Coupling of digital drives (Page 56)

3.5 Encoders and encoder parameters

3.5.1 Overview of encoders and encoder parameters

Sensing of the actual position value can be performed using:

- Motor measuring system (motor encoder)
- Additional direct measuring systems (external encoders), if necessary

The adaptation of relevant drive data is activated automatically as of SIMOTION V4.2 in conjunction with SINAMICS S120 as of V2.6.2. The encoder parameters are adapted automatically.

A direct measuring system measures the technological parameter directly, i.e. without the interference of influences such as torsion, backlash, slip, etc.
This may facilitate improved smoothing of mechanical influences by means of the closed-loop control.

Up to eight encoders can be generated on the axis. All generated encoders are internally active, and the measured values are updated cyclically. It is possible to switch between the encoders via data sets. Whether an encoder is active for closed-loop control is displayed in the **sensorData[n].control** system variable.

Several data sets can be assigned to one axis. These data sets can contain different encoders. A data set is active.
Encoders can be added and parameterized in the Axis technology object under **Configuration** on the **Encoder configuration** tab.

Via the
typeOfAxis.numberOfEncoders.Encoder_1.sensorControlConfig.tolerateSensorDefect setting (V4.0 and higher), it is possible to tolerate the failure of an encoder which is not selected or which is not involved in the closed-loop control. This setting is defined for individual encoders. It is output via **Alarm 20015**.

For more information on data set switchover and encoder configuration, see Data set switchover / encoder switchover (Page 200).

See also

Data set overview (Page 200)

3.5.2 Encoder for position

The controller uses an encoder to detect the axis position.

From a technological standpoint, the following encoder types can be differentiated:

- Incremental encoder
  On the controller side, only the difference between two read encoder values is evaluated. Values are read out in the servo cycle clock at equal intervals. In order to determine the mechanical axis position, the axis must be homed each time the system is switched on. Position zero is displayed after it is switched on.
- Absolute encoder
  This encoder supplies the absolute value or, in the case of an absolute value in the PROFIdrive message frame, the absolute value is read one time after the system is switched on. After this, the actual value is processed in the same way as with the incremental encoder.
  The absolute value supplied by the encoder is assigned to the associated mechanical axis position by means of the absolute encoder adjustment. The absolute encoder adjustment occurs only one time, and the controller remembers the compensation value/absolute encoder offset even after it is powered on/off.
  Certain situations, such as an encoder failure, a restart, etc., can require an absolute encoder re-adjustment. For more information, refer to the chapter titled *Absolute encoder homing/ Absolute encoder adjustment*.
  The following absolute encoder types are differentiated:
  - Absolute encoder with absolute encoder setting
    The measuring range of this encoder is greater than the axis traversing range.
    The axis position results directly from the current encoder value since this can be uniquely mapped.
    An offset may be entered - it is not necessary to hold overflows within the controller.
    There are no overflows of the absolute actual value stored when SIMOTION is switched off. The next time it is switched on, the actual position value is generated from the absolute actual value only.
  - Absolute encoder with cyclic absolute encoder setting
    The axis traversing range is greater than the range of values measured by the encoder, and the encoder returns an absolute value within its measuring range.
    The controller also counts the number of measuring ranges internally in order to hold a unique actual axis position beyond the range of measured values.
    When SIMOTION is switched off, the overflows of the absolute actual value are stored in the retentive memory area of SIMOTION. The next time it is switched on, the stored overflows are taken into account for the calculation of the actual position value.
    The actual position of the axis is passed internally in a 64-bit integer variable.
    Example of a single-turn encoder with 4,096 increments:
    The position of the encoder is mapped in bits 0 to 11, and the number of overflows of the encoder range in bits 12 to 63.
    Example of a multi-turn encoder with 4,096 x 4,096 increments:
    The position of the encoder is mapped in bits 0 to 23, and the number of overflows of the encoder range in bits 24 to 63.
    The overall position of the axis is retained after the controller is switched off. When the controller is switched on again, if the actual encoder value does not match the actual position stored in the controller, it will be corrected to a maximum of ± ½ the encoder measuring range.

    Note

    If the axis/encoder is moved by more than one half the encoder measuring range with the controller switched off, the actual value in the controller no longer matches the real axis.

See also

Overview of homing (Page 84)

3.5.3 Encoder for velocity

Encoders for detection and display of the speed/velocity can only be applied to speed-controlled axes.

Options are:

- Incremental encoders/absolute encoders with number of increments or pulses/revolution (for electric axes)
- Interval counters (for hydraulic axes)
- Encoders providing the velocity as a direct value in the I/O area (for hydraulic axes)
- Read-out of the speed from the PROFIdrive message frame and provision for technological functionality, e.g. velocity monitoring

3.5.4 Encoder assignment and terminology

The encoder type is specified with the encoder mode.

Table 3-19 Definable encoder mode depending on the encoder type

Encoder mode	Encoder type		
	Absolute encoder	Cyclic absolute encoder	Incremental encoder
PROFIdrive (with adaptation [1]) (as of V4.2)	x	x	x
Endat (encoder data interface)	x	x	x
SSI (synchronous serial interface)	x	x	-
Sinusoidal	-	-	x
Rectangle	-	-	x
Resolver	-	x [2]	x
Analog encoder (value in the I/O area)	x	-	-

[1] The adaptation setting (set by the system by default with V4.2 or higher and SINAMICS) makes changes based on the encoder set in SINAMICS. The encoder resolution and fine resolution are adopted during runtime.

[2] Possible with single-pole-pair resolver only

When the encoder signals are evaluated in an encoder module or drive, the actual values are transferred to the controller in the normalized format according to the PROFIdrive profile. The type of measured value acquisition is dependent on the drive or encoder module, e.g. Sinus, SSI or Endat. The PROFIdrive encoder mode can therefore be set in the controller (as of V4.2).

As of SIMOTION V4.2, the system makes the settings for communication between SIMOTION and SINAMICS drive (encoder). The project setting **Use symbolic assignment** is activated by

default for new projects as of V4.2. Telegrams are set automatically. Encoder characteristics such as fine resolution, grid spacing, and the absolute value for data width are adapted automatically when the system ramps up.

For encoder data, refer to the data sheet or type plate of the encoder. With SINAMICS, encoder data can be transferred from the drive.

The rest of this chapter is only relevant if the project default setting **Use symbolic assignment** is deactivated as of V4.2, or if you are working with a project which uses SIMOTION V4.1.

Encoder pulses per revolution

The encoder pulses per revolution is specified on the encoder type plate as the number of signal periods per revolution (incremental encoder: Pulses/rev; absolute encoder: Pulses/rev; resolver: Number of pole pairs (for SINAMICS and MASTERDRIVES)).

Configuration data:

- **AbsEncoder.absResolution**
- **IncEncoder.incResolution**

Grid line spacing (linear encoder system)

The grid line spacing is specified on the type plate of the encoder as the distance between lines on the linear measuring system (linear scale).

Configuration data:

- **Resolution.distance**

Fine resolution

The fine resolution of the actual value is the interpolation result of a signal period of an encoder pulse.

The fine resolution steps are generated by the measuring electronics from the raw signal of the encoder pulses. Factors as a multiple of 2 are possible.

Example:

- A rectangle signal has a fine resolution of 1
- Two rectangle tracks (TTL signal) offset by 90° have a fine resolution of up to 4
- Depending on the measuring electronics, a sinusoidal signal can have any fine resolution, in principle, e.g. 2,048

Depending on the defined encoder type, the default value 0 is interpreted differently in SIMOTION (see table: *Default settings for fine resolution in SIMOTION*).

In SIMOTION, the multiplication factor is specified, rather than the shift factor/number of bits (x).

The actual value, including the fine resolution, is indicated in the **sensorData.incrementalPosition** system variable.

Configuration data:

- **AbsEncoder.absResolutionMultiplierCyclic**
- **IncEncoder.incResolutionMultiplierCyclic**

Data width of absolute value (without fine resolution) for absolute encoders

The data width of the absolute value (without fine resolution) is a result of the sum of the bits for representing the number of encoder pulses and the bits for representing the maximum number of revolutions that can be registered by the encoder according to the type plate.

Example:

4,096 pulses/rev (= 2^{12}) and a maximum of 4,096 revolutions that can be registered yields a 12 + 12 = 24-bit data width of the absolute value.

Configuration data:

- **AbsEncoder.absDataLength**

Fine resolution of absolute value in Gn_XIST2.

This parameter for the format of the Gn_XIST_2 is only relevant for encoders via PROFIdrive telegram (for more information, see *Encoder interconnection via PROFIdrive telegram*).

Configuration data:

- **AbsEncoder.absResolutionMultiplierAbsolute**

The fine resolution of the absolute value in Gn_XIST2 must be less than or equal to the fine resolution of the absolute value in Gn_XIST1.

Default setting for fine resolution.

Depending on the encoder mode, the default settings are evaluated by the system as described in the table below. The default settings are used if 0 is assigned to the value.

Table 3-20 Default settings for fine resolutions in SIMOTION

Encoder type	Encoder mode	Fine resolution (Gn_XIST1)	Fine resolution on the absolute value (Gn_XIST2)
Onboard C2xx			
Incremental encoder	Rectangle	4	-
	Stepper motor	1	-
Absolute encoder	SSI	1	1
Encoder in PROFIdrive axis telegram (applies to SINAMICS, SIMODRIVE 611U and MASTERDRIVES)			
Incremental encoder	Rectangle	2048	-
	Sinusoidal	2048	-
	Resolver	2048	-
	Endat	2048	-
Absolute encoder	Endat	2048	512
	SSI	1	1
...Cyclic absolute	Resolver (only pole pair number 1 possible)	2048	512
	Endat	2048	512
	SSI	1	1
PROFIBUS absolute encoder in PROFIdrive encoder telegram			
Absolute encoder	SSI	$2^{(32 - \text{Number of data bits})}$	1

These settings are aligned to the default parameter settings of the corresponding Siemens devices. If the behavior is different, alignment with the encoder should be performed by making a corresponding entry in the configuration data of the technology object or in the parameters of the drive or encoder. Depending on the device it may be necessary to assign the corresponding parameters in the drive or encoder with the value of the exponent (shift factor).

Device-specific features for Masterdrives with Endat encoders:

For Masterdrives with Endat encoders, Endat or SSI can be selected in the encoder mode. However, the fine resolution must always be configured in the axis wizard of the Axis or External Encoder technology object differently from the default settings (see Encoder list (Page 67)).

Default setting:

- Fine resolution (**~Encoder.~ResolutionMultiplierCyclic**) = 0
- Fine resolution of the absolute encoder in Gn_XIST2 (**AbsEncoder.absResolutionMultiplierAbsolute**) =0
- If encoderMode=PROFIDRIVE, the values are evaluated as available, since the correct values are adapted by the drive. A default setting of 0 is not permitted for this setting.

See also

Encoder interface as a direct value in the I/O area (Page 72)

Encoder list (Page 67)

3.5.5 Encoder list

For the most up-to-date list of encoders you can use with SIMOTION in conjunction with SINAMICS, SIMOVERT-MASTERDRIVES, and SIMODRIVE 611U, go to **http://support.automation.siemens.com/WW/view/de/18769911**.

The encoder list is also documented under *FAQs > Drives > Parameters of the connectable encoders* in the *SIMOTION Utilities & Applications* and in the online help (look for encoder parameter assignment in the index). *SIMOTION Utilities & Applications* is part of the scope of delivery of SIMOTION SCOUT.

3.5.6 Onboard encoder interface on SIMOTION C2xx

Incremental encoders with TTL signal and absolute encoders with SSI protocol can be connected directly to C230-2 or C240. (See the C230-2 and C240 operating instructions and Encoder list (Page 67))

See also

Encoder list (Page 67)

3.5.7 Encoder interface using the PROFIdrive message frame

As of SIMOTION V4.2, the system sets up communication between SIMOTION and SINAMICS drive (encoder). For SINAMICS drives and encoders with V4.2 and higher, encoder resolution data is transferred directly from the drive during runtime. Telegrams are set automatically.

The rest of this chapter is only relevant if the project default setting **Use symbolic assignment** is deactivated as of V4.2, or if you are working with a project which uses SIMOTION V4.1.

The encoder values are transmitted in the PROFIdrive telegram (see Table *Telegram types* in Section *Setting as a real axis with digital drive coupling*).

Encoder forced values, status values, and actual values are transmitted in the PROFIdrive telegram.

The encoder behavior on SIMOTION is set as represented in the PROFIdrive protocol.

The encoder parameters are defined via the drive wizard during drive configuration (either user-defined or by selecting the encoder).

Encoder parameters that are entered subsequently in the SIMOTION axis wizard must match the encoder parameters in the drive.

Note

For SINAMICS drives with earlier versions than SIMOTION V4.2, it is possible to transfer the encoder parameters from the drive. When assigning encoders in the axis wizard, click **Data transfer from the drive**.

If you are using a DRIVE-CLiQ component with electronic type plate (e.g., SMI motor, DRIVE-CLiQ encoder) you must first upload the parameters from the drive and save them in the project (online commissioning). If the online commissioning is carried out at a later point, you can work with the default settings of the axis wizard in the meantime during offline configuration. Once online commissioning is complete, upload the drive parameters, save them in the project, run the axis wizard again, and perform the **Data transfer from drive** function.

If you change the encoder data in the drive, you must perform an alignment again in the axis wizard.

Further sources of information:

- Encoder list (Page 67)
- User Manual *SIMODRIVE sensor Absolute encoder with PROFIBUS-DP*
- Operating Instructions *Absolute encoder with PROFINET IO* (Chapter *Operating with SIMOTION*)

Encoder value via PROFIdrive axis telegrams

For further information, refer to the commissioning manuals for the drives.

The first and (if present) second encoder of the PROFIdrive axis telegram can be assigned freely to an External Encoder technology object or to the encoder of an axis.

Encoder value via PROFIdrive encoder telegram 8x

PROFINET/PROFIBUS encoders can be set in accordance with the current specifications of the encoder profile with telegram type 81 and as of V4.2, with telegram type 83. These encoders can be assigned freely. See also **PROFINET/PROFIBUS absolute encoder via PROFIdrive encoder telegram** in the next chapter.

Inconsistent configuration

In the event of errors or inconsistencies between the configuration data in SIMOTION and the parameter settings for the encoder in the drive, a technological alarm is triggered as soon as an online connection is established between the control and the drive/encoder.

As off SIMOTION Runtime V4.4: **Technological alarm 20025** with applicable reason

In SIMOTION Runtime < V4.4: **Technological alarm error 20005: Device type:2, log.address:1234 faulty. (Bit:0, reason: 0x80h)**

For PROFIdrive encoders and encoders on the axes according to PROFIdrive, a comparison of the parameter assignment takes place via the following drive/encoder parameters:

P979 (SensorFormat) according to PROFIdrive, which contains information about the type, resolution, and shift factors.

For drives or encoders that do not support parameter P979, the configuration data is evaluated as valid without alarm message.

Actual value Gn_XIST1

The incremental actual value is transferred cyclically with the defined fine resolution in Gn_XIST1. The incremental actual value in Gn_XIST1 is steadily continued according to the actual value change and reset when the data width of Gn_XIST1 is exceeded. If operating with incremental and absolute encoders, the control evaluates the incremental actual value in Gn_XIST1 according to the settings made for encoder pulses per revolution and fine resolution, or grid line spacing for linear scaling.

When the controller is switched on, the fine resolution value within one encoder signal period is indicated correctly in Gn_XIST1. The initial value for the number of signal periods is set by the drive/encoder, and the actual value can then be steadily continued from this initial value.

In the PROFIdrive profile, the fine resolution is given as "shift factor" (x).

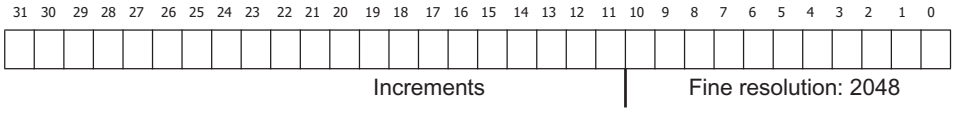


Figure 3-15 Example composition of the 32-bit encoder data of the cyclic actual value Gn_XIST1

Example of for an encoder with number of encoder pulses = 2048 (data width, 11 bits)

The fine resolution in SIMOTION in the **Inc/AbsResolutionMultiplierCyclic** configuration data element is set to the default setting 0 and is thus evaluated as a default fine resolution of 2048 (the default value depends on the encoder mode setting; see Table *Default settings for fine resolution in SIMOTION*).

SIMODRIVE 611U:

Table 3-21 Settings

SIMOTION		611U	
Encoder pulses per revolution [1]	=2048	P1007	=2048
Fine resolution [2]	=0 (≡ 2048)	P1042	=11

[1] Inc/AbsEncoder.Inc/AbsResolution

[2] Inc/AbsEncoder.Inc/AbsResolutionMultiplierCyclic

SINAMICS:

Table 3-22 Settings

SIMOTION		SINAMICS	
Encoder pulses per revolution [1]	=2048	P408	=2048
Fine resolution [2]	=0 (≡ 2048)	P418	=11

[1] Inc/AbsEncoder.Inc/AbsResolution

[2] Inc/AbsEncoder.Inc/AbsResolutionMultiplierCyclic

Note the information about the SINAMICS alignment.

Actual value Gn_XIST2

If the positions for the measuring input or homing functions are transferred to Gn_XIST_2 (n = 1 or 2, number of encoder), they are transferred with the fine resolution defined for the encoder. When the absolute value is read, the value in Gn_XIST_2 is evaluated based on the settings for the data width of the absolute value (without fine resolution) in **AbsEncoder.absDataLength** and the fine resolution absolute value in Gn_XIST2 is evaluated in **AbsEncoder.absResolutionMultiplierAbsolute**.

The fine resolution of the absolute value in Gn_XIST2 indicates the fine resolution factor included in the absolute value transfer. This can match the fine resolution of the actual value, but it can also be smaller, for example, if the 32-bit data width in Gn_XIST2 is not sufficient for the entire fine resolution factor as a result of the data width of the absolute value (without fine resolution).

Example:

Encoder pulses per revolution = 2048 (11 bits) and multi-turn resolution of 4,096 revolutions (12 bits)

Thus, the data width of the absolute value without fine resolution is 11 bits + 12 bits = 23 bits.

Therefore, 9 bits remain for the fine resolution in Gn_XIST2 (32 bits - 23 bits = 9 bits). The setting 0 for the fine resolution of the absolute value in Gn_XIST2 is thus evaluated by the system as 512 (= 9 bits).

Table 3-23 Setting the encoder data

Encoder pulses per revolution [1]	2048
Data width of absolute value (without fine resolution) [2]	23

Fine resolution of absolute value in Gn_XIST2 [3]	0 (= 512)
Fine resolution [4]	0 (= 2048)

[1] AbsEncoder.AbsResolution

[2] AbsEncoder.absDataLength

[3] AbsEncoder.absResolutionMultiplierAbsolute

[4] AbsEncoder.AbsResolutionMultiplierCyclic

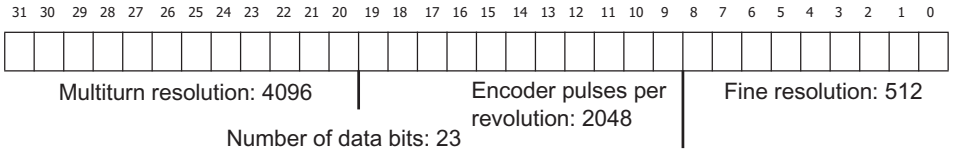


Figure 3-16 Example composition of the 32-bit encoder data of the absolute actual value Gn_XIST2

The number of bits resulting from the data width of the absolute value (without fine resolution) and the number of data bits for the fine resolution of the absolute actual value must not exceed 32. If it is less than 32, leading zeroes are added in Gn_XIST2.

Resolver in PROFIdrive axis telegram

For SINAMICS and MASTERDRIVES, parameters are assigned for the pole pair number of the resolver rather than the encoder pulses per revolution (example: 8-pole resolver = 4 pole pairs → input value = 4).

With SIMODRIVE, parameters are assigned for the encoder pulses per revolution based on parameter P1011.2

As of V4.1 SP1, the resolver with pole pair number 1 is supported as an absolute encoder with the cyclic absolute setting. (encoder pulses per revolution = 1, data width of the absolute value = 0, default value evaluation: fine resolution = 2048, fine resolution Gn_XIST2 = 512)

When a 1-pole resolver is used as Endat encoder, the p418(XIST1) and p419(XIST2) parameters must be set to "11" to prevent information loss of the absolute position. (Settings on the axis: absolute encoder, cyclic absolute, Endat, line count = 1, fine resolution = 2048, fine resolution absolute value = 2048, data width = 0)

See also the Encoder list (Page 67).

PROFINET/PROFIBUS absolute encoder via PROFIdrive encoder telegram

The data width of the encoder value in the technology object configuration data in SIMOTION must match the parameter settings for the PROFINET/PROFIBUS absolute encoder in HW Config.

See also the Encoder list (Page 67).

Example:

Parameter settings of a PROFIBUS absolute encoder in HW Config with 24-bit data width of the absolute value.

The 'SIMODRIVE isochronous sensor' PROFIBUS absolute encoder in HW Config is defined according to the default setting for 24-bit data width and encoder pulses per revolution of 4096: measuring steps per revolution = 4096
24-bit data width for the total resolution yields 0x01000000 (32-bit HEX number). This number, represented separately in HighWord and LowWord, equals 0x0100 in the HighWord and 0x0000

in the LowWord. The decimal values of these two parts (0x0100 = 256 decimal) are to be entered as follows:
total resolution (high) = 256
total resolution (low) = 0

This results in the following consistent configuration for the technology object:
the encoder value is transferred left-justified to Gn_XIST1; the unused bits of the fine resolution are set to 0 according to PROFIdrive, but must be specified in the fine resolution of the actual value. This results in a fine resolution of 8 bits (32 bits - 24 bits = 8 bits) (2^8 = 256 as a factor).

According to the setting above, the absolute value in Gn_XIST2 has a right-justified alignment and therefore a fine resolution of the absolute value in Gn_XIST2 of 0 bits (2^0 = 1 as a factor).

Encoder via PROFIdrive axis telegram on ADI4 and IM174

At least one electric or hydraulic axis must be configured on ADI4/IM174.

The defined update rate (BaudRate) for SSI encoders must be supported by the encoder.

See also the Encoder list (Page 67).

For more information about configuration and operation, refer to the *ADI4 - Analog Drive Interface for 4 Axes Manual* and *Distributed I/Os IM 174 PROFIBUS Module Manual*. These documents are provided on the *SIMOTION Documentation* DVD.

Note

An encoder in a PROFIdrive axis telegram can only be used for an external encoder TO or for the encoder of a TO axis in cyclical operation if a TO axis is also created on the PROFIdrive telegram and is not deactivated.

See also

Coupling of digital drives (Page 56)

Encoder list (Page 67)

3.5.8 Encoder interface as a direct value in the I/O area

Encoders can be used that

- Provide actual value information directly as absolute value in the input/output area.
- Provide a counter value in the peripheral area (as of V4.0).
- Provide an actual velocity in the peripheral area.

Actual value information directly as absolute value

These encoders must be parameterized and operated as absolute encoders, for example with respect to homing.

The following settings are provided to set the **adaptation to the properties of the measured value**:

- The **justification of the measured value** in the **NumberOfEncoders.Encoder_n.AnalogSensor.DriverInfo.format** configuration data element
  - Signed left-justified (VALUE_LEFT_MARGIN)
  - Signed right-justified (VALUE_RIGHT_MARGIN)
  - Unsigned left-justified (VALUE_LEFT_MARGIN_WITHOUT_SIGN)
  - Unsigned right-justified (VALUE_RIGHT_MARGIN_WITHOUT_SIGN)
- The **data width of the measured value** without the sign bit in the **NumberOfEncoders.Encoder_n.analogSensor.DriverInfo.resolution** configuration data
- The upper limit and lower limit, the **maximum limits of the measured value** in the following configuration data
  - **NumberOfEncoders.Encoder_n.AnalogSensor.DriverInfo.maxValue**
  - **NumberOfEncoders.Encoder_n.AnalogSensor.DriverInfo.maxValue**

Example: Use of an ET 200S, SSI module, or an analog input.

Justification of the measured value

The measured value is mapped to an internal 32-bit wide signed data value of the DINT type in accordance with the justification specification. The mapped value is then tested with actual value limitation against the maximum limits and evaluated using the weighting factor for the **NumberOfEncoders.Encoder_n.AnalogSensor.ConversionData.factor** direct value that specifies the technological resolution or assignment of the LSB.

⚠ WARNING

Weighting factor for analog sensors

The scaling or conversion in the value of the configuration data **ConversionData.factor** for the position of the analog sensor has changed in SIMOTION V4.4.

As of V4.4, the following applies:

The values are entered directly in inches and no longer converted from mm to inches for the configured Inch unit system. This can result in a higher velocity being detected (by a factor of 25) if it is an axis that works in inches.

The following applies for SIMOTION up to V4.4:

The values were entered in mm and then converted to inches (1 inch corresponds to approx. 25 mm). Please check the values entered for **ConversionData.factor** in older projects and correct them when necessary.

Regardless of the sign, the encoder resolution/measured value width is limited to maximum 31-bit, because the configuration data of the maximum limits are signed data values of the DINT type.

For the **Left-justified/Right-justified without sign** setting, the measured value for the setting of the encoder resolution/measured value width is processed as follows:

- At ≤16 bits the measured value left-justified/right-justified is mapped to the least-significant byte 1 and byte 2 of an internal data value of the DINT type. In this process, the missing 16 bits minus the measured value width of the least-significant byte 1 and byte 2 are right/left-padded with zeros and the most-significant byte 3 and byte 4 are filled with zeros.
- > At 16 bits the measured value left-justified/right-justified is mapped to an internal data value of the DINT type. In this process, the missing 32 bits minus the measured value width are right/left-padded with zeros.
  As this mapped measured value is tested against the maximum limits of the data type DINT, the encoder resolution/measured value width is limited to maximum 31 bits. This means the measuring range is limited to maximum 50% of the measured value width.

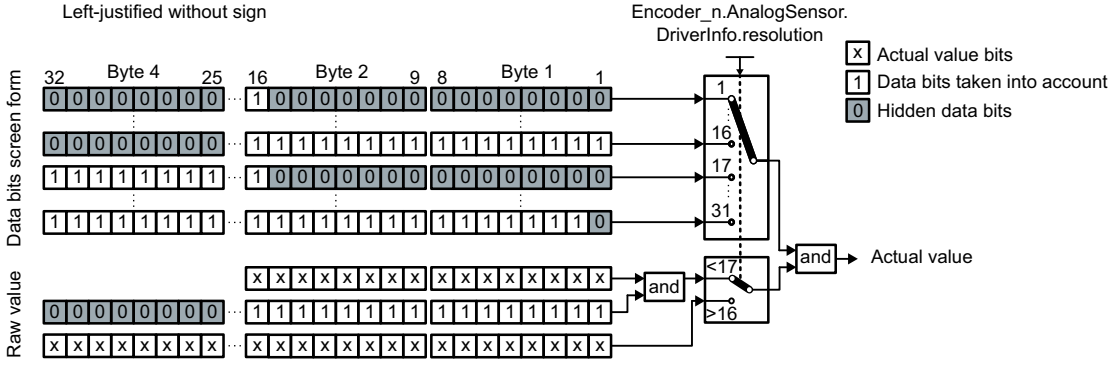


Figure 3-17 Masked reading of the analog IO value left-justified without sign

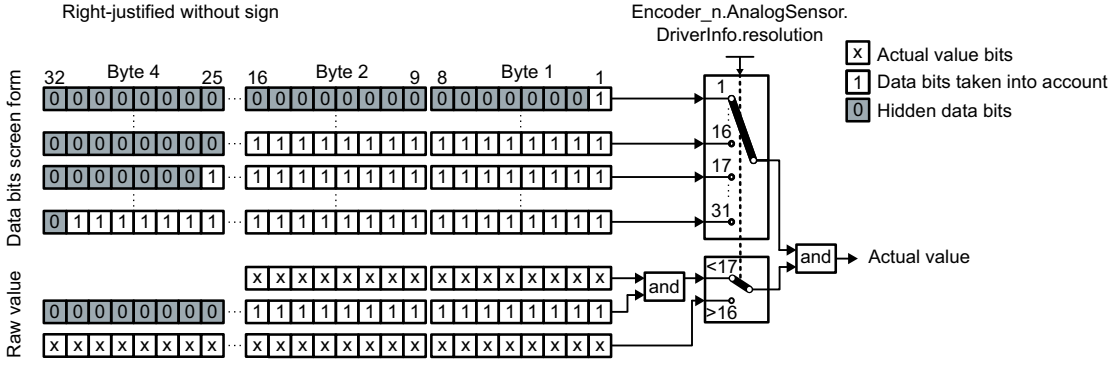


Figure 3-18 Masked reading of the analog IO value right-justified without sign

For the **Left-justified/Right-justified without sign** setting, the measured value for the setting of the encoder resolution/measured value width is processed as follows:

- At <16 bits the measured value left-justified/right-justified is mapped to the least-significant byte 1 and byte 2 of an internal data value of the DINT type. For left-justified, the missing 15 bits minus measured value width of the least-significant byte 1 and byte 2 are right-padded with zeros and the sign bit is extended left to bit 32. For right-justified, the sign bit is padded left to bit 32.
- At ≥16 bits the measured value left-justified/right-justified is mapped to an internal data value of the DINT type. For left-justified, the missing 31 bits minus measured value width are right-padded with zeros. For right-justified, the sign bit is padded left to bit 32.

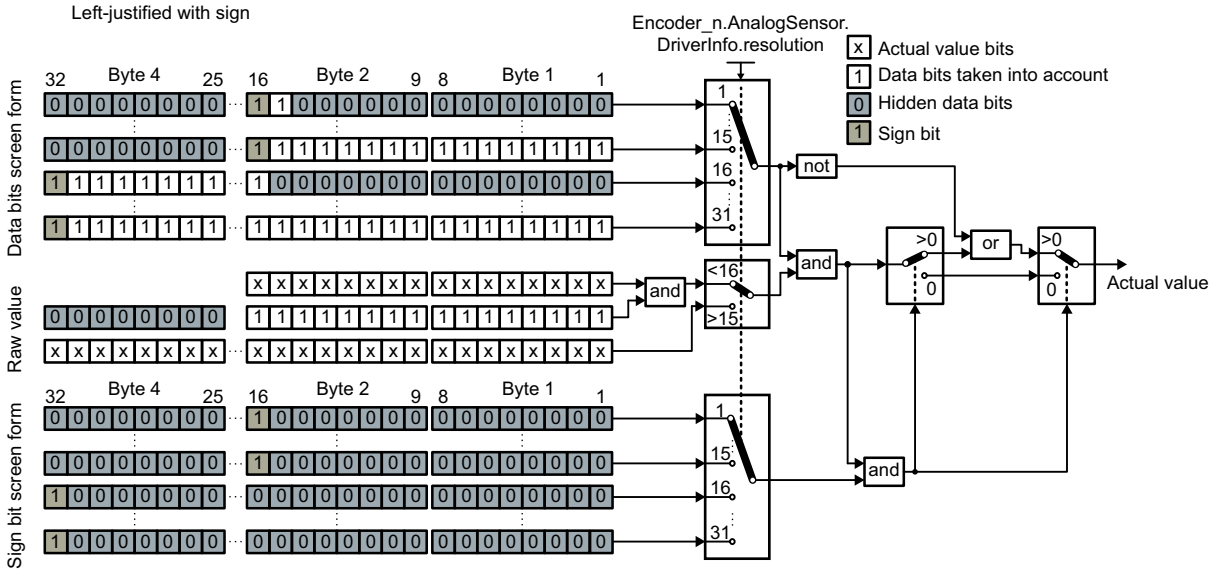


Figure 3-19 Masked reading of the analog IO value left-justified with sign

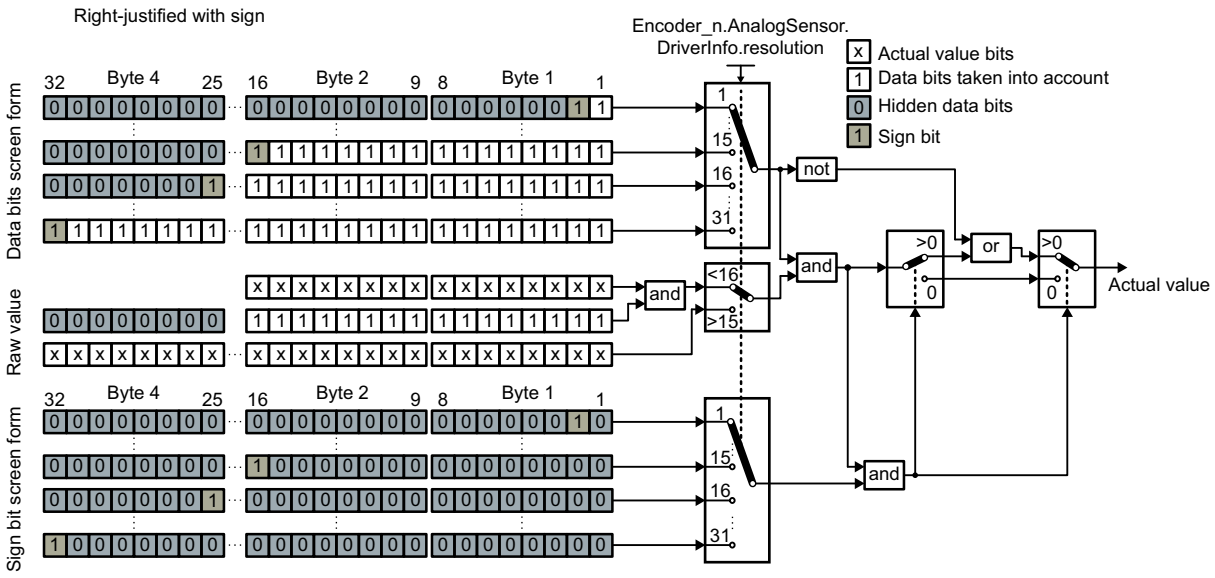


Figure 3-20 Masked reading of the analog IO value right-justified with sign

Counter value (as of V4.0)

The encoder is set as an incremental encoder. 16 bits or 32 bits can be set as counter value width.

Example: Use of an ET 200S, COUNT module

Actual velocity

The actual value information can include the number of pulses between two scans or, alternatively, the time interval between two sequential pulses. This type of encoder is used, for example, to measure actual velocities with the hydraulic axis functionality.

Example: Use of an ET 200S, COUNT module

Direct value as absolute value in I/O area (as of V4.1 SP1)

The following bits can be configured for the direct value as absolute value in the I/O area:

- **Ready bit** via the elements of the **analogSensor.readyStateMonitoring** configuration data element
- **Error bit** via the elements of the **analogSensor.errorStateMonitoring** configuration data element

These can be used for evaluation on the Axis technology object of a *ready identifier* and an *error identifier*, which are available from the peripheral module in addition to the measured value.

In SIMOTION V4.1 SP1, this configuration data is defined directly in the expert list.

If the *not ready* status or an error is displayed in these identifiers during operation, and the **ready bit** or **error bit** is configured, technology alarm 20005 is issued with the *sensor error* identifier.

If the Axis technology object is ready during ramp-up, but the direct value in the I/O area is not yet in ready status, the *WAIT_FOR_VALID Sensor* status is displayed on the encoder. (**sensorData[n].state** system variable)

As of V4.1 SP1, the direct value in the I/O area does not have to be updated in every equidistant communication cycle (for example, if an encoder connected to the peripheral module is not able to supply a new measured value in each cycle during a fast communication cycle clock due to measurement reasons or processing time reasons). In such cases, the actual value is extrapolated by the control.

The control supports the following options:

- The peripheral module displays the new measured value set in an update bit/update counter. The update bit or update counter is defined in the **analogSensor.UpdateCounter** configuration data element.
  UpdateCounter configuration: The update counter can be one (toggle) bit or several (counter) bits in width.
- The refresh cycle of the actual value in the peripheral module is known and is defined directly in the **analogSensor.UpdateCounter.updateCycle** configuration data element.
  Default setting with refresh cycle = 1 (default behavior for updating in every cycle clock)

3.5.9 Encoder value via system variable

A **position or velocity value** can also be defined directly in a system variable (sensorSettings.actualValue) from the user program, for example. With this value, it is possible, for example, to simulate the actual technological value for an axis or to provide the value from any encoder/I/O value (for which there does not exist a TO interface) via the user program and to apply it to the axis as an actual value.

This behavior is defined in **TypeOfAxis.numberOfEncoders.Encoder_x.encoderIdentification**=SET_ACTUAL_VALUE.

This encoder type can be defined on the axis and the external encoder.

The update rate for the default setting (**updateCycle**) and the maximum number of servo clocks (**maxFailures**) in which the system variable is not written are specified in the **StructAxisSensorSetActualValueConfig** configuration data structure.

The **sensorSettings.actualValue** system variable must be provided in the cyclical level set in the **typeOfAxis.NumberOfEncoders.Encoder_x.SensorSetActualValue.updateCycle** configuration data. If, for example, in the user program of the IPOsynchronousTask, the actual position is written cyclically to **sensorSettings.actualValue**, **~.updateCycle** = IPO must be set.

The value specified in **sensorSettings.actualValue** is accepted using the set refresh rate (**SensorSetActualValue.updateCycle**) and extrapolated linearly in between.

SensorSetActualValue.maxFailures always refers to the servo clock. The **SensorSetActualValue.updateCycle** setting does not have any affect.

Example:
If the **sensorSettings.actualValue** system variable is updated in the IPO cycle clock and IPO:Servo is set 2:1, **SensorSetActualValue.maxFailures** = 1 must be set.

3.5.10 Non-exclusive encoder assignment (as of V4.1 SP1)

The sensor/encoder of the axis can be defined as exclusive or non-exclusive in **typeOfAxis.NumberOfEncoders.Encoder_x.interfaceAllocation**. If it is defined as non-exclusive, you must also specify whether the encoder interface should be activated when the technology object is ramped up.

The functionality supports the configuring of an encoder on several axes. However, the encoder can only be evaluated on one axis at a time.

The **_enableAxisInterface()** or **_disableAxisInterface()** commands and the sensor=<sensorNumber> function parameter are used to activate and deactivate the encoder interface during operation and the specification of the sensor. Because the sensor function parameter is encoded as bits, it is also possible to activate/deactivate several encoders concurrently. If the bit is not set, the encoder interface status remains unchanged.

The Activated/Deactivated encoder interface status is displayed in the **sensorData.sensorData[i].input** system variable.

Example:

sensor=5 Activation/deactivation of the encoder interface 1 and 3

Also refer to the parameters description in the reference lists.

These commands are not available for enabling/disabling the actuator/sensor interface on the hydraulic axis. For information on the hydraulic functionality, see Access to the same final controlling element from multiple axes (Page 303).

See also

Setting a non-exclusive drive assignment (as of V4.1 SP1) (Page 59)

3.5.11 Diagnostic features

The measuring system increments are displayed as 32-bit information in the **sensorData.incrementalPosition** system variable.

It can first be verified whether the change in this variable over one encoder revolution with allowance for the measuring gear corresponds to the increments of the encoder. If this is not the case, the configuration of the axis and the drive must be modified accordingly (e.g. measuring gear, (drive) parameters for setting the resolution: number of increments per revolution, fine resolution, etc.).

In addition, the path (angle) change/revolution (**sensorData.position**) can be checked. If this check reveals problems, the axis configuration must be checked (e.g., leadscrew pitch, gear ratios, etc.).

3.6 Input limits, technological limiting functions

System-side input limits

All parameters that can be entered have a lower and an upper limit derived either from the variable format range or from limits set by the system for each parameter. See the TP Cam System Variables Reference List.

Internally there are also limits for the position specifications resulting from the accuracy of the data format of the variables. When these limits are violated, an error message is output on the axis: **Internal traversing range limit reached**.

3.7 Setting for axis and encoder mechanics

3.7.1 Overview of setting options for axis and encoder mechanics

The following options are available on the drive side of a real axis:

- Consideration of load gear
- Consideration of spindle pitch on linear axis
- Inversion of manipulated variable/drive direction

The following options are available on the encoder side of a real axis:

- Setting of encoder mounting method
- Consideration of measuring gear
- Setting of distance per revolution

- Settings for load compensation
- Consideration of the count direction (inversion of actual position value)

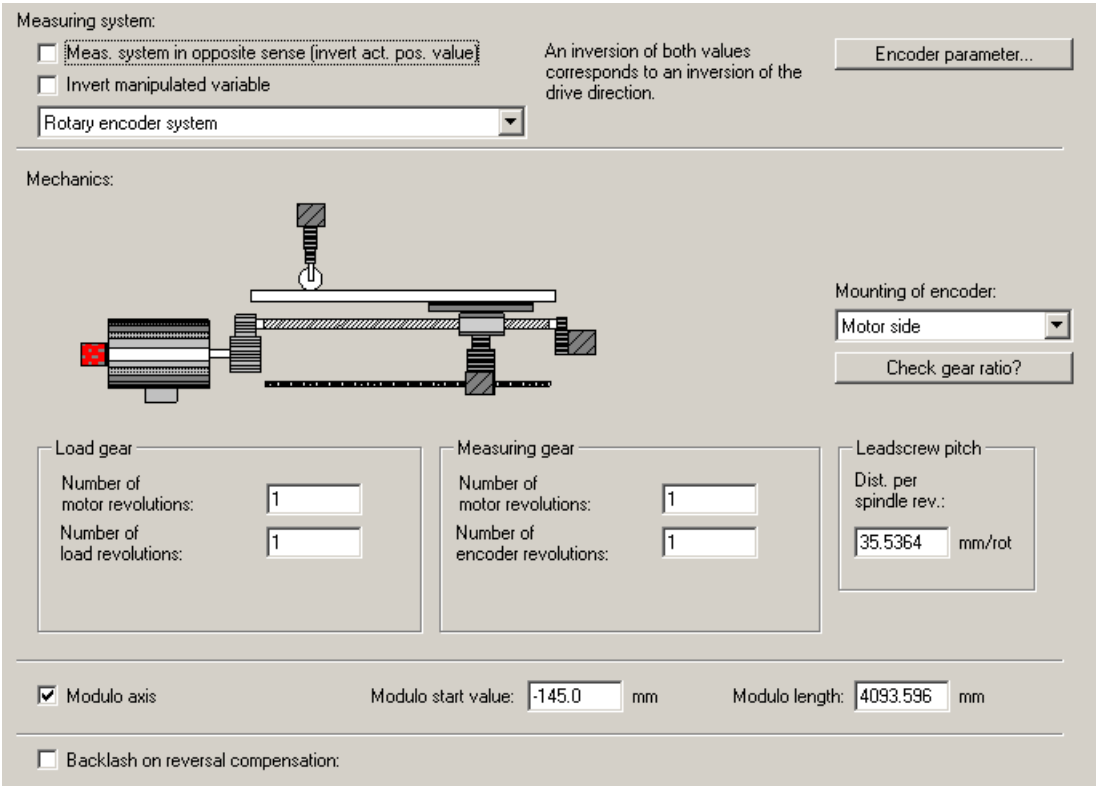


Figure 3-21 Overview of encoder/linear axis mechanics

The following options are available on the final control element side for the hydraulic functionality:

- Consideration of output characteristic
- Inversion of manipulated variables

The available encoders in SIMOTION can be used for the hydraulic functionality. They can be configured on the **Encoder configurations** tab in the **Configuration** dialog or via the expert list. (see SIMOTION SCOUT online help)

3.7.2 Inversion of actual position value

You can invert the count direction of an incremental or absolute encoder by selecting the **Meas. system in opposite sense** check box in the **Encoder/linear axis mechanics** screen form. This changes the count direction to match the technological view.

Once the Invert actual position value setting has been modified, the encoder should be adjusted once again, this time with the new absolute encoder offset which has been set accordingly, just as the home position offset needs to be redefined for an incremental encoder.

Special features of the absolute encoder

With an n-bit encoder (n-bit data length), the incremental position lies between 0 and 2^n-1. The position is then derived independently of the resolution, e.g. from 0 to 100 mm.

If the sign is changed by inverting the actual position value (measuring system is in opposite direction or encoder is mounted backwards), the incremental position lies between 2^n-1 and 0. The position range is inverted accordingly, e.g. from 100 to 0 mm. If the actual position in this case is 15 mm without sign change when the encoder is switched on, then the position with sign change is 85 mm.

Example:
A linear scale with a grid line spacing of 0.005 mm and a measuring range of 30 mm or 6,000 increments is specified. The maximum value for the incremental position can be displayed on a computer as a data value with a width of 13 bits (2^{13} = 8,192).
Therefore, the encoder's maximum position is in theory 40.96 mm (8,192 * 0.005 mm). If the axis is offset in the positive direction, the displayed encoder position increases its value, moving from 0.0 up toward 40.96 mm, and the incremental position increases from 0 up toward 8,191 increments. An incremental position of 1,000 increments, for example, corresponds to an axis position of 5.0 mm. By contrast, changes to the axis position in the positive direction on an encoder where the invert actual position value function has been activated will cause the displayed encoder position to decrease its value, moving from 46.96 down toward 0.0 mm. The inverted encoder position is derived from the difference between the encoder's theoretical maximum position and the encoder position determined by the current incremental position. This means that the encoder incremental position of 1,000 increments given in the example results in a position of 35.96 mm (40.96 mm - 5.0 mm = 35.96 mm).

3.7.3 Boundary conditions for mechanical settings for modulo axes (long-term stability / long-term accuracy)

For modulo axes, SIMOTION checks whether the long-term accuracy can be guaranteed. Testing takes place on initial download, each restart of the CPU and each restart of the technology objects, if relevant data have changed. The long-term accuracy ensures the following:

- That the axis position displayed by the controller always conforms to the real position, i.e. the axis position can be displayed uniquely from the accumulated encoder increments.
- That the calculated position in the controller does not differ from the real position. Without long-term accuracy, differences result, e.g. because of rounding errors or calculations with finite accuracy.

This enables positions to be approached exactly, even after any number of modulo overflows.

Note on non-modulo axes:

Non-modulo axes are “always accurate” because of an endless traversing range, even if the convertibility of gear ratios is not stable over the long term.

If the maximum internal position ($9x10^{12}$ at 1000 increments/unit) is reached and the convertibility of gear ratios is not stable over the long term, an alarm 50011 occurs (range limit of the incremental actual value exceeded) and the axis must be re-homed.

The long-term accuracy/stability of modulo axes is dependent on the following:

- Internal resolution, increments/position [units/mm]
  (can be set at **Configuration > Axis type > Configure units**)
- Encoder resolution: [incr/rev]
- Encoder fine resolution
- Spindle pitch [mm/rev] (for linear modulo axes)
- Measuring gearbox
- Load gearbox

Checking long-term accuracy as of V4.2

The long-term accuracy is only checked during runtime, as a result of the encoder data adaptation.

Long-term accuracy means that the technological position can be determined from the accumulated encoder increments (without rounding error) and is therefore always accurate.

With long-term accuracy, the encoder position is converted to the mechanical position via the numerator/denominator factor of the gear ratio. If there is no long-term accuracy, an LREAL factor is used as substitute for the conversion. Alarm 50024 (long-term stability of the actual values is not guaranteed) is then output. NONE is the default setting for the local response.

When using the LREAL factor to take into account the gear ratio, modulo axes are accurate as long as the total traversing distance of the axes without modular conversion does not exceed the maximum traversing range of the axis (representation range $9x10^{12}$ at 1000 increments/unit).

If the maximum value of the representation range is exceeded, this results in inaccuracy of the technological position. The unavoidable rounding error in the last available position of the LREAL factor applies in this case. However, for the generally used settings, this is only reached after years.

Determining the maximum traversing time of an endlessly traversing axis

The maximum traversing time can be determined based on the following estimate. The maximum position and the traversing velocity are decisive:

Maximum position = $9x10^{12}$mm at an internal resolution (default) of 1000 internal increments/ position unit

Traversing time = maximum position / velocity

Table 3-24 Example of a linear modulo axis

Velocity:	20 m/min = $2x10^4$ mm/min
Maximum position:	$9x10^{12}$mm
Traversing time = $9x10^{12}$ mm / ($2x10^4$ mm/min) = $4.5x10^8$ min = 856.16 years	

At a higher set internal computational accuracy, the traversing time is correspondingly lower.

If the determined traversing time does not meet your requirements, you can use the alternative procedure presented below or identify the required settings for long-term stability in accordance with the section “Checking long-term accuracy < SIMOTION V4.2”.

Alternative procedure

Reset the position by homing if the maximum traversing time does not meet your requirements.

Incremental encoder

Home the incremental encoder again before the position value $9x10^{12}$ is reached.

Absolute encoder (as of SIMOTION V4.3 SP1 HF17)

Perform absolute encoder adjustment with preset current known position via the **_homing()** function using homingMode=SET_OFFSET_OF_ABSOLUTE_ENCODER_BY_POSITION.

This resets the position values to the current measurement range in a cyclic absolute encoder. The reset may be performed via a user program e.g. after a

Power ON.

Checking long-term accuracy < V4.2

The long-term accuracy of a modulo axis is a precondition for operating the axis.

If long-term accuracy cannot be guaranteed as a result of the mechanical settings, then one of the following error messages is output in the engineering system during the consistency check.

- Configured gear ratio cannot be represented
- Configured modulo length cannot be represented

A test of the long-term accuracy is also performed on the target device during runtime. If the long-term stability is not ensured due to the configuration, the following alarm is issued: **Alarm 20006 Configuration error, reason 3041**.

The reason for this error is the unsuitable selection of values in the configuration data. The Excel list available on the Siemens Service & Support (https://support.industry.siemens.com/cs/ww/en/view/40786053) website assists you when checking and calculating gearbox parameters.

See also

Encoder interface using the PROFIdrive message frame (Page 68)

Overview of setting options for axis and encoder mechanics (Page 79)

3.8 Defaults

The axis has system variables that can be set to default values for the axis. The most important default values, e.g., for the dynamic response of the axis, are compiled in the Dynamic Response screen form. These default values will be used if you use **Default** or **USER DEFAULT** in your programming.

Table 3-25 Other default values that can be set via the expert list

Default values for	System variable
Clamping values	**userDefaultClamping**
Dynamic values	**userDefaultDynamics**
Switchover to force control	**userDefaultForceControl**
Force/pressure limiting	**userDefaultForceLimiting**
Homing	**userDefaultHoming**
Positioning	**userDefaultPositioning**
Hydraulics	**userDefaultQFaxis**
Torque limiting	**userDefaultTorqueLimiting**

See also

Default settings for dynamic response parameters (Page 158)

3.9 Homing

3.9.1 Overview of homing

On the positioning axis, inputs and displays concerning the position refer to the coordinate system of the axis.

The coordinate system of the axis must be aligned with the real, physical position of the axis.

Absolute encoder

For absolute encoders, inclusion of the absolute encoder offset must be activated **once**.

For more information, refer to Section States that require re-adjustment of the absolute encoder (Page 95).

Incremental encoder

For **incremental encoders**, synchronization is performed by homing when the home position coordinates, or the home position coordinate taking into account the home position offset, are set in active homing to a defined mechanical position of the axis. This defined mechanical position of the axis is signaled to the controller via the encoder zero mark of the measuring system or via the external zero mark.

For incremental encoders, if you want to establish a direct reference to the position, you must synchronize the actual value system of the axis after every activation.

Note

Traversing commands with **relative position specification** can always be executed. The axis configuration can be set to determine whether traversing commands with **absolute position specification** can also be executed on a non-homed axis. Configuration data: **referencingNecessary**.

See also

Homing (Page 327)

3.9.2 Terminology

Home position

When the axis has been synchronized and the reference point offset has been applied, the axis is at the reference point and has the value specified in the reference point coordinate.

Set the home position or the home position coordinate in the **userDefaultHoming.homePosition** system variable. The setting is axis-specific.

The **homingCommand.state** system variable displays the homing status on the axis.

Synchronization point

In the synchronization point, the actual value of the axis due to an external or internal event is set to the value **home position coordinate minus home position offset**.

Reference point offset

The offset between the home position and the synchronization point is only effective during active homing. It is applied after synchronization of the axis by means of the homing command. In modulo axes, the **home position offset** is always applied with the **Shortest_Way** direction setting. (Exception: As of 4.1 SP1, the direction can be specified with the "Home in one direction only" setting.)

Specify the home position offset in the **TypeOfAxis.NumberOfEncoders.Encoder_1.IncHomingEncoder.proceedShiftPos** configuration data element.

Reference mark

A homing mark is a hardware signal that is used for homing.

- Encoder zero mark
  The encoder zero mark of an incremental encoder is used as a reference mark.
- External zero mark
  An external signal (zero mark substitute) is used as the reference mark.

Homing output cam

The homing output cam outputs an enable signal for the actual reference signal (encoder zero mark or external zero mark). The drive reduces the velocity on the basis of the switching edge returned by the homing output cam and waits for the next incoming homing signal in order to perform the homing operation.

Note

Device-specific properties

The homing output cam is connected to any control input for homing with homing output cam and encoder zero mark.

For more information about the device-specific boundary conditions and additional parameter setting requirements, see the supplementary information for drives and the product manuals.

3.9.3 Homing types

Open-loop control supports the following types of homing:

- **Active homing**
  A special traversing motion is carried out for this type of homing. The following homing modes can be selected via the configuration:
  - Homing with homing output cam and encoder zero mark
  - Homing with external zero mark only
  - Homing with encoder zero mark only.
- **Passive homing/flying homing**
  This type of homing occurs during motion that was not initiated by a homing command. The following homing modes can be selected via the configuration:
  - Homing with homing output cam and encoder zero mark
  - Homing with external zero mark only
  - Homing with encoder zero mark only.
- **Direct homing / Set home position**
  The axis position is set without a traversing motion having taken place.
- **Relative direct homing**
  The actual position value of the axis is shifted by a specified offset without a traversing motion having taken place.
- **Absolute encoder homing/absolute encoder adjustment**
  The zero point of the absolute encoder is adjusted.

  Note

  As of V4.2, the inputs for homing output cams and for the status of the external zero mark and of the reversing cam can be interconnected to SINAMICS inputs symbolically instead of via the intermediary of a logical address.

3.9.4 Active homing

During active homing, the homing operation is performed according to the specified mode by means of a motion initiated by the homing command.

A home position offset is in effect and is applied after synchronization with the reference mark. Once the home position offset has been applied, the axis position has the value specified in the home position coordinate.

In active homing, the velocity specifications for the homing approach velocity, the homing reduced velocity, and the homing entry velocity are effective.

The axis has **synchronized** or **homed** status once the reference mark is detected.

Note

The direction of approach to the reference cam or reference mark must be specified in the user program. As of V4.1, support is provided for using hardware limit switches as reversing cams.

Active homing with reference cam and encoder zero mark mode

The homing command causes the axis to move toward the reference cam. After the axis has detected and left the reference cam, the axis moves to the next encoder zero mark of the position measuring system. You can specify in a configuration data element whether the encoder zero mark is in the positive or negative traversing direction. The axis is synchronized with the first encoder zero mark detected after the reference cam. Then, the home position offset is traveled at the homing entry velocity. The axis position now has the value defined in the home position coordinate.

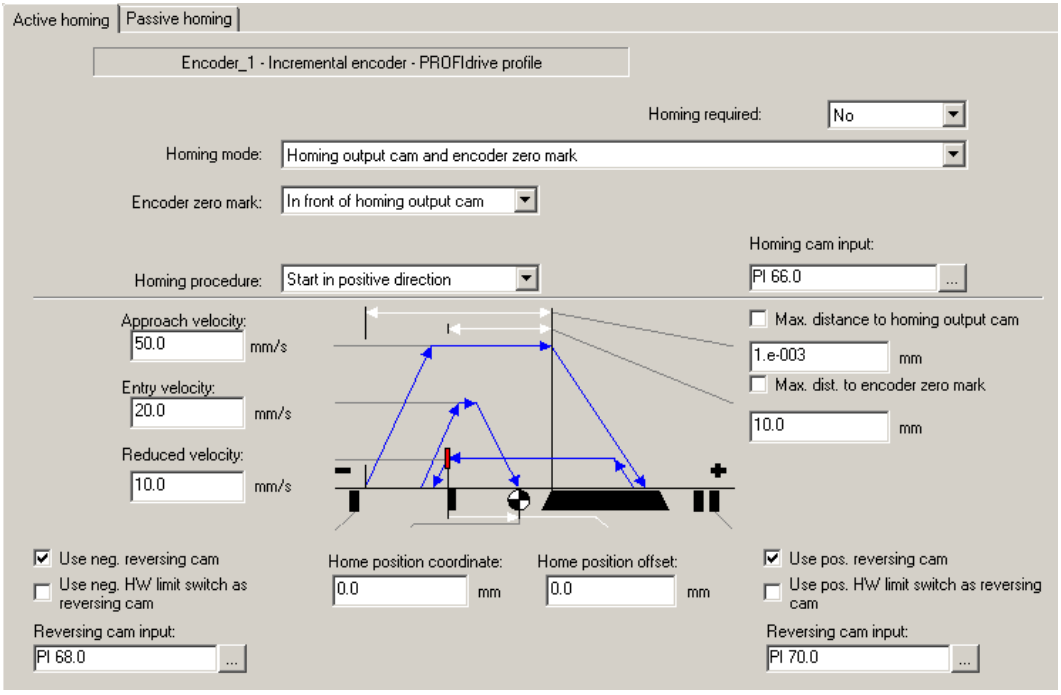


Figure 3-22 Parameters for active homing with reference cam and encoder zero mark

The sequence can be divided into three phases:

- **Phase 1**
  Travel to the reference cam
  The axis moves to the reference cam at the **approach velocity**. The approach direction is parameterized.
- **Phase 2**
  Synchronization with encoder zero mark
  The axis moves to the encoder zero mark of the incremental position encoder at the **reduced velocity**. The position of the encoder zero mark relative to the reference cam can be parameterized. Depending on this position, the movement is either continued in the same direction or reversed. The controller is synchronized to the first detected encoder zero mark which is detected according to the configuration setting. When the encoder zero mark is detected, the axis is considered to be synchronized, and the axis position is set to the value specified in the home position coordinate, minus the home position offset.
- **Phase 3**
  Travel to the home position
  When the encoder zero mark is detected, the axis moves to the home position at the **entry velocity**.

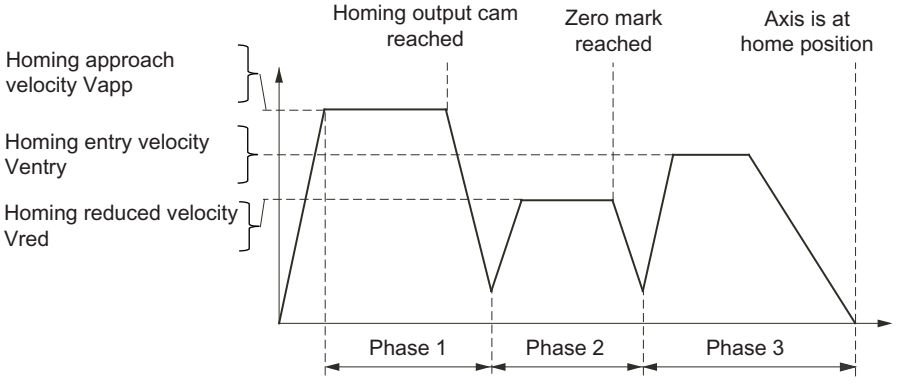


Figure 3-23 Homing with incremental measuring systems

Hardware limit switch as reference cam (as of V4.1 SP1)

The hardware limit switches can also be defined as a reference cam. The axis is homed to the first encoder zero mark after the direction is reversed as a consequence of approaching the hardware limit switch. The axis cannot continue to travel in the direction of the hardware limit switch.

This corresponds to homing with reference cam and encoder zero mark, whereby the encoder zero mark lies in front of the reference cam.

Alternatively, a left and right hardware limit switch can be used (positive or negative approach direction). In this case, the hardware limit switch is deactivated during homing.

Active homing with external zero mark only

Here, the homing command initiates motion toward the external zero mark. Once the axis reaches the configured edge of the external zero mark, the home position offset is applied at the homing entry velocity. The axis position now has the value defined in the home position coordinate.

Note

For homing to the external zero mark of axes with a digital drive link via PROFIdrive, the external zero mark must be configured as a digital input on the drive (configuration of zero mark substitute for SIMODRIVE 611U, SINAMICS). The external zero mark signal must be connected at the same location where the encoder is measured, e.g. on the drive, on the ADI4/IM174, or on the inputs of the C2xx intended for the external zero mark.

SINAMICS

With SINAMICS, a positive direction of motion is synchronized to a positive trigger edge and a negative direction of motion is synchronized to a negative trigger edge, i.e. on the left side of the external zero point signal in each case.

By inverting the signal (setting option on the drive: SINAMICS parameter via P490), synchronization is also possible on the right side of the external zero point signal.

In SINAMICS, the homing to the encoder zero mark or the external zero mark is set in P495.

To be able to detect for drives coupled using PROFIdrive that the drive is positioned at the external zero point, the state of the external zero point on the axis must be made available. Because the corresponding status bit is not provided in the PROFIdrive telegram of the drive DO, an additional input bit containing the status of the external zero mark can be configured on the axis in the **incHomingEncoder.stateDriveExternalZeroMark** configuration data element as of V4.1 SP1. For drives linked with SINAMICS, the corresponding status bit in PZD2 of telegram 390 or 391 of the CU is transferred to the controller.

If symbolic assignment is activated, the status of the external zero mark can be assigned to the corresponding onboard terminal on the control unit directly using the Assign dialog.

Refer to the SINAMICS documentation for the assignment of digital inputs of the drive to the status bits of the PZD2.

The system will detect whether the axis is already positioned at the external zero point. Travel from the external zero point is made opposite to the approach direction. This is followed by normal homing.

SIMODRIVE 611U

With SIMODRIVE 611U, only falling edges are permitted in the positive traversing direction, and only rising edges are permitted in the negative traversing direction, meaning that the same output cam side is always used.

Inverting is not possible with SIMODRIVE 611U. The edges can be inverted by using initiators of the inverted type or via an application (inaccurate).

MASTERDRIVES

With MASTERDRIVES Motion Control, only rising edges are permitted.

OnBoard C2x0

With OnBoard, only rising edges are permitted.

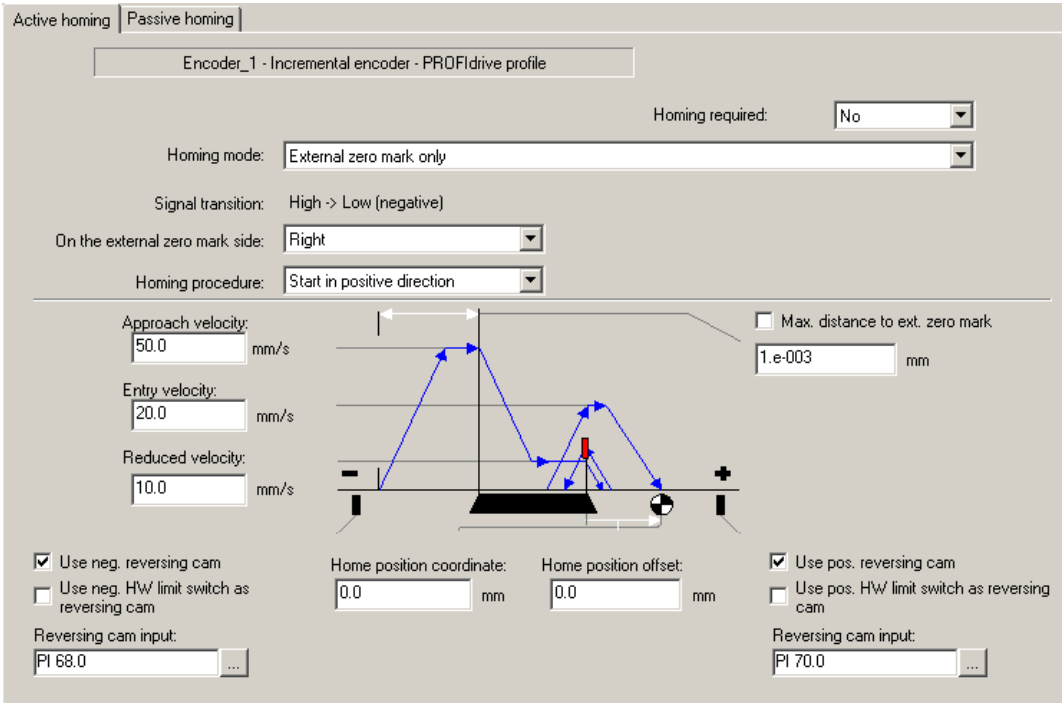


Figure 3-24 Parameters for active homing with external zero mark only

Note

- Only homing with a reference cam and an encoder zero mark should be considered for a resolver with a pole pair number > 1.

- With a digital link to SINAMICS S120, the settings for homing are read out from the drive on request. Changes are not written back to the drive.

Active homing with encoder zero mark only (without reference cam)

Homing without a reference cam is used, for example, in axes for which the encoder has only one encoder zero mark in the entire axis traversing range. This homing command causes the axis to travel to the encoder zero mark. After the encoder zero mark is detected, the axis approaches the shifted reference point at homing velocity. The axis position now has the value defined in the home position coordinate.

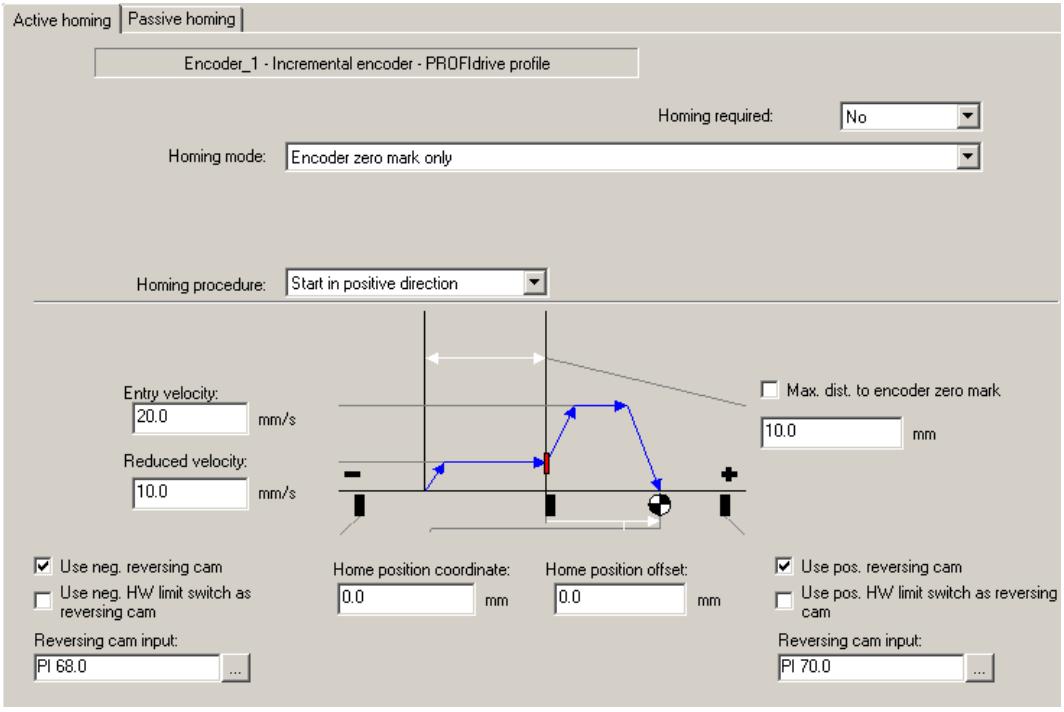


Figure 3-25 Parameters for active homing with encoder zero mark only

Note

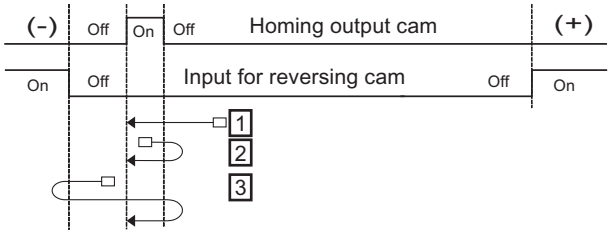
Only homing with a reference cam and an encoder zero mark should be considered for a resolver with a pole pair number > 1.

Reversing cams for homing (as of V4.1 SP1)

Reversing cams, which are effective only during active homing, are used to reverse the direction of the home position travel when approaching a reversing cam.

The reversing cams are configured as two additional input signals. The left reversing cam and the right reversing cam can be configured and activated separately. The reversing cams are defined once on the axis. They cannot be defined specifically for an encoder.

The following figure illustrates homing sequences based on the starting point position with respect to the reference cam.



All examples are programmed with the Left-hand approach direction (negative)

1 Starting point in front of reference cam

2 Axis is on reference cam
This is detected by the system, and the axis travels against the approach direction away from the reference cam. This is followed by normal homing.

3 Axis is behind the reference cam on the left
A normal homing operation with approach direction left is started. The axis reverses direction at the reversing cam and travels against the approach direction until it is over the reference cam. This is followed by normal homing.

Figure 3-26 Reversing cams for homing

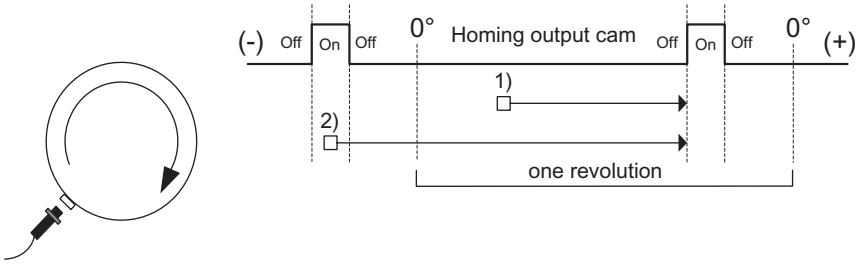
The hardware limit switches can also be defined as reversing cams. In this case, the hardware limit switch is deactivated during homing.

The reversing cams are defined via the configuration data elements **typeOfAxis.homing.reverseCamPositive** and **typeOfAxis.homing.reverseCamNegative**.

Homing in one direction only (as of V4.1 SP1)

As of V4.0, you can specify in the **typeOfAxis.homing.direction** configuration data element that a direction reversal during homing should be suppressed and the specified traversing direction should always be maintained. This setting is effective only during active homing.

The figure below shows an application example featuring a rotary axis that is not permitted to reverse direction.



All examples are programmed with the Right-hand approach direction

1) Starting point in front of reference cam
Reference cam is found in the modulo area.

2) Axis is on reference cam
This is detected by the system. The axis travels in the programmed approach direction to the next reference cam, even if the modulo area is overtraveled.

Figure 3-27 Example of homing in one direction only

3.9.5 Passive homing/on-the-fly homing

In passive homing, the homing is performed according to the mode setting using a motion not initiated by the homing command. Passive homing is possible in position-controlled mode in association with motion commands. A home position offset is not applied. The axis has synchronized or homed status once it has detected the homing mark. Specifications for the homing approach velocity, reduced velocity, and entry velocity are not applied.

Default homing mode (DEFAULT_PASSIVE)

With this setting, the homing mode is defined by the system based on the encoder type:

- With incremental sin/cos encoders, TTL encoders, or resolvers, homing is executed to an encoder zero mark.
- With Endat encoders defined as incremental encoders, homing is executed to an external zero mark.

Passive homing with homing output cam and encoder zero mark mode

Once the homing output cam has been detected, the next encoder zero mark takes effect according to the synchronization specified in the configuration. Synchronization is performed with the first encoder zero mark detected after the homing output cam. When the encoder zero mark is detected, the axis is set to the value specified in the home position coordinate. The axis is then given Synchronized and Homed status.

Passive homing with external zero mark only

Once the external zero mark has been detected, synchronization takes place according to the configuration. When the external zero mark is detected, the axis is set to the value specified in the home position coordinate. The axis is then in synchronized and homed status.

Passive homing with encoder zero mark only

Homing without a homing cam cam is used, for example, in axes for which the encoder has only one homing mark in the entire axis traversing range. When the homing mark is detected, synchronization takes place according to the configuration. Once the homing mark has been detected, the axis is set to the value specified in the home position coordinate and is in synchronized or homed status.

Note

Use the "homing output cam and encoder zero point" setting for homing for measuring systems with more than one encoder zero point in the traversing distance of the axis. This ensures that travel is made to an exact, reproducible homing position.

As an alternative, you can use the "only external zero point" setting. The external signal means, however, that the attainable homing position is less accurate.

3.9.6 Direct homing/setting the home position

The current position of the axis is set to the value specified in the home position coordinate. A home position offset is not applied. A traversing motion is not carried out. When the command is executed, the axis is in synchronized or homed status.

The setting of parameters for axis homing is irrelevant here. The home position coordinate is specified in the command.

Direct homing is not possible if the axis is traversed in speed control. Switch to position control for the homing.

See also

Positioning (Page 330)

Synchronization/homing with incremental encoders (Page 379)

3.9.7 Relative direct homing/relative setting of home position (V3.2 and higher)

The current position of the axis is offset by the value specified in the home position coordinate, so in this case the home position coordinate should be regarded as an offset.

This homing method can also be used during operation (while traversing).

The setting of parameters for axis homing is irrelevant here. The home position coordinate is specified in the command.

3.9.8 States that require a new homing procedure for incremental encoders

With incremental encoders, the status is reset to not homed in the following cases:

- Error in the sensor system / encoder failure
- New homing command
- Power off
- Termination of the communication to the drive/encoder
- SCOUT is downloaded with selection of the *Initialization of all retentive data* setting
- Changes requiring a download or restart are made to the axis configuration
- Technology object restart is performed on the Axis technology object
- When active homing is initiated

3.9.9 Absolute encoder homing / absolute encoder adjustment

The current axis position is set to equal the encoder value + absolute encoder offset using the **_homing()** command in ENABLE_OFFSET_OF_ABSOLUTE_ENCODER homing mode.

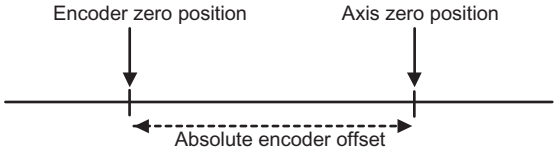


Figure 3-28 The axis zero position is the encoder zero position plus the absolute encoder offset

The axis must be in position-controlled mode while the absolute encoder is adjusted.

The absolute encoder offset (as of V3.2) may be set as an additive or absolute value.

This absolute encoder offset is stored in the NVRAM and remains in effect until the next absolute encoder adjustment. This function must therefore be executed once when the control is commissioned.

The offset value and its calculation are set during **configuration**.

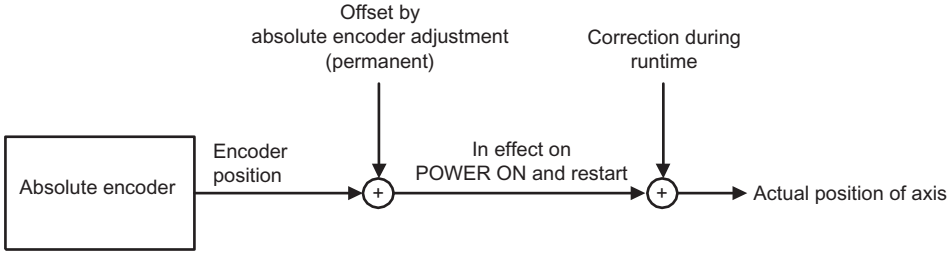


Figure 3-29 Incorporating the absolute encoder offset

A value may be set as a total offset using the **absHomingEncoder.setOffsetOfAbsoluteEncoder** and **absshift** configuration data.

Setting an additive offset

Setting **absHomingEncoder.setOffsetOfAbsoluteEncoder**=RELATIVE (default behavior):

- Actual axis value = actual encoder value + (previous offset already in effect + absshift)
- (new) offset = previous offset + absshift

absHomingEncoder.absshift is added to the existing absolute encoder offset whenever the **_homing()** function is called.

Setting an absolute offset (as of V3.2)

Setting **absHomingEncoder.setOffsetOfAbsoluteEncoder**=ABSOLUTE (as of V3.2):

absHomingEncoder.absshift is set as the absolute encoder offset whenever the **_homing()** function is called.

- Actual axis value = actual encoder value + **absshift**

Example of ABSOLUTE offset where actual encoder position value = 100.000:

absshift = 5.000	
_homing() command called for the first time	⑧ Position = 105.000
_homing() command called for the second time	⑧ Position = 105.000
absshift = 7.000	
_homing() command called for the third time	⑧ Position = 107.000

Setting the axis to a predefined position (as of V4.1 SP1)

When the function parameter **homingMode**=SET_OFFSET_OF_ABSOLUTE_ENCODER_BY_POSITION is set on the **_homing()** command, the value in the *homePosition* parameter is set as the current position. The resulting absolute encoder offset is calculated with this value, indicated in the **absoluteEncoder[n].totalOffsetValue** system variable, and stored as a retain variable in the system.

The value in the **absHomingEncoder.absshift** configuration data element is not changed.

Displaying the offset

The offset can be read out. (as of V3.1)

The total offset is indicated in the **absoluteEncoder[x].totalOffsetValue** system variable, while the activation status of the total offset is indicated in the **absoluteEncoder[x].activationState** variable.

In addition, the status indicates whether the **_homing()** function where **homingMode**= ENABLE_OFFSET_OF_ABSOLUTE_ENCODER was executed at least once after the project was downloaded.

Absolute encoder adjustment

Perform the following steps to adjust the absolute encoder:

1. Disable the software limit switches, because you cannot adjust the absolute encoder while these are active.
2. Perform the absolute encoder adjustment:
   - **_homing()** where **homingMode**= ENABLE_OFFSET_OF_ABSOLUTE_ENCODER
     The value of the **absHomingEncoder.absshift** configuration data element is included when the command is called once. (**absHomingEncoder.absshift** is a configuration data element that can be modified online, i.e. any changes take effect immediately.)
     or
   - **_homing()** where **homingMode**=SET_OFFSET_OF_ABSOLUTE_ENCODER_BY_POSITION (as of V4.1 SP1)
     Move the axis to the desired reference position and call the command once.
     This sets the value in the *homePosition* parameter as the current position. The resulting absolute encoder offset is calculated using this value, indicated in the **absoluteEncoder[n].totalOffsetValue** system variable, and stored as a retain variable in the system.
     The value in the **absHomingEncoder.absshift** configuration data element is not changed.
3. Enable the software limit switches again (if necessary).

Note that the absolute encoder adjustment only acts as an offset to the absolute encoder value. The offset from the absolute encoder adjustment and the value of the absolute encoder are determining factors for the position after a power off or restart. During operation, the current actual position is also affected by the modulo settings for the axis and by position setting or position correction tasks.

States that require re-adjustment of the absolute encoder

- Once a new project has been downloaded to the control, the stored offset is no longer available.
  If the control already contains a project before the new project is downloaded, and if the technology object name is not changed, the stored offset is retained (as of V4.1 SP1). This behavior also applies for an upgrade, i.e. it is not version-dependent.
- After power is cycled off and on, the offset is deleted unless the project was saved to the ROM.
- After a memory reset.

See also

Encoder for position (Page 62)

3.9.10 Homing mark monitoring

If the homing mark is not reached within the defined travel path, an alarm is triggered. In homing with homing output cam and homing mark, the path is monitored only after the axis leaves the homing output cam.

If reversing cams are present, monitoring is applied again when the direction reverses as a result of the revering cams.

When homing mark monitoring is enabled, both active and passive homing procedures are monitored.

3.9.11 Homing output monitoring

If the homing output cam is not reached within the defined travel path, an alarm is triggered.

If reversing cams are present, monitoring is applied again when the direction reverses as a result of the revering cams.

When homing output cam monitoring is enabled, both active and passive homing procedures are monitored.

3.9.12 Displaying actual value change during homing (V4.0 and higher)

A change of the actual value during homing is displayed in the **homingCommand.positionDifference** system variable.

3.9.13 Traversing with a non-homed axis

With the **referencingNecessary** configuration data element, you define whether absolute positions can be used with a non-homed axis.

Settings:

- referencingNecessary = NO
  - Relative and absolute motions are possible.
  - The software limit switches are monitored with the setting **swlimit.state** = YES.
- referencingNecessary = YES
  With a non-homed axis:
  - Only relative motion is possible
  - The software limit switches are not monitored even with the setting **swlimit.state** = YES.

3.9.14 Correcting the actual position/set position without homing

Position correction can also be used to manipulate the actual values and setpoints of individual coordinates (basic coordinates, superimposed coordinates).

The homing status of the axis (Homed/Not homed) does not change.

See also

Resetting the set and actual positions (Page 339)

3.9.15 Differential position measurement (V3.2 and higher)

The axis position is not measured directly; it is determined as the difference between two encoders/actual values. The system adopts this differential position as the actual position and uses it as an absolute value.

The encoders for the individual positions may be either absolute encoders or incremental encoders.

Homing of the differential position is not permitted.

The offset relative to the differential position can be modified online using the **positionDifferenceMeasurement.Offset** configuration data element.

The differential position and the position sensors are used as encoders.

Differential position value =
Position (numberEncoderA) - Position (numberEncoderB)

Proceed as follows to specify the setting via the expert list:

- Configure at least two position sensors on the axis.
- In the expert list, increase the value of **TypeOfAxis.NumberOfEncoders.NumberOfEncoders** by 1.
- Set the other sensor as the "differential position sensor" via the **TypeOfAxis.NumberOfEncoders.Encoder_n.EncoderType** configuration data element with SENSOR_POSITION_DIFFERENCE_MEASUREMENT.
- The sensors whose values are used and the individual factors are set in the elements of the **TypeOfAxis.NumberOfEncoders.Encoder_n.PositionDifferenceMeasurement** structure.

See also

Using the expert list for an axis (Page 229)

3.10 Monitoring/limiting functions

3.10.1 Overview of monitoring/limiting functions (block diagram)

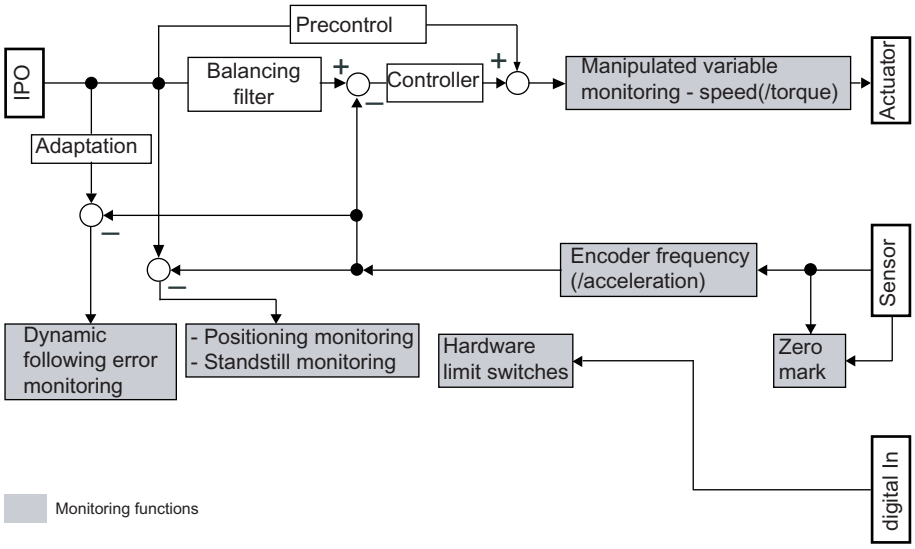


Figure 3-30 Block diagram of positioning axis monitoring functions

3.10.2 Dynamic monitoring of following errors

The following error monitoring on the position-controlled axis is performed on the basis of the velocity-controlled following error limit.

The maximum permissible following error is dependent on the set velocity.

For velocities less than a specified lower velocity, the maximum permissible following error is constant and set in **typeOfAxis.NumberOfDataSets.DataSet_N.DynamicFollowing.minPositionTolerance**. The lower velocity value for the characteristic of the maximum following error is set in **typeOfAxis.NumberOfDataSets.DataSet_N.DynamicFollowing.minVelocity**.

Above this lower velocity, the maximum permissible following error is proportional to the set velocity up to the maximum permissible velocity.

The maximum permissible following error for the maximum velocity of the axis is set in **typeOfAxis.NumberOfDataSets.DataSet_N.DynamicFollowing.maxPositionTolerance**. The gradient factor of the characteristic for the maximum following error is calculated from these settings.

The **dynamicFollowing.warningLimit** configuration data can be used to specify a threshold as percentage value based on the maximum permissible following error of the associated set velocity which, when exceeded, causes the issuance of a warning.

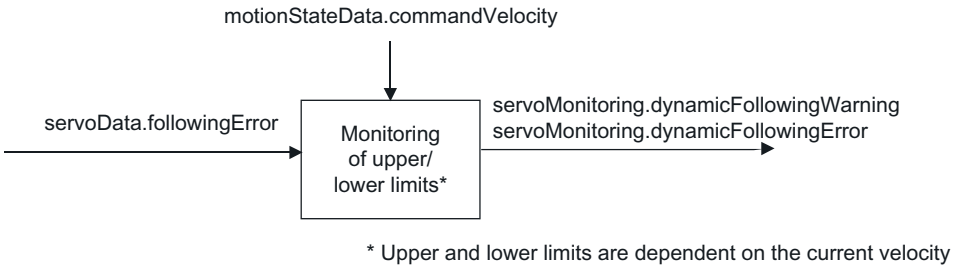


Figure 3-31 Dynamic monitoring of following errors

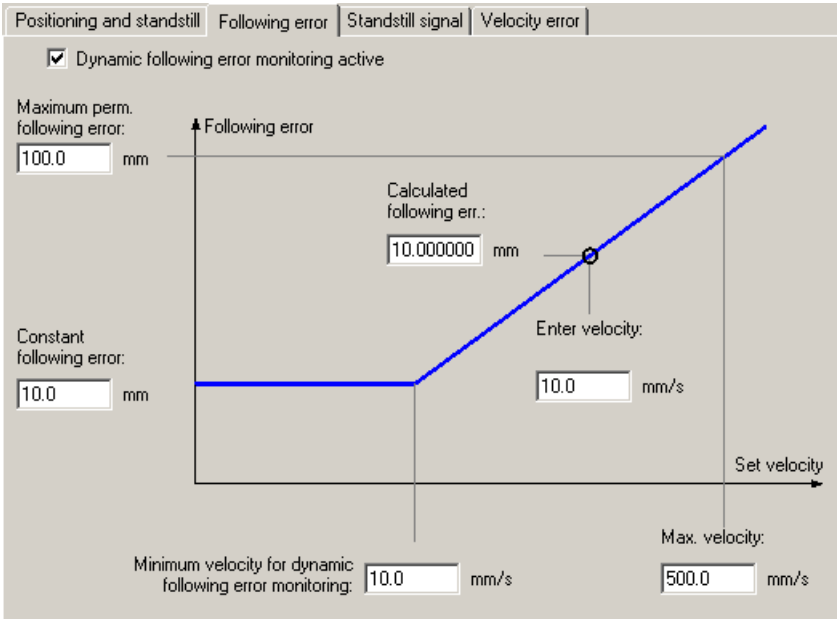


Figure 3-32 Function and parameters for following error monitoring

System variable on the TO with current position lag limit

The **Permissible following error** block is provided to monitor the stability of the axis. This contains the permissible dynamic following error as system variable (**servoData.permissibleFollowingError**).

Following error determination without DSC

The following error is determined from the difference between the non-symmetrical setpoint prior to inclusion of the dynamic response adjustment and the actual value present in the control.

Therefore, the transfer times of the setpoint to the drive and the actual value to the control are included in the following error.

Following error determination with DSC

The following error is determined from the difference between the non-symmetrical setpoint delayed by the transfer times (T_i+T_o+T_{dp}+T_{servo}) + **systemDeadTimeData.additionalTime** prior to inclusion of the dynamic response adjustment and the actual value present in the control.

The transfer times of the setpoint to the drive and the actual value to the control are calculated from the following error in order to refer to the following error present on the position controller in the drive.

3.10.3 Positioning and standstill monitoring

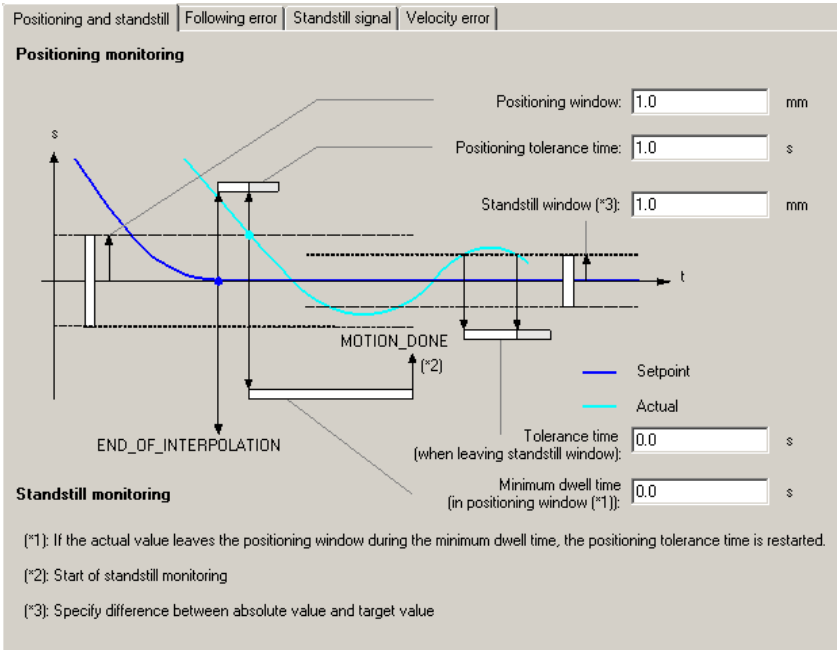


Figure 3-33 Function and parameters for positioning monitoring

Positioning monitoring

The behavior of the actual position at the end of the setpoint interpolation is monitored. (Positioning monitoring)

This positional monitoring does not distinguish whether the setpoint interpolation is ended as a result of reaching the target position from the setpoint side or due to a position-controlled stop during the motion performed by the interpolator (e.g. with a _stop() command). This monitoring is referred to as positioning monitoring, although the position does not have to be the same as the target position. The tolerance window specified for this monitoring is called the positioning window.

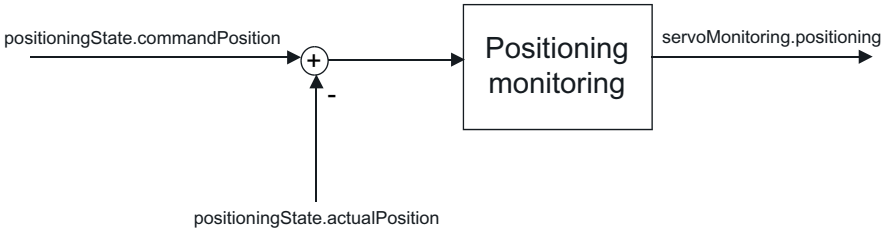


Figure 3-34 Positioning monitoring

Sequence:

- When the setpoint interpolation is complete, the **typeOfAxis.positionMonitoring.posWinTolTime** monitoring time is started.
- If, before this time expires, the actual value reaches a definable window **typeOfAxis.positionMonitoring.tolerance** around the existing position at the end of the setpoint interpolation, monitoring of the minimum delay time **typeOfAxis.positionMonitoring.posWinTolDelayTime** is initiated. This window is indicated as a deviation in **typeOfAxis.standStillMonitoring.stillStandTolerance**, meaning that half of the window width is set.
  If the actual value does not reach the window within the monitoring time **typeOfAxis.positionMonitoring.posWinTolTime**, alarm 50106 (positioning monitoring) is issued.
- If the actual value leaves this window again during the minimum dwell time **typeOfAxis.positionMonitoring.posWinTolDelayTime**, the monitoring time **typeOfAxis.positionMonitoring.posWinTolTime** is restarted; each time it re-enters this window, the minimum delay time **typeOfAxis.positionMonitoring.posWinTolDelayTime** is restarted.
- If the actual value remains in this window for the minimum delay time **typeOfAxis.positionMonitoring.posWinTolDelayTime**, the MOTION_DONE status is set in the system variable **motionStateData.motionCommand** and the standstill monitoring is started.

In addition, the individual monitoring phases are displayed in **servoMonitoring.positioningState** (as of V4.1 SP1):

- ACTUAL_VALUE_OUT_OF_POSITIONING_WINDOW = setpoint interpolation is complete; actual value has not yet reached the positioning window.
- ACTUAL_VALUE_INSIDE_POSITIONING_WINDOW = actual value is inside positioning window; standstill monitoring is not yet started.
  The ACTUAL_VALUE_OUT_OF_POSITIONING_WINDOW is displayed when the actual value has left the positioning window again.
- STANDSTILL_MONITORING_ACTIVE = standstill monitoring is active; positioning to the position reached at the end of the setpoint interpolation has taken place.
- INACTIVE

Note

Positional monitoring (e.g. following error monitoring or positioning monitoring) are disabled when the pressure control is enabled or via the pressure limiting or torque limiting command.

Standstill (zero-speed) monitoring

Standstill monitoring is defined by the standstill window and the tolerance time during which the standstill window may be exited without alarm 50107 (standstill monitoring) being triggered.

Standstill monitoring starts when the minimum dwell time in the positioning window has elapsed.

The standstill window is indicated as a deviation in **typeOfAxis.standStillMonitoring.stillStandTolerance**, meaning that half of the window width is set.

The status of the standstill monitoring is displayed in **servoMonitoring.stillstand**. Standstill monitoring is not available on the speed-controlled axis.

3.10.4 Standstill signal

The standstill signal **motionStateData.stillstandVelocity** is ACTIVE when the current velocity is less than a configured velocity threshold for at least the duration of the delay time.

Note

Below this velocity, the motion is stopped in response to **_stopEmergency()** at zero setpoint without a preconfigured deceleration ramp.

In speed-controlled motion, the criterion for completing the command is standstill monitoring. The completion of commands for position-controlled motions is described in **Positioning and standstill monitoring**.

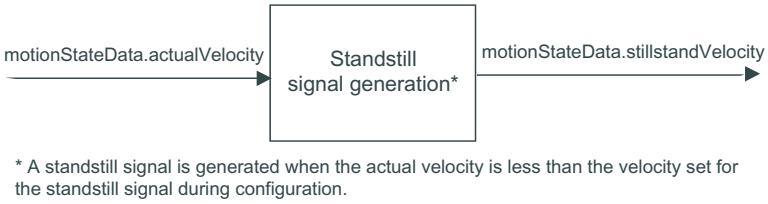


Figure 3-35 Standstill signal generation

The standstill signal is available on the positioning axis and the speed-controlled axis.

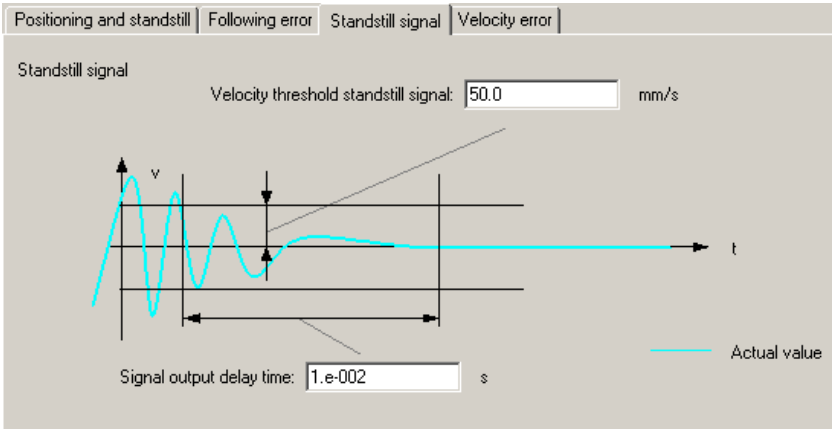


Figure 3-36 Function and adjustable parameters for the standstill signal

Standstill signal monitoring

motionStateData.ActualVelocity shows the actual velocity after the smoothing filter. The axis velocity before the smoothing filter is used for the standstill signal monitoring.

The unfiltered velocity used for the standstill window is not displayed specifically.

See also

Positioning and standstill monitoring (Page 102)

3.10.5 Manipulated variable monitoring

The maximum manipulated variables are limited for monitoring of the assigned speed limits. If the manipulated variables exceed the maximum value configured in **MaxSpeed**, alarm "50005 speed setpoint monitoring" is triggered.

With the help of increase monitoring of the manipulated variable, the maximum possible acceleration and thus the maximum torque are monitored. If the monitoring function is switched on and the acceleration set in the configuration data **MaxAcceleration** is exceeded, alarm 50003 (limit of speed set acceleration is active) is output, and the parameter **SetPointChangeRate** is set to LIMIT_EXCEEDED.

3.10.6 Manipulated variable limiting (backstop) (V3.1 and higher)

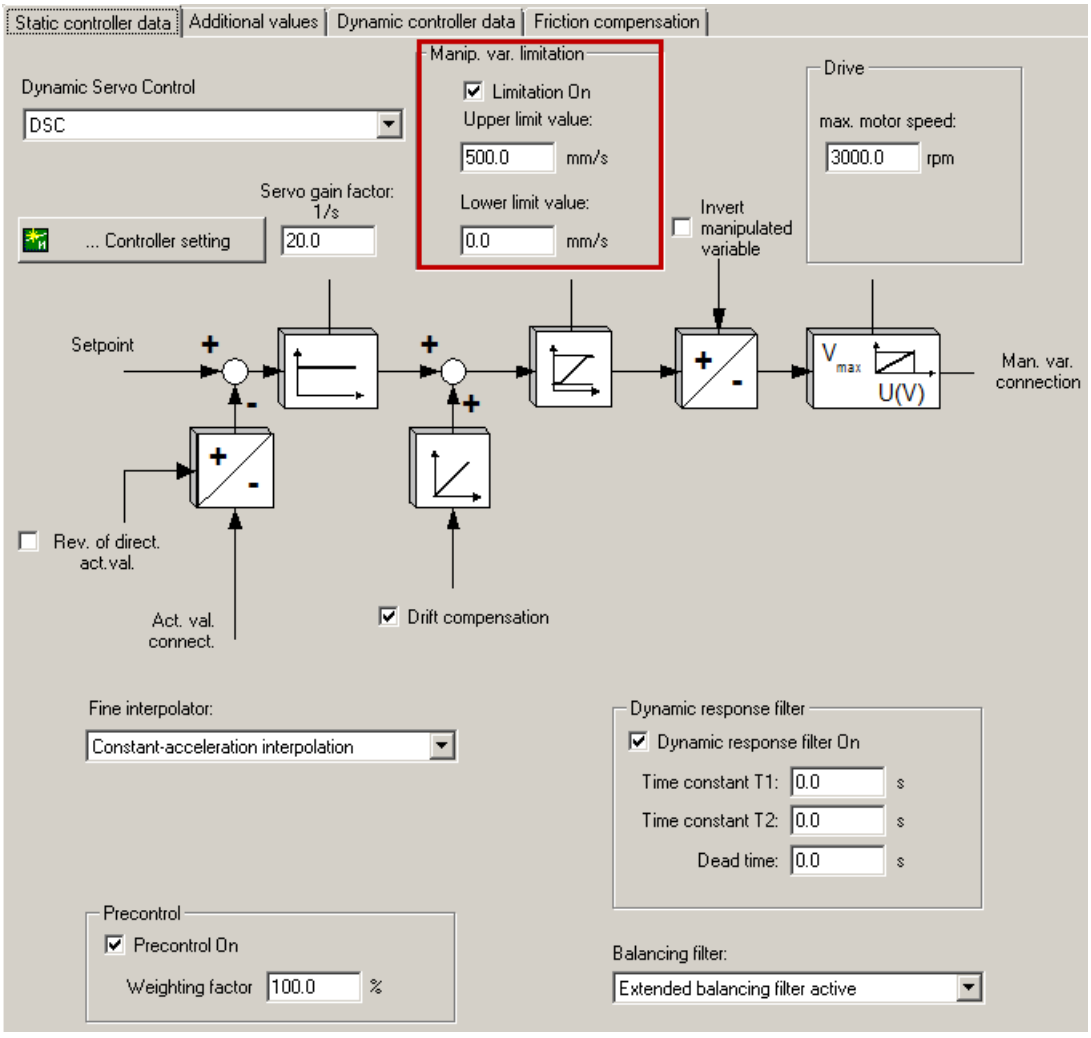


Figure 3-37 Manipulated variable limitation in static controller data

Manipulated variable limitation on an electrical positioning axis in the servo performs absolute limiting of the position value to an upper and a lower limit value.

This limitation is applied prior to inversion.

The values can be modified online (with immediate effect).

If the upper value is positive and the lower value is 0, the axis can, for example, be traversed in the positive direction only.

The manipulated variable limitation can be configured and activated by means of the **speedLimitation** configuration parameter. The zero velocity must lie within the permissible range to enable the axis standstill.

Only the setpoint is limited; reversing of the drive must also be prevented.

Note

When the **Dynamic Servo Control** (position controller in the drive) function is active, backstop (limiting of the manipulated variable for the drive) is only effective for the precontrol.

Therefore, when DSC is active, the backstop must be generated in the drive.

3.10.7 Hardware limit monitoring

Traversing range limits are monitored by means of digital inputs and limit switches. Hardware limit switches are **always normally closed switches** and should **always be active outside** the permissible traversing range.

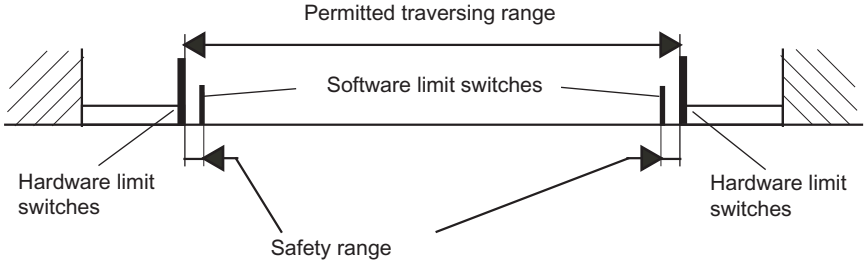


Figure 3-38 Monitoring of the permissible traversing range by limit switches

Approaching the limit switch triggers **technology alarm 50007**.

This state can be remedied in two ways:

- **Manual retraction (without drive)**
  The axis is returned manually to the permissible traversing range. The system variables remain in LIMIT_EXCEEDED state until the axis leaves the limit switch. Only then can the alarm be acknowledged.
- **Retraction with drive**
  The technology alarm must be acknowledged. **Warning 50009: Limit switch overtraveled** is retained. The axis can only travel in the direction of retraction as long as the limit switch is active. Travel in the opposite direction triggers a technology alarm and cancels the enables. When the axis has left the limit switch, the system variables **sensorMonitoring.hwLimitSwitchMinus** and **sensorMonitoring.hwLimitSwitchPlus** assume the **O_K** state, and the **Limit switch overtraveled** warning can now be acknowledged.

When a limit switch is approached, its position is stored. The limit switch is not considered to have been left until the axis moves back from the position.

NOTICE
Switching the controller off after the limit switch has been overtraveled
Once the limit switch has been overtraveled, the controller must not be switched off to avoid a conflict between the polarity monitoring of the limit switches and the overtravel monitoring of the limit switches in the direction of the permissible area.
If the controller is switched off, the information of the limit switch polarity is lost. The axis must then be moved into the permissible range by the user.
When the controller is **switched on**, the axis must be positioned within the permissible traversing range.
If the hardware limit switches are configured as reversing cams or reference cams for homing, these hardware limit switches can be overtraveled during homing, even if they are activated.

If the limit switch is overtraveled and the configuration reloaded, internal states are lost. Reloading without loss of the approach information is only possible within the valid range. Exception: Deactivation of limit monitoring after a polarity reversal error.

A cable break can only be reset with **Power On** or by a one-time deactivation of the function.

The hardware end position interface can be activated and deactivated on the axis using the **_enableAxisInterface()** and **_disableAxisInterface()** commands.

The Activated/Deactivated hardware end position interface status is displayed in the **sensorMonitoring.hwLimitSwitchInput** system variable.

3.10.8 Software limit monitoring

Software limit switches can be specified and monitored as soon as the actual values are valid. Monitoring is enabled/disabled using the **swLimit.state** system variable. The software limit switch positions are defined in the **swLimit** system variable. Software limit switches should be inside of the hardware limit switches.

If an axis motion, synchronous motion, or path motion is canceled by approaching the software limit switch, because continuing the motion would violate the software limit switch, movement to the software limit switch is performed at the braking ramp's maximum dynamic values. Technological alarm 40106, "Approach SW limit switch", is only output when the software limit switch is reached.

If the software limit switches are also to be monitored with non-homed axes, **homing.referencingNecessary** = NO must be set in the configuration data element.

With the setting **Homing required**, active limit monitoring is only effective if the axis is homed. If the setting **Homing required** is not selected, active limit monitoring is always effective.

The software limit position monitoring is effective also for isochronous movement and path movement. For a violation of the software limit position, the isochronous movement or path movement will be terminated and travel made to the software limit switch with maximum deceleration and maximum jerk.

If valid actual values are present for a speed-controlled procedure, these serve as limits for the software end position monitoring.

Monitoring the software limit switch at motion start (V3.2 and higher)

The controller checks for a violation of the limits before the motion starts. If the software limit switch is exceeded, the axis is limited to the software limit switch position and alarm 40105 is triggered.

If alarm 40105 is active, no more movement commands will be accepted and the limit switch will be approached at the programmed dynamic values. The error only needs to be acknowledged when, for example, using an application program, to stop before the limit switch or to travel in the reverse direction.

If, for example, a second motion is superimposed and acts in the opposite direction, it is possible for the software limit monitoring to signal an alarm when the first motion is activated, even though the software limit switch is not reached.

The **monitoringAtMotionStart** configuration data element can be used to enable/disable the check when the motion starts.

Note

The cyclical software limit monitoring check is always performed during the motion.

Passing a software limit switch

If the axis is positioned after the software limit switch due to reconfiguring of the software limit switch, for example, or the axis has passed the software limit switch in speed-controlled mode, alarm 40107 *Software limit switch passed* is output. The axis can be traversed out of the limit switch area in position-controlled mode depending on the local error response set in alarm 40107, e.g. when setting FEEDBACK_EMERGENCY_STOP.

For retraction in speed-controlled mode, a tolerance window with the configuration data **relieveWindow** can be defined. The tolerance window always refers to the current actual value. This means that jittering of the actual value will not cause the software limit switch to respond again during retraction. Traversing within the tolerance window is possible in both directions.

3.10.9 Encoder limit frequency monitoring

The system monitors the limit frequency of the encoders for compliance. The monitoring function triggers an alarm. It has no effect on axis motion.

3.10.10 Velocity error monitoring

An encoder must be connected and configured to monitor the velocity error (setpoint minus actual value) on the axis. The controlled system is simulated using a PT1 model. This model is supplied with the setpoint as the input value, and the difference of the output value is compared with the real actual value curve. The time constant for the PT1 model is set during axis configuration in the **dynamicData.velocityTimeConstant** or **dynamicQFData.velocityTimeConstant** configuration data element. Velocity error monitoring is of relevance for the drive axis and for the position axis in SPEED_CONTROLLED mode.

3.10.11 Measuring system differential/slip monitoring

The system can monitor a measuring system difference / slip between two encoders on the axis to a specified maximum value. The monitoring is activated with **_enableMonitoringOfEncoderDifference()** and deactivated with **_disableMonitoringOfEncoderDifference()**.

The maximum value is specified in the **_enableMonitoringOfEncoderDifference()** command in the *maximalEncoderDifference* function parameter and transferred with the execution of the command in the **sensorMonitoring.maximalSensorDifference** system variable or the existing value in the **sensorMonitoring.maximalSensorDifference** system variable is used optionally by setting the *maximalEncoderDifferenceType* function parameter.

If the maximum measuring system difference specified in **sensorMonitoring.maximalSensorDifference** for activated monitoring is exceeded, the alarm 20009, "the permissible difference between encode (/1/%d) and (/2/%d) has been exceeded" will be generated and signaled with LIMIT_EXCEEDED in the **sensorMonitoring.slippageTolerance** system variable.

Before activating the function (and if applicable while the function is active), check whether the two encoders involved in the measuring system difference are valid. The check can be performed using the system variables **sensormonitoring.cyclicinterface**=ACTIVE and **sensordata.state**=VALID, for example.
The absolute encoder is restarted, for example, on switchover by means of data set changeover and may then supply other values over the course of a small number of cycle clocks, so that in this case measuring system difference monitoring may respond.

3.11 Positioning axis with position control

3.11.1 Overview of positioning axis with position control

This figure shows the block diagram of the position axis with position control.

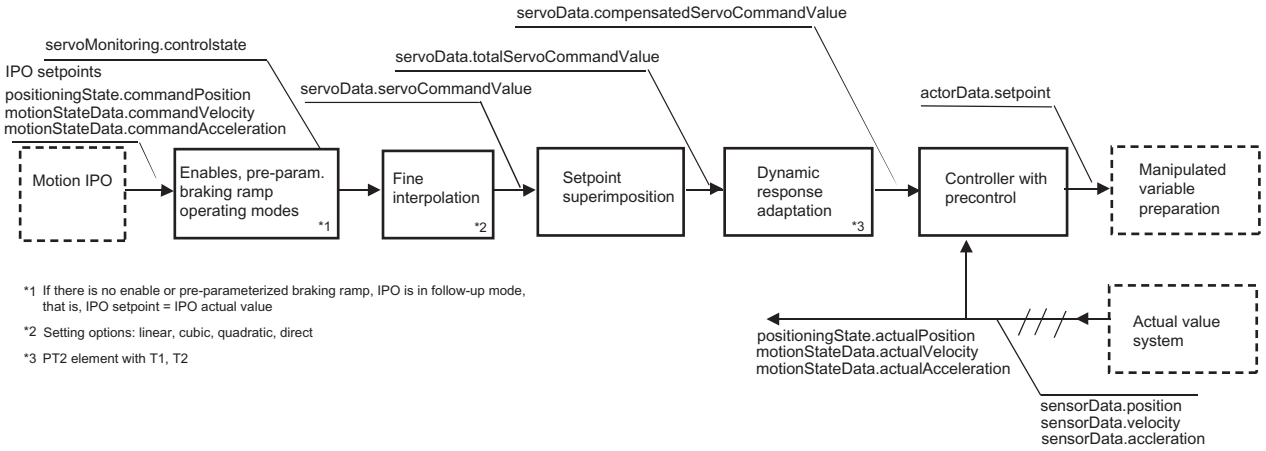


Figure 3-39 Overview of position axis with position control

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

You can call signal flow diagrams under the Axis object in the project navigator. The signal flow diagram illustrates the signal path and enables major items of configuration data and important system variables to be parameterized using the signal flow.

3.11.2 Position control

When position control is **active**, controllers, monitoring, and compensation are active. There are modes in which monitoring mechanisms are deactivated. They include torque limitation and the position-related monitoring functions for pressure limitation, for example.

All compensation functions can be activated/deactivated.

Encoder systems, actual value calculation, and monitoring are active on the actual value side when position control is not activated. Compensation functions are not taken into account.

The **servoMonitoring.controlState** system variable indicates whether the position controller is active.

SIMOTION provides a P-controller with or without precontrol and a PID controller.

Controller with precontrol

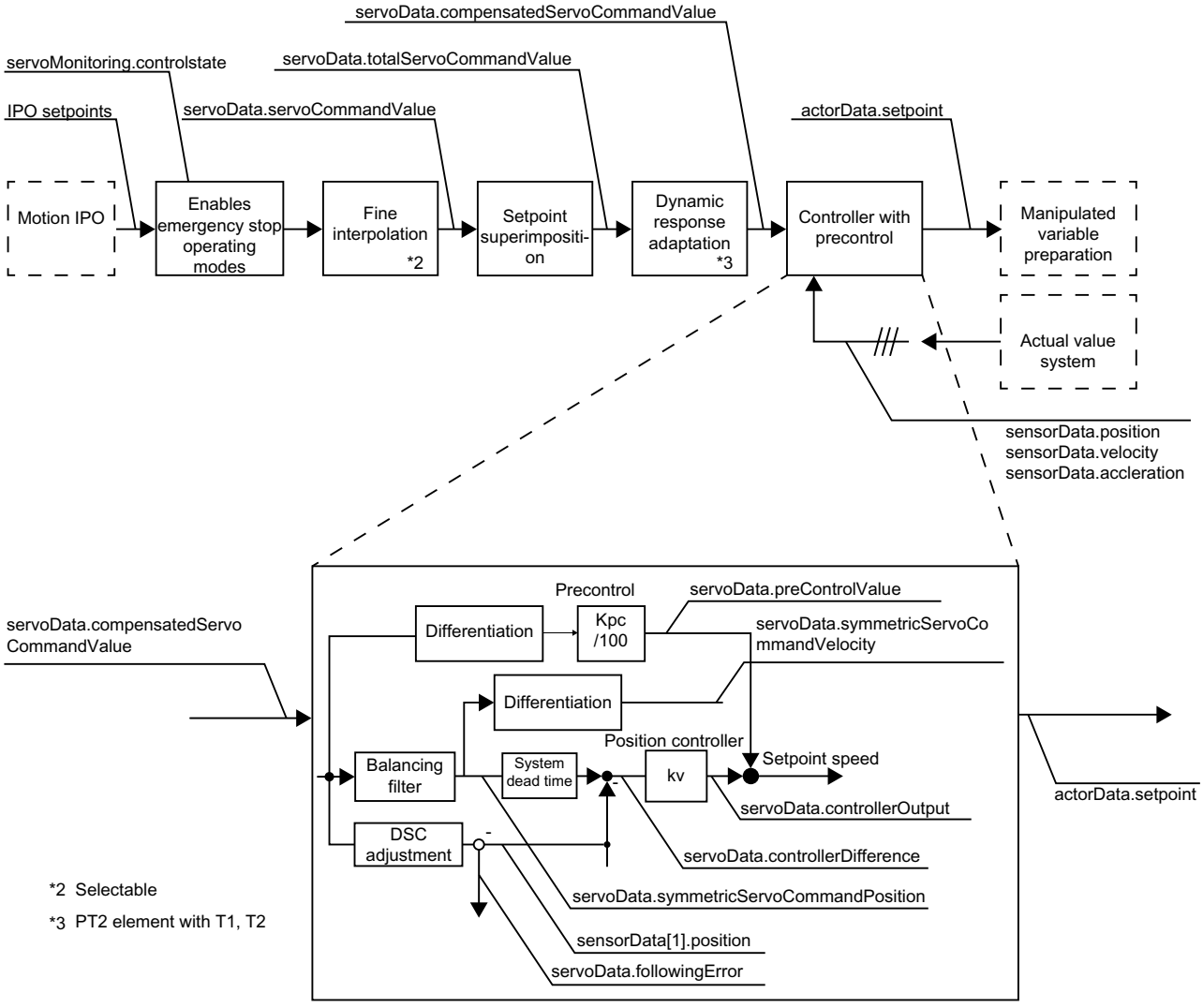


Figure 3-40 Proportional-action controller with precontrol

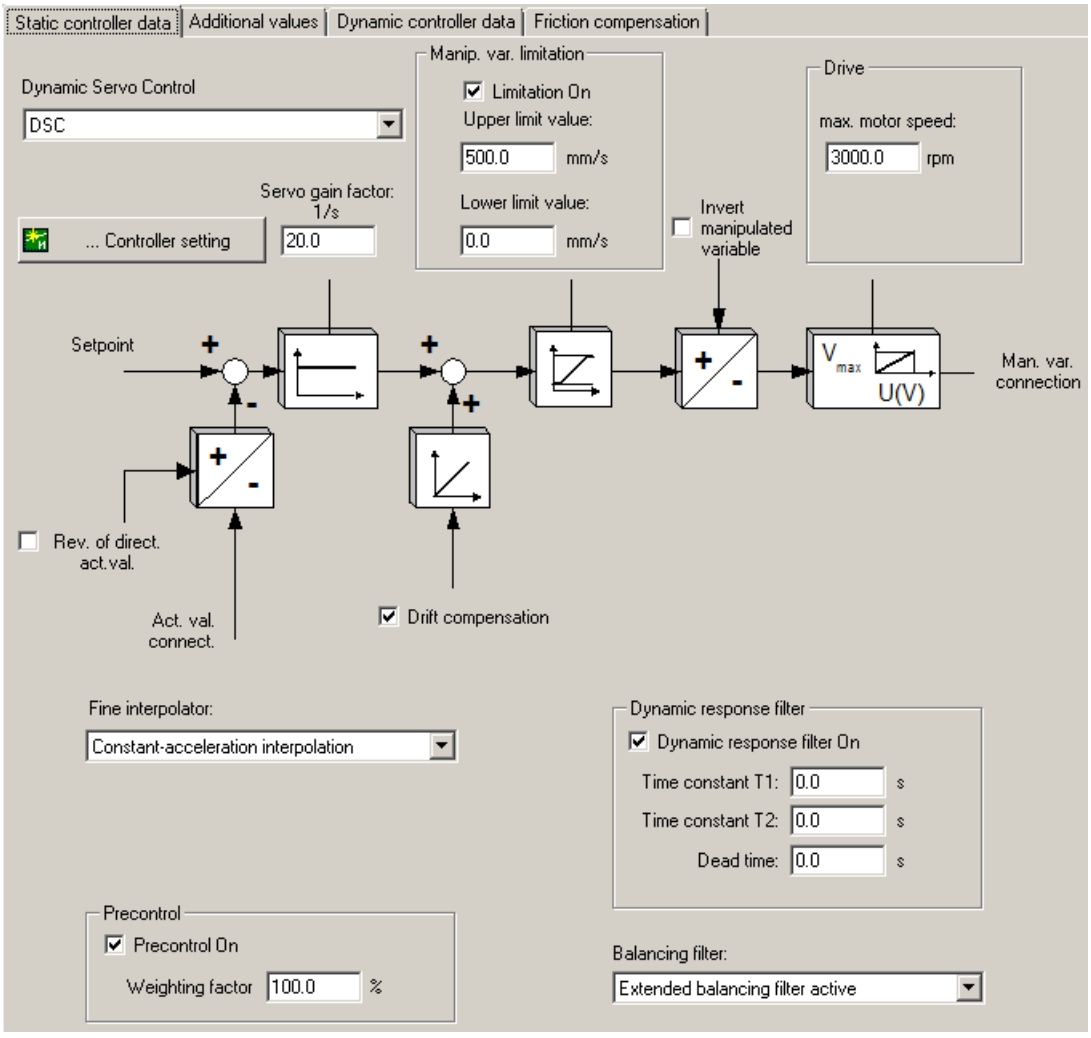


Figure 3-41 Static controller data

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

Axis wizard default settings

- P-action controller with precontrol for the electric axis
- DSC to improve the control quality (higher servo gain values) for digitally coupled drives (only with P controllers with precontrol)
- PID controller for hydraulic axes (allows you to switch the actual value directly to the D component).

See also Position control for setting a positioning axis with hydraulic functionality (Page 293).

Position loop gain (servo gain factor)

For a P controller with or without precontrol, the gain of the P controller can be specified via the servo gain factor. The manipulated variable/traversing velocity component of the closed-loop control is generated from the control error via the servo gain factor Kv.

The mathematical (proportional) relationship is:
Servo gain factor Kv = (traversing velocity v / control error Δs) [1/s]

The servo gain factor Kv affects the following characteristic values:

- Positioning accuracy and holding control
- Uniformity of motion
- Positioning time

The better the axis design (high degree of stiffness), the higher the achievable servo gain factor Kv, and therefore the better the axis parameters are from a technological perspective (lower following error and higher dynamic response).

Precontrol

The velocity precontrol can be used to limit the velocity-related following error during position control. It can also help to achieve faster positioning.

With precontrol, the velocity setpoint is also switched additively to the position controller output. This additional setpoint can be weighed with a factor.

Balancing filter for feedforward control

The balancing filter is a simplified model of the speed control loop. It is used to prevent the position controller from overriding the manipulated velocity variable during the acceleration and deceleration phases. This is accomplished by delaying the position setpoint of the position controller by the balancing time with reference to the velocity precontrol.

Function of the balancing filter

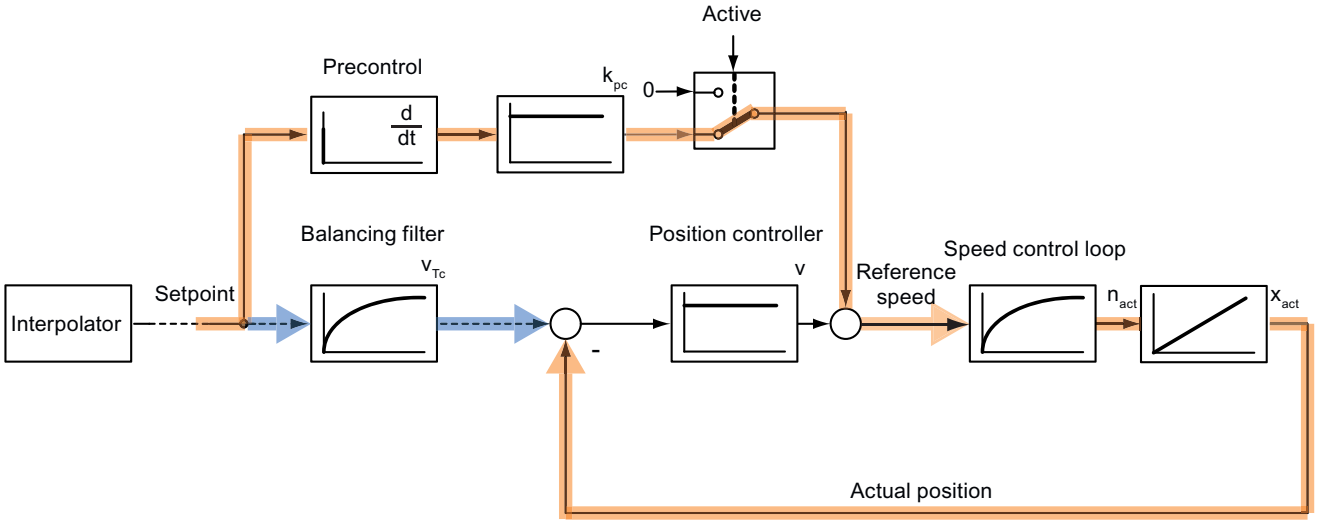


Figure 3-42 Balancing filter - Example of an electric axis without DSC

The speed is precontrolled; to this end, the position setpoint is adopted from the reference variable calculation directly or with differentiation applied, and the set velocity is defined on the speed controller directly. The position controller then simply has to compensate for any position errors which may be present in spite of the precontrol.

The position error, therefore, is generated from the position setpoint, delayed by the speed control loop equivalent time, and the existing actual position value.

Settings and configuration data

The setting can be made using the *Closed-loop control* dialog for *Dynamic controller data* (activate expert mode).

When precontrol is active, allowance can be made in the balancing filter for the response of the speed control loop prior to formation of the system deviation from the position setpoint and actual position.

- With the **balanceFilterMode=OFF** configuration setting, the balancing filter is switched off.
- With the **balanceFilterMode=MODE_1** configuration setting, a PT1 filter is used as the balancing filter.
  - For electric axis without DSC:
    The following must be set as the time constant for the balancing filter in **dynamicData.velocityTimeConstant**: $T_{Balancing\ filter}=T_{Speed\ control\ loop\ equivalent\ time}+T_i+T_o+T_{dp}+T_{servo}$
    The transmission dead times and process response (equivalent time of speed control loop) must be taken into account in the time constant.
    systemDeadTimeData.additionalTime is not included additively by the system in the balancing filter.
  - For electric axis with DSC:
    The following must be set as the time constant for the balancing filter in **dynamicData.velocityTimeConstant**: $T_{Balancing\ filter}=T_{Speed\ control\ loop\ equivalent\ time}$, because the transmission dead times for the position controller in the drive are not present in the position control loop.
    systemDeadTimeData.additionalTime is not included additively by the system in the balancing filter.
  - For the hydraulic axis:
    The following must be set as the time constant for the balancing filter in **dynamicQFData.qOutputvelocityTimeConstant**: $T_{Balancing\ filter}=T_{qOutputEquivalent\ time}$ + Communication times according to the setpoint output/actual value interface.
    ($T_{qOutputEquivalent\ time}$ as equivalent time for the QOutput process response)
- With the **balanceFilterMode=MODE_2** configuration setting (as of V3.1), the equivalent time of the speed control loop, the dead time determined by the system for the drive, and a dead time that can be input in additive increments by the user are taken into account in the balancing filter.
  Maximum time constant = 16 x T_{servo}
  - For electric axis without DSC:
    The following must be set as the time constant for the balancing filter in **dynamicData.velocityTimeConstant**: $T_{Balancing\ filter}=T_{Speed\ control\ loop\ equivalent\ time}$ $T_i+T_o+T_{dp}+T_{servo}$ is taken into account by the system.
    The value set in **systemDeadTimeData.additionalTime** is included additively by the system in the balancing filter.
  - For electric axis with DSC:
    The following must be set as the time constant for the balancing filter in **dynamicData.velocityTimeConstant**: $T_{Balancing\ filter}=T_{Speed\ control\ loop\ equivalent\ time}$, because the transmission dead times for the position controller in the drive are not present in the position control loop.
    systemDeadTimeData.additionalTime is not included additively by the system in the balancing filter.

– For the hydraulic axis:
The following must be set as the time constant for the balancing filter in **dynamicQFData.qOutputvelocityTimeConstant**: $T_{Balancing\ filter}=T_{qOutputEquivalent\ time}$ + Communication times according to the setpoint output/actual value interface ($T_{qOutputEquivalent\ time}$ as equivalent time for the QOutput process response)

The setting **balanceFilterMode=MODE_2** on the balancing filter is recommended.

An overswing is apparent should the **dynamicData.velocityTimeConstant** value be too small. If the value is too large, the axis has limited dynamic behavior and creeps to the end position.

System deviation

The control deviation is the difference between the balanced setpoint and the actual value and is displayed in the **servoData.ControllerDifference** system variable.

Note

As of runtime version V4.1 SP1, the control deviation present on the position controller in the drive is displayed for DSC. This is calculated in the control using a model (see the Control Structure figure in the next chapter).

With versions prior to V4.1 SP1, the control deviation present in the control is displayed with DSC.

The **servoData.controlDifferenceCommandPosition** system variable (as of V4.4) displays the setpoint present in the drive. With reference to the current actual value, this setpoint is relevant to control deviation generation. The signal runtime and output times are taken into account.

Calculating the servoData.controlDifferenceCommandPosition system variable without balancing filter (configuration setting balanceFilterMode=OFF):

- Without DSC
servoData.controlDifferenceCommandPosition
= **servoData.symmetricServoCommandPosition**
- With DSC
servoData.controlDifferenceCommandPosition
= **servoData.symmetricServoCommandPosition** delayed by $T_i+T_o+T_{dp}+T_{servo}$

Calculating the servoData.controlDifferenceCommandPosition system variable with balancing filter (configuration setting balanceFilterMode<>OFF):

- Without DSC
servoData.controlDifferenceCommandPosition
= **servoData.symmetricServoCommandPosition** delayed by $T_i+T_o+T_{dp}+T_{servo}$
- With DSC or DSC with spline (requirement: position-control in the controller and in the drive is to the same encoder)
servoData.controlDifferenceCommandPosition
= **servoData.symmetricServoCommandPosition** delayed by $T_i+T_o+T_{dp}+T_{servo}+T_{additionalTime}$

Control loop structures

When the mode is changed from speed-controlled to position-controlled mode while the axis is in motion, the equivalent time of the position controller is required to apply the setpoint. The equivalent time is set during configuration in **dynamicData.positionTimeConstant** (for an electric axis) or in **dynamicQFData.positionTimeConstant** (for the hydraulic functionality).

Quantization of the control error for stepper motors or low-resolution encoders

The **commandValueQuantization.enable**=YES configuration data can be used to activate the quantization of the control deviation. A quantization of the control deviation is performed in accordance with the encoder resolution (distance per increment) or stepper motor increment.

This prevents, for example, the motor from oscillating between two increments while at a standstill.

With the setting **commandValueQuantization.mode**=DIRECT, the value for quantizing the control deviation can also be specified directly in **commandValueQuantization.value** (as of V4.1 SP1). This is sensible when for stepper motors, the encoder has a higher resolution than the increment of the stepper motor.

Note

Quantization of the control deviation should be activated for stepper motors.

Axis data sets

Like other items of axis configuration data, controller data can be assigned to multiple data sets. A data set is assigned to every axis when it is created.

Axis data sets are used as a means of activating multiple controller settings simultaneously.

Axis data sets are configured for axes on the **Axis data sets** tab under **Configuration**.

See also

Overview of commissioning the position controller of positioning axes (Page 147)

Hydraulic axis with position control/velocity control (Page 293)

Data sets (Page 200)

3.11.3 Dynamic Servo Control (DSC)

With the **Dynamic Servo Control (DSC)** function, the position controller in the drive is executed in the cycle clock of the speed control loop. In the drive, intermediate setpoints are generated in the speed controller cycle clock from the position setpoints transferred in the position controller/communication cycle clock via linear fine interpolation.

It is thus possible to set a substantially greater position controller gain factor **K**v. This increases the dynamic response for the reference variable sequence and disturbance variable compensation for highly dynamic drives.

For a position axis with position control and an assigned SINAMICS drive, the system sets DSC by default provided this is available at the drive's DO type (e.g. when a servo drive is assigned).

Advantages of DSC (compared to a position controller in the controller)

- Higher servo gain factor Kv (position controller gain) possible
- Larger bandwidth -> higher dynamic response
- Shorter response times for disturbance characteristic

Detailed information

The position difference (XERR) and the gain factor for the position controller in the drive are transferred in the PROFIdrive telegram, in addition to the speed precontrol value.

To activate the **DSC** function, the position controller must be set as PV controller (P controller with precontrol). In addition, the encoder in **typeOfAxis.NumberOfEncoders.DSCEncoderNumber** on the axis must specify to which increments the position differential (XERR) is normalized during operation. In SINAMICS, this is by default the motor encoder. **typeOfAxis.NumberOfEncoders.DSCEncoderNumber** must be initialized to the first encoder of the axis TO.

DSC is supported by MASTERDRIVES (standard telegrams 5 and 6 according to PROFIdrive), SIMODRIVE 611U and SINAMICS S120 (additional SIEMENS telegrams 105 and 106).

A SCRIPT is available to support you when commissioning MASTERDRIVES.

In SINAMICS, SIMODRIVE 611U, and MASTERDRIVES, the motor measuring system is used for normalizing the position difference in the drive.

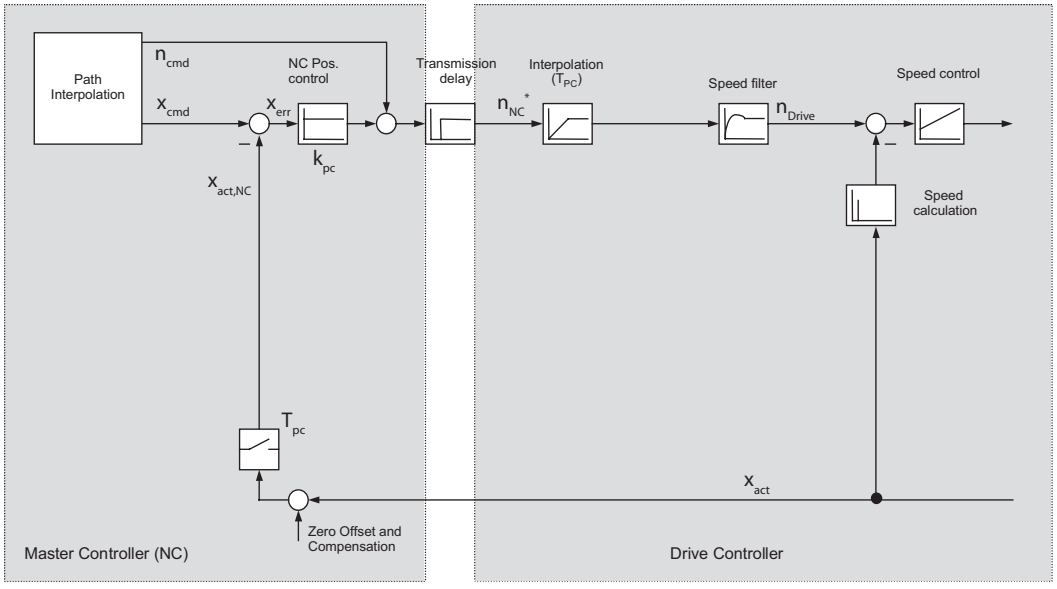
The following is to be taken into account for DSC:

- With DSC, XERR (position error) and Kpc (position control loop gain) are also transmitted, i.e., an 8-byte long setpoint telegram is required.
- With DSC, the communication times are taken into account in determining the following error of the setpoint since the actual comparison of setpoint and actual value is generated in the drive in the speed control cycle clock.
- The response time to a change in the actual position value is one speed controller cycle clock.
- With DSC and synchronous operation with actual value coupling, the extrapolation time must be increased by one position control cycle clock.

- If precontrol is deactivated for the DSC, zero is shown for the system variables with reference to the controller output or manipulated variable value.
  System variables:
  - **servodata.controllerOutput**
  - **actorData.setPoint**
  - **actorData.compensatedSetPoint**
  - **actorData.totalSetPoint**

  The axis is moved in accordance with the system deviation (XERR) given to the drive.
- If the direction in the drive is inverted for DSC p0410.0 and p0410.1 must be set the same, i.e. either the actual speed value and the actual position value are not inverted, or they are both inverted.

Structure



n_{cmd}: speed command x_{err}: position error command $_{pc}$: position control sampling time (=T_{MAPC})
x_{cmd}: position command x_{act}: actual position k_{pc}: position control gain

Figure 3-43 Structure of the position-control loop with the velocity setpoint interface to the drive without DSC

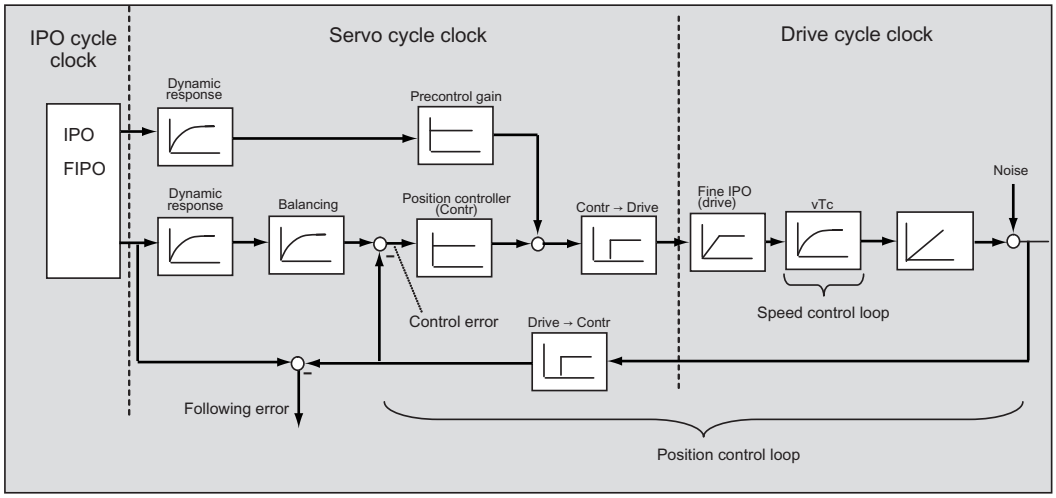
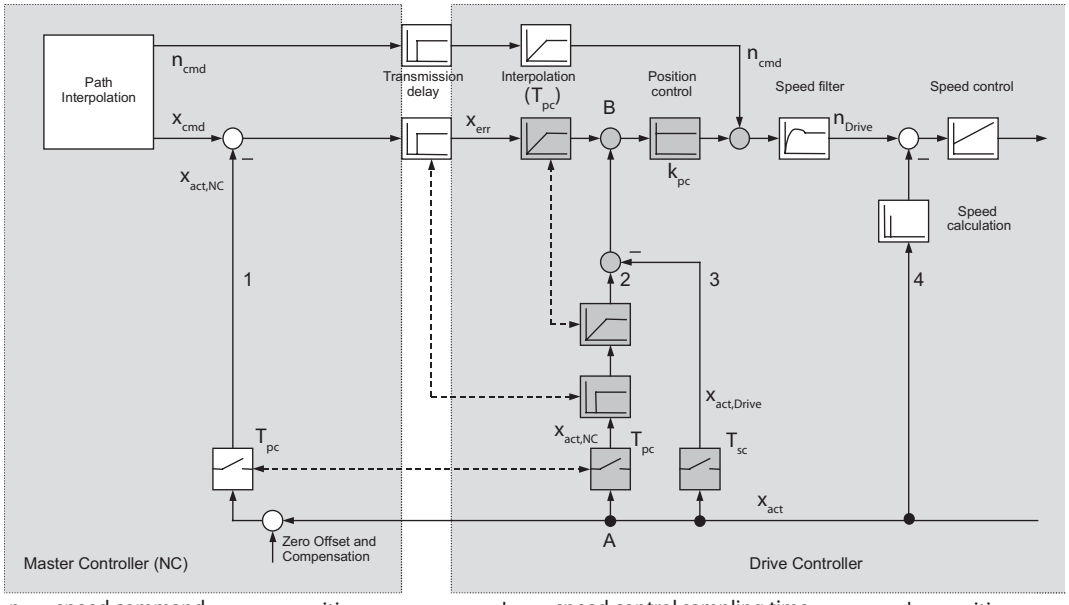


Figure 3-44 Control structure without DSC (simplified)



n_{cmd}: speed command x_{err}: position error command $_{sc}$: speed control sampling time k_{pc}: position control gain
x_{cmd}: position command x_{act}: actual position $_{pc}$: position control sampling time (=T_{MAPC})

Figure 3-45 Structure of the position-control loop with DSC functionality in the drive

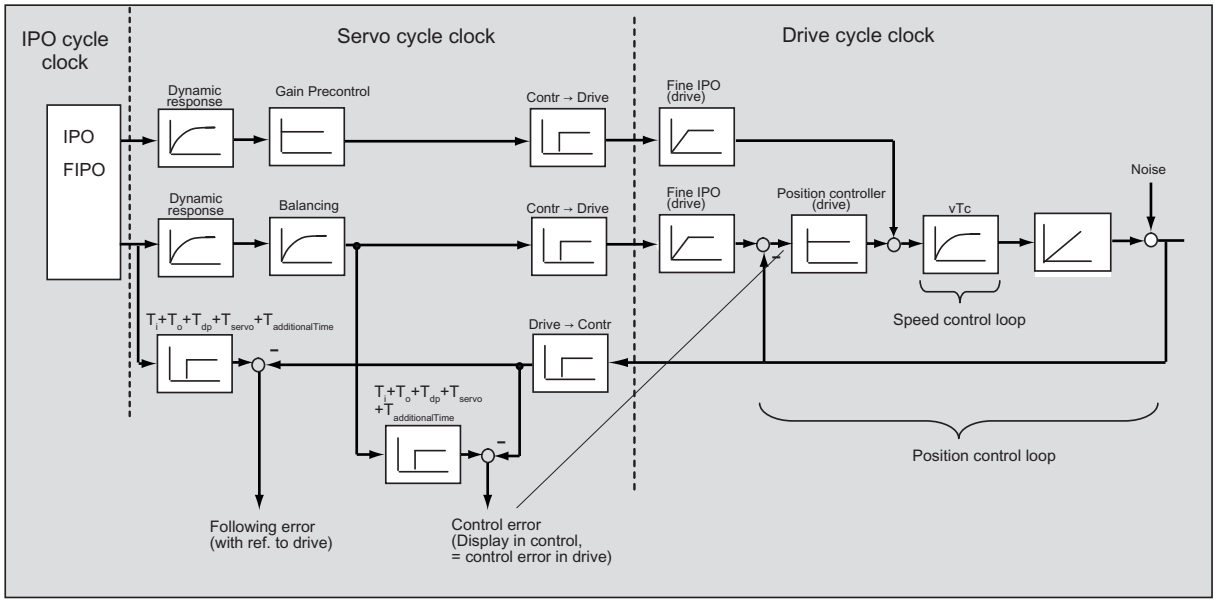


Figure 3-46 Control structure with DSC (simplified)

For more information about DSC, refer to the corresponding drive documentation, e.g. the SINAMICS S120 Function Manual for SINAMICS.

See also

Overview of commissioning the position controller of positioning axes (Page 147)

3.11.4 DSC with spline (as of V4.4) [Expert]

With the **Dynamic Servo Control (DSC)** function, the position controller in the drive is executed in the cycle clock of the speed control loop. In the drive, intermediate setpoints are generated in the speed controller cycle clock from the position setpoints transferred in the position controller/communication cycle clock via linear fine interpolation.

With the **DSC with spline** function, the position controller is also executed in the drive in the cycle clock of the speed control loop. The intermediate values in the drive are generated via polynomial functions. Intermediate setpoints are also generated for speed and torque, in addition to position. Using the intermediate setpoints, highly dynamic motions down to the current controller cycle clock are simulated exactly and precontrol is possible right down to the torque.

To use DSC with spline, activate “DSC with spline” in the drive of the function module.

The setpoints are fine-interpolated in the following way:

- Position setpoint - cubic fine interpolation
- Speed setpoint - quadratic fine interpolation
- Acceleration setpoint/torque setpoint - linear fine interpolation

DSC with spline provides the following advantages and functionality:

- Linear fine interpolation and precontrol of the torque
- Extended support for highly dynamic motions
- Process response is determined by the lower current controller equivalent time

DSC with spline can be used or set as follows:

1. **DSC with spline and torque precontrol**
   - The torque is precontrolled.
   - Speed and position are controlled taking into account the equivalent time of the current controller in the balancing filter in the drive.
   - The torque is calculated in the drive from the acceleration and the total moment of inertia.
   - Balancing of the position setpoint and of the speed setpoint using the current controller equivalent time (ttc).

   The total moment of inertia J is used to calculate a torque from the acceleration.
   The following parameters are available in the drive for input:
   - p0341 Motor moment of inertia (in motors on the motor list this parameter is automatically preset)
   - p0342 Moment of inertia ratio Load+motor / motor (should be defined during speed controller optimization, reference model determination)
   - p1498 Load moment of inertia
   - p1497 Moment of inertia scaling factor
2. **DSC with spline and speed precontrol**
   - The speed is precontrolled.
   - The position is controlled taking into account the equivalent time of the speed controller (VTC) in the balancing filter in the drive.
   - The speed controller equivalent time is taken into account in the balancing filter.
3. **DSC with spline without precontrol**
   - Speed and torque are not precontrolled.
   - The position is controlled in the drive.
   - The balancing filter in the drive and in the controller is not included.

Use DSC with spline with torque precontrol for constant load torque and
DSC with spline with speed precontrol for non-constant load torque.

If the torque constant of the feed motor does not remain in the overload range, the function module **Extended torque control** functions can be activated. This improves the accuracy of the torque simulation.

Requirements and settings in SIMOTION

The following requirements must be satisfied for the use of DSC with spline:

- Axis type must be a real electrical axis (REAL_AXIS)
- Telegrams 125 and 126 must be supported by the drive
  (SINAMICS drives as of V4.7 that support telegrams 125 and 126)
- Activation of DSC with spline
  DSC with spline is first activated when the axis is powered up if the axis is at standstill (standstill signal active).
- Consistent setpoints
  Position and velocity setpoints must be consistent with each other, i.e. the setpoint pair must originate from the same cycle clock time of the motion control.
  In the setpoint calculation of the technology object, the position setpoints and the velocity are consistent with each other and are limited to the maximum values.
  If the setpoints are modified using compensation values or limits after the setpoint calculation, the consistency of position and velocity must be ensured by the user.
  If the setpoints are defined by the user, the user must ensure consistency.
  System variable with DSC spline status display: **servoData.dscSpline**
- Fine interpolation type
  Use the default setting (configuration data item **FineInterpolator._type**=CUBIC_MODE)
- High-resolution measuring system
  DSC with spline requires a high setpoint resolution. The setpoint resolution in the system is based on the actual value resolution. If the actual value only has a low resolution, you can increase the setpoint resolution internally.
  (Set p0418, e.g. to 18-bit)
- Recommendation: Set the equivalent time of the speed control loop and the current control loop to at least 3x current control cycle clock
  VTC configuration data item **dynamicData.velocityTimeConstant**
  TTC configuration data item **dynamicData.torqueTimeConstant**
- Switching the setting from DSC with spline to standard DSC and back
  controllerStruct.PV_Controller.enableDSCSpline configuration setting
  Restart required
- Speed control mode
  Performed via Kpc=0.
  The position for the generation of the position difference (XERR) is simulated internally.
- Speed precontrol = 100%
  Precontrol factor is not evaluated for the DSC with spline setting.
- Friction compensation
  This setting can cause irregular splines at the start of motion.
- Backlash compensation
  This setting can cause irregular splines at the direction reversal.

- Setpoint superimposition
  DSC without spline is used by the system, as long as superimposition is active, by activating system variable **servoSettings.additionalCommandValueSwitch**.
- Manipulated variable superimposition
  DSC without spline is used by the system, as long as superimposition is active, by activating system variable **servoSettings.additionalSetPointValueSwitch**.

Restrictions

The following restrictions apply in SIMOTION when using DSC with spline:

- Encoder switchover during operation or during a motion is not supported
- Pressure control is not possible
- Pressure limiting is not possible

Closed-loop control on a direct measuring system

A direct measuring system can be controlled on condition that the control is set to the same encoder system in the controller and in the drive.

The encoder for the position control in the drive is set in p1192.

The transmission factor between the motor measuring system (MMS) (motor encoder) and the direct measuring system (DMS) is set in p1193.

X_ERR is further normalized to the motor measuring system if not converted in the drive.

Encoder switchover during operation or during a motion is not supported.

Note

Position and velocity values must be consistent

Closed-loop control on a direct encoder in the controller in conjunction with closed-loop control on a motor encoder in the drive is not permitted. The position and velocity values would then no longer be consistent in the spline calculation.

The user must ensure the values are consistent with each other if the position setpoints and velocity setpoints change.

Balancing filter

When DSC with spline is activated, the balancing filter is calculated in the drive. The balancing time is transferred to the drive in the telegram. The setpoint to calculate X_ERR is therefore tapped before the balancing filter.

The controller calculates the actual control error in the drive taking into account the balancing filter.

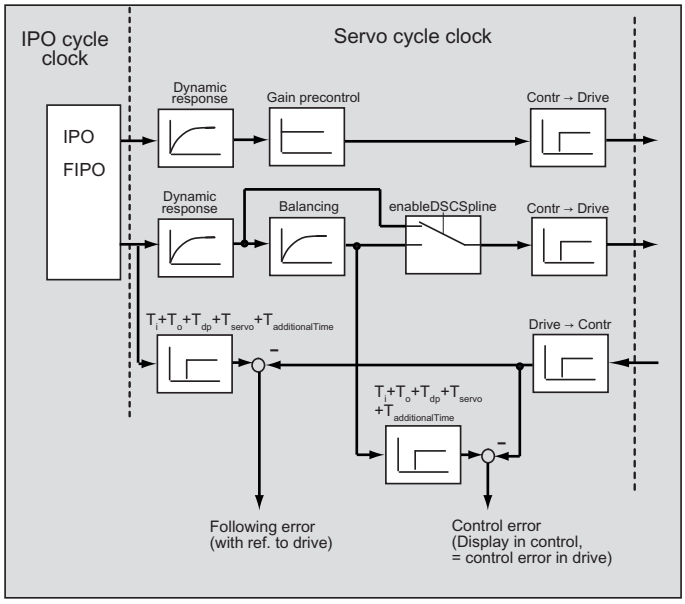


Figure 3-47 Effect with activated DSC with spline

3.11.5 Fine interpolation

The purpose of the fine interpolator (FIPO) is to generate interim setpoints for the position setpoints when the interpolator (IPO) and the controller (servo) have different cycle clock ratios.

Interpolation types

During configuration, the following interpolation types can be set by means of the **FineInterpolator._type** configuration data element:

- **DIRECT_MODE:** when no fine interpolation is required
- **LINEAR_MODE:** linear interpolation (continuous position for positioning axis)
  Use for discontinuous velocity setpoints (no precontrol)
- **QUADRATIC_MODE:** quadratic interpolation (continuous velocity for positioning axis)
  Use for continuous velocity setpoint curves
- **CUBIC_MODE (recommended, default setting):** cubic interpolation
  Use for continuous velocity or continuous acceleration setpoint curves

With the positioning-axis setting, the set position is interpolated.

With the speed-controlled axis setting, the set velocity is interpolated.

3.11.6 Dynamic controller data

The equivalent time of the current control loop is set in the **dynamicData.torqueTimeConstant** configuration data element. The equivalent time of the current control loop is not used at present.

The equivalent time of the speed control loop is set in the **dynamicData.velocityTimeConstant** configuration data element and used in the balancing filter. See also *Balancing filters for precontrol* in Position control (Page 111).

The setting can be made using the *Closed-loop control* dialog for *Dynamic controller data* (activate expert mode).

The equivalent time of the position control loop is set in the **dynamicData.positionTimeConstant** configuration data element. The equivalent time of the position control loop is set in the following cases:

- Preassigned braking ramp
- Switchover from SPEED_CONTROLLED to POSITION_CONTROLLED
- Switchover from pressure-controlled to position-controlled operation
- A moving axis is enabled with **_enableAxis()**

If the equivalent time of the position control loop has not be set correctly, compensation movements can occur for switching tasks.

With DSC, the equivalent time of the position control loop can be set as follows:

- Without precontrol
  PTC = 1/Kv
- For 100% precontrol of the velocity setpoint:
  The equivalent time of the position control loop can be set as equal to the equivalent time of the speed control loop (PTC = VTC).
  (Control loop optimization for minimal overshoot)

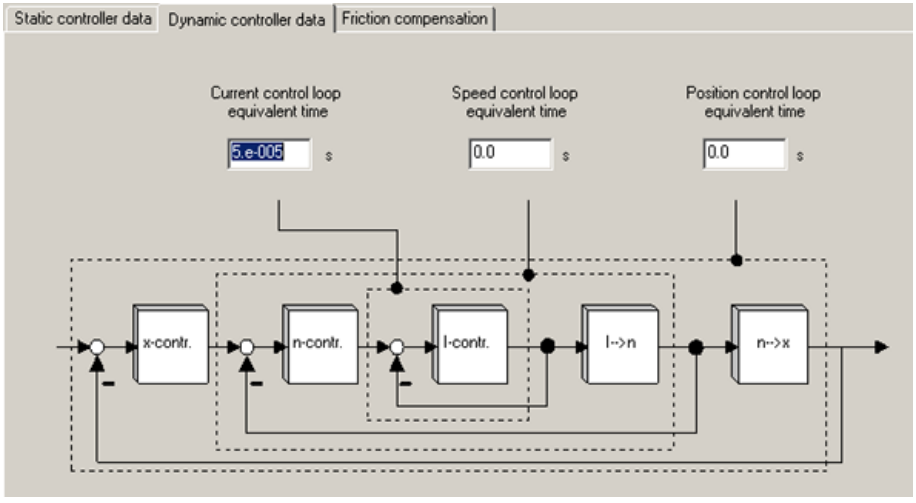
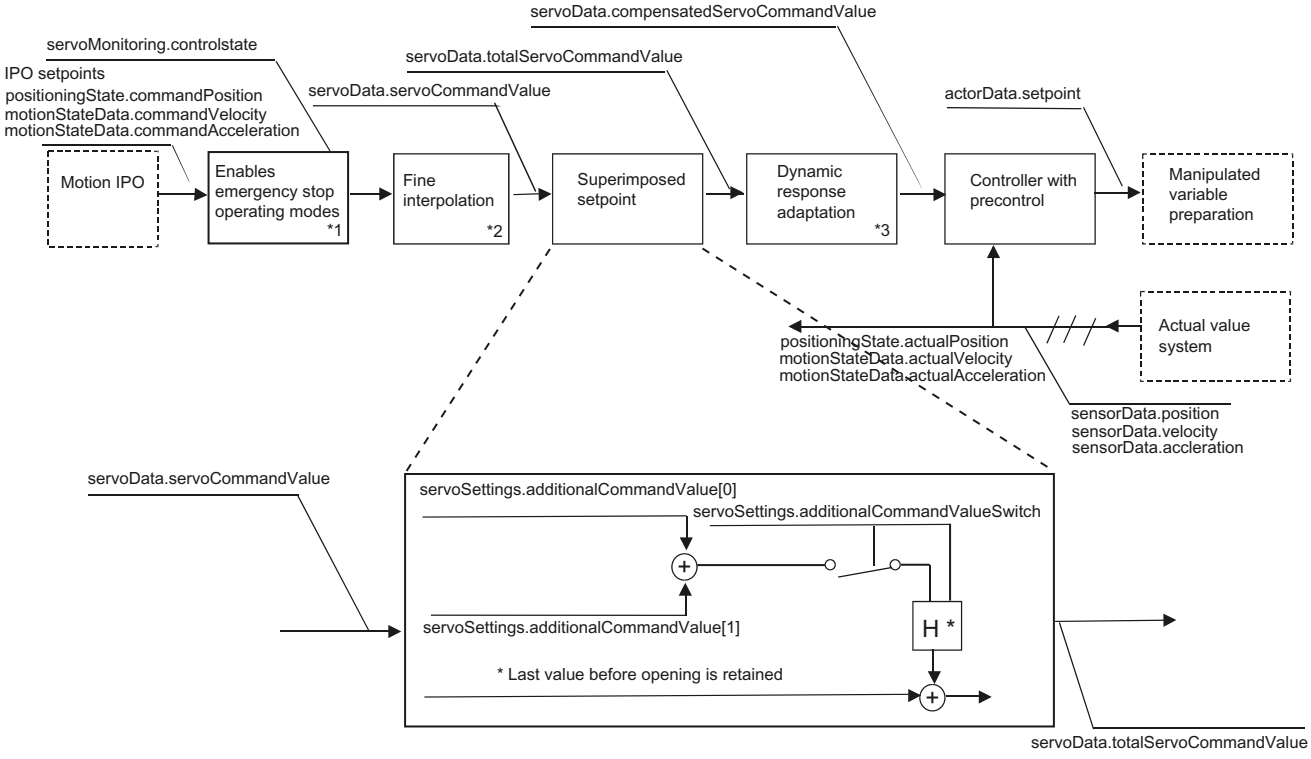


Figure 3-48 Dynamic controller data

3.11.7 Setpoint superimposition

The setpoint specified by the interpolator can be superimposed in the servo via the cyclically active system variables
(**servoSettings.additionalCommandValue**,
servoSettings.additionalCommandValueSwitch)



*1 If there is no enable or emergency stop, IPO is in follow-up mode, i.e. IPO setpoint = IPO actual value

*2 Selectable: linear, cubic, quadratic, direct

*3 PT2 element with T1, T2

Figure 3-49 Setpoint superimposition

Superimposition has an effect on the position of the position axis. It is also effective during active position control and interpolator (IPO) in follow-up mode.

No setpoint superimposition occurs for traversing the axis in the SPEED_CONTROLLED mode and for active force/pressure control.

If the setpoint superimposition is disconnected, the last value of the setpoint superimposition is retained. If the value should no longer function, it must be explicitly set to zero.

Note

During execution of the alarm response FEEDBACK_EMERGENCY_STOP and the execution of **_stopEmergency()** with stopMode=STOP_WITH_COMMAND_VALUE_ZERO:

- The actual value (position and velocity) is accepted once and the preassigned deceleration ramp is applied
- The superimposed setpoint is unclamped (as of V4.0)
- The switch for the superimposed setpoint is opened (status displayed in system variable).
- A switch is prevented from closing or the switch is closed via the system variable and triggers Alarm 50021 "ServoSettings system variable (Element /1/%d) cannot be write-accessed due to a stop response".

With the FEEDBACK_EMERGENCY_STOP alarm response, there is no setpoint superimposition.

When a stop ramp is assigned (e.g. **_stopEmergency(**...WITH_COMMAND_VALUE_ZERO) **)**, the superimpositions are deleted and transferred to the setpoint generation. This triggers Alarm 50020 "Servosettings system variable (Element /1/%d) is reset due to a stop response".

For switching to SPEED_CONTROLLED (as of V4.1 SP1) and in pressure control (as of V4.1 SP1), the setpoint superimposition is not active.

3.11.8 Dynamic response adaptation

In order to adapt the dynamic behavior of axes, the setpoint branch contains a programmable PT2 setpoint filter with the time constants T_1, T_2, and T_t. This allows compensation for axes with higher dynamic behavior with the lowest dynamic behavior (axis with the largest equivalent time constant of the T_{LR} position controller).

The dynamic response of the axes is determined by T_{Res}, the resulting total time constant.

$T_{Res} = T_{LR\,1}$ (axis with the smallest dynamic behavior)

$T_{Res} = T_{da} + T_{LR\,2}$ (considered dynamic axis)

The dynamic adaptation, T_{da}, must be selected so that the resulting total time constants are equal for all axes to the adapted.

The dynamic adaptation consists of
$T_{da} = T_1 + T_2 + T_t$

T_1	additive time constant 1 (setting in the configuration data: **NumberOfDataSets.DataSet_1.DynamicComp.T1**)
T_2	additive time constant 2 (setting in the configuration data: **NumberOfDataSets.DataSet_1.DynamicComp.T2**)
T_t (V4.1 SP1 and higher)	Dead time (setting in the configuration data: **NumberOfDataSets.DataSet_1.DynamicComp.deadTime**)

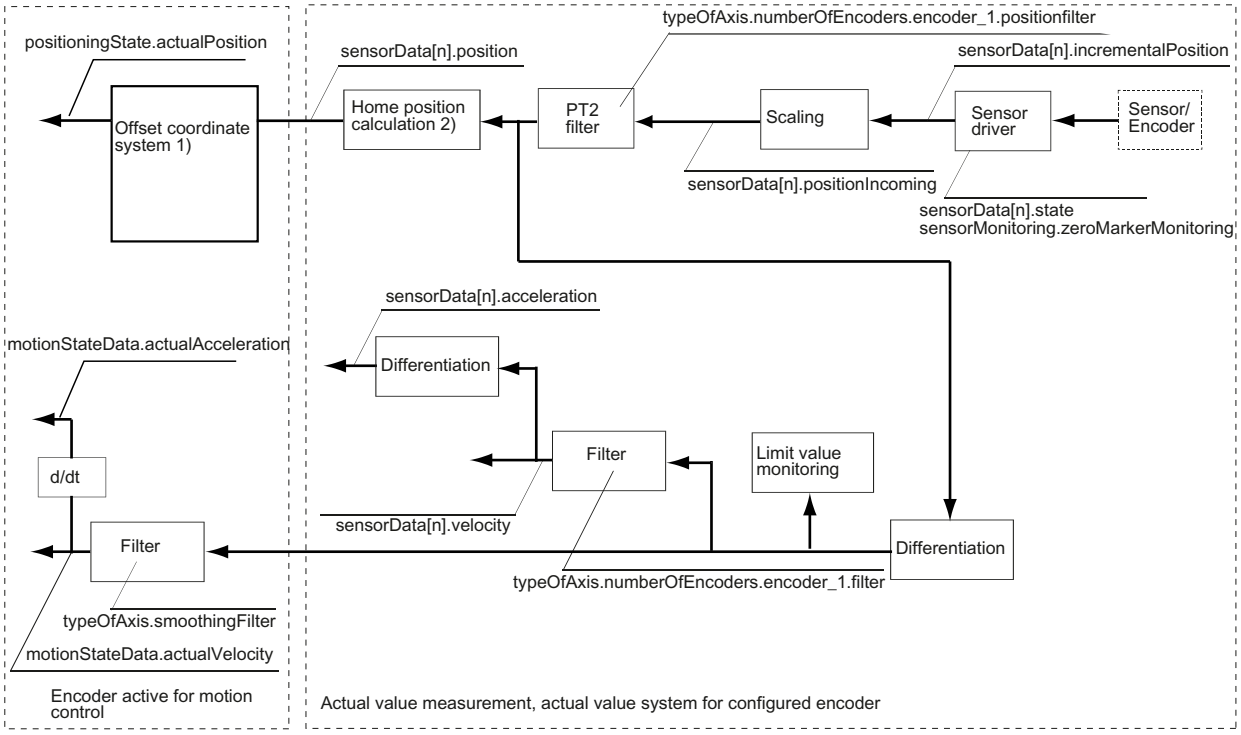
T_{Res}	(Desired) resulting total time constant of the axis
T_{LR}	Equivalent time constant of the closed position control loop of the axis (see **dynamic~Data.positionTimeConstant**)

The function is enabled/disabled via **dynamicComp.enable**.

Note

The used procedure means that a setpoint delay can be implemented exactly by specifying a dead time.

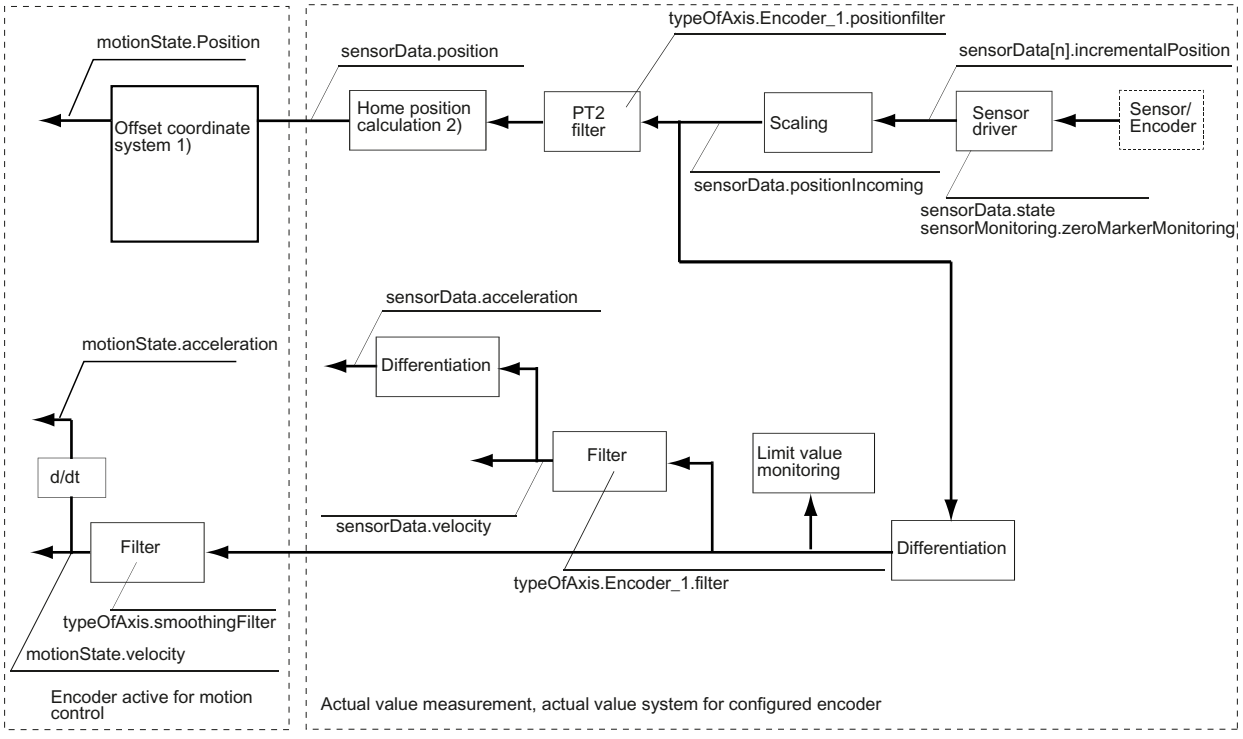
3.11.9 Actual value measurement / actual value system



1) Absolute and incremental encoder: Offset coordinate system e.g. using _redefinePosition() and _homing() with the DIRECT_HOMING setting
Incremental encoder: Home position userDefaultHoming.homePosition, home position offset incHomingEncoder.proceedshiftpos

2) Incremental encoder: Positions are zeroed at zero mark
Absolute encoder: Absolute encoder adjustment in absShift or absoluteEncoder.totalOffsetValue is included

Figure 3-50 Actual value system of technology object axis



1) Offset coordinate system e.g. using _redefineExternalEncoderPosition and _synchronizeExternalEncoder with the DIRECT_HOMING setting

2) Incremental encoder: userDefaultHoming.homePosition
Absolute encoder: absShift

Figure 3-51 Actual value system of technology object external encoder

The monitoring of the **actual velocity** and **actual acceleration** is used to identify errors in the control loop of the drives. If the rise in the actual value exceeds the **encoder limit frequency**, an **alarm** is triggered.

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

Display

The actual data for each sensor/encoder is displayed in the following variables:

- sensorData[n].position
- sensorData[n].velocity
- sensorData[n].acceleration

The system variables for **sensorData** are calculated in the servo cycle clock.

The actual axis value that is active for closed-loop control, the IPO cycle clock, and master value coupling is displayed in the following variables:

- positioningState.actualPosition
- motionStateData.actualVelocity
- motionStateData.actualAcceleration

The system variables for **positioningState** and **motionState** are calculated in the IPO cycle clock.

These actual values comprise the reference for the output cam calculation in the IPO cycle clock, the actual value coupling for external encoders without extrapolation, and the actual value reference in the IPO cycle clock, e.g. for actual-position-related profiles.

Noisy encoder signals lead to high velocity jumps. These can be reduced or compensated by using suitable filter settings.

Filtering of the velocity

The **smoothingFilter** configuration data element refers to the velocity calculated in the IPO cycle clock. Here, you can select whether a PT1 filter is to be applied to the data or whether the data is to be generated from the mean value. The mean value is determined from the ratio of the servo cycle clock to the IPO cycle clock.

The **numberOfEncoders.encoder_1.filter** configuration data element relates to the velocity calculated in the servo cycle clock. A PT1 filter is used.

During synchronous operation with master value reference to the actual axis values, these axis values are derived separately (refer to *Synchronous Operation description of functions, Actual value coupling*).

Actual position filtering (as of V4.1 SP1)

A sensor/encoder-specific actual position value filter is available.

This actual value filter is set in **typeofAxis.numberOfEncoders.encoder_1.positionfilter.T1** and **~.T2** and activated via **~.enable**.

The *positionFilter* has no effect with the velocity encoder setting **encoderValueType**=VELOCITY.

The actual velocity and actual acceleration are derived from the filtered position.

The *positionFilter* is calculated based on the actual values in SIMOTION and so does not act for active DSC for the actual values of the lower-level control loop in the drive.

The *positionFilter* present on the analog sensor input (for raw value filtering) is not dependent on the filter described here.

Extrapolation

For a synchronized group with actual value coupling (e.g. master value is the actual encoder value of an axis or an external encoder), the associated principle means delay times result because of bus communication, system cycle clocks and clock-pulse scaling, fine interpolation, position setpoint filters, and controller settings. These times can be compensated using an extrapolation (**Extrapolation.extrapolationTime**).

Extrapolation means knowing what has gone before and casting a glance at what is to come in the future (extrapolation time). In the case of dynamic changes to the master value, the extrapolation time should be as short as possible. An IPO cycle clock ratio: Servo pulse duty factor of 1:1 is good.

Note

Extreme care must be taken when changing the extrapolation time to the runtime; otherwise knocking could result in the machine.

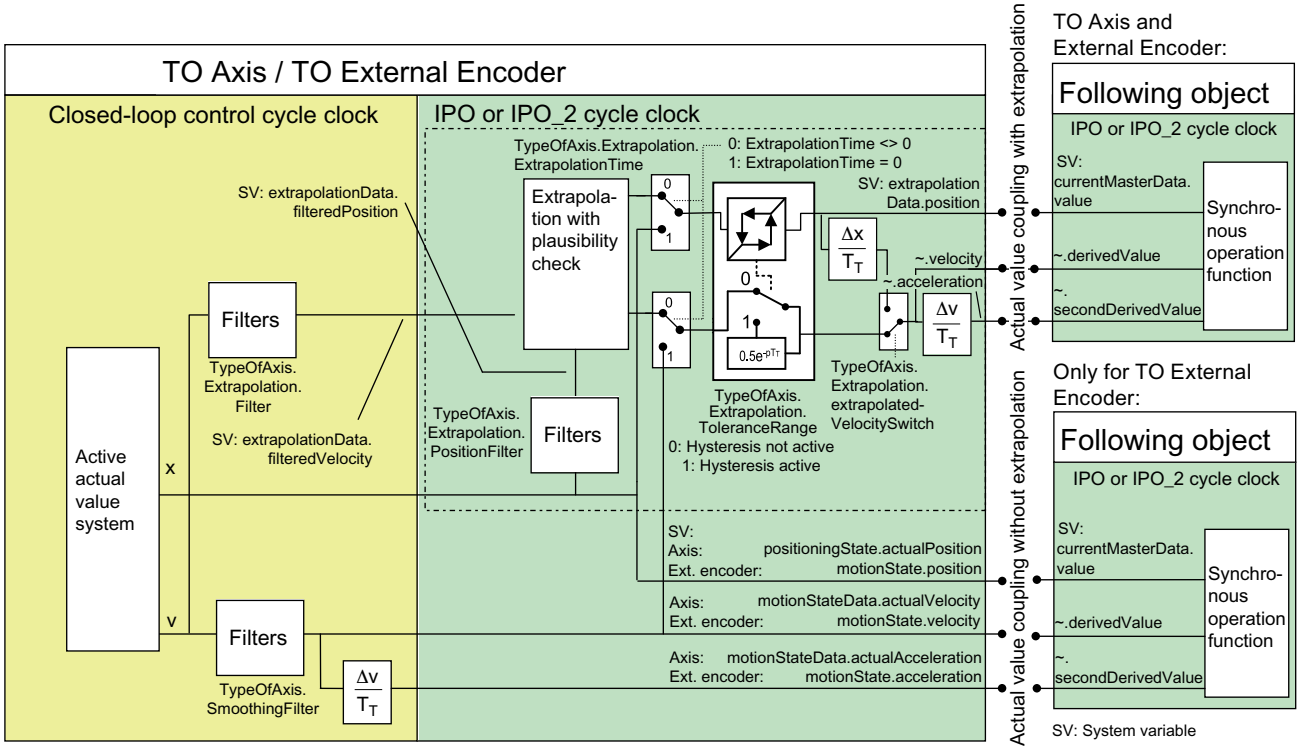


Figure 3-52 Actual value coupling with extrapolation for the Axis technology object or External Encoder technology object

During master value extrapolation, filtering on the actual velocity value is performed separately by means of a PT1 filter/mean value generation that is set with **typeOfAxis.extrapolation.Filter**.

The actual position value for synchronous operation can be filtered separately during extrapolation by means of a PT2 element. (as of V4.1 SP1)

As with extrapolation, there is only one filter for each axis rather than for each sensor/encoder. It is set in **typeOfAxis.extrapolation.positionFilter.T1** and **~.T2**. The filter acts on the actual position for the extrapolation, see also *Technology Objects Synchronous Operation, Cam* function manual, *Actual value coupling with extrapolation* section.

The velocity for the extrapolation is taken over from the actual values of the axis or External Encoder before application of the smoothing filter (**typeOfAxis.smoothingFilter**).

Although the velocity filter (**Extrapolation.Filter**) does not affect the extrapolation time, it is significant in the case of dynamic changes to the master value (due to the values being delayed).

The filter for the velocity during extrapolation is independent of this filter.

We recommend setting the velocity filter (**Extrapolation.Filter**) first; if the result is not sufficient, use the position filter as well. The position filter times are an additional factor to be taken into account in the extrapolation time.

For extrapolation, you can specify in **typeOfAxis.extrapolation.extrapolatedVelocitySwitch** whether the master value velocity should be recalculated from the extrapolated position or whether the velocity derived for the extrapolation should also be used as the master value velocity.

The extrapolated and filtered actual values can be checked in the following system variables:

- Filtered and extrapolated
  - **extrapolationData.position**
  - **extrapolationData.velocity**
  - **extrapolationData.acceleration**
- Filtered and not extrapolated
  - **extrapolationData.filteredposition**
  - **extrapolationData.filteredvelocity**

(This topic is presented in detail in the *Technology Objects Synchronous Operation, Cam* Function Manual under Actual value coupling with extrapolation.)

Note

You can find a tool to help you calculate extrapolation times under *Tools and Documentation > Calculating the extrapolation time for an actual-value-coupled drive* in the *SIMOTION Utilities & Applications*. *SIMOTION Utilities & Applications* is part of the scope of delivery of SIMOTION SCOUT.

Transferring the actual velocity from the drive (as of V4.1 SP1)

With velocity precontrol, sudden changes in velocity directly affect the control behavior (quantization jumps primarily in the case of low-resolution encoders).

With the setting **typeOfAxis.numberOfEncoders.encoder_n.encoderValueType**=POSITION_AND_PROFIDRIVE_NIST_B, you have the option of converting the speed of rotation transferred in PROFIdrive NIST_B to a velocity and applying this value as the actual velocity of the encoder/sensor. In this case, the actual position of the sensor does not need to be differentiated to derive the actual velocity.

With the setting **typeofAxis.numberOfEncoders.encoder_n.encoderValueType**= POSITION_AND_DIRECT_NIST, a speed of rotation transferred in the I/O area and normalized as NIST_B is taken as the actual value and converted to an actual velocity. In this case, 4000H corresponds to 100%. The address is set in **typeofAxis.numberOfEncoders.encoder_n.sensorNist.logAddress**, and the reference value is set in **typeofAxis.numberOfEncoders.encoder_n.sensorNist.referenceValue**.

With encoders with NAct evaluation, the speed determined by the encoder and the resulting velocity can be accepted by the encoder. In this case, the actual position of the sensor does not

need to be differentiated to derive the actual velocity.
Two transmission options are available:

- Transmission in the PROFIdrive telegram
- Transmission in the I/O area

Number of modulo revolutions (as of V3.2)

The number of modulo revolutions is displayed in the **positioningState.commandModuloCycles**, **positioningState.actualModuloCycles** and **sensorData.moduloCycles** system variables with the following constraints:

- The value is not initialized at first.
  The start value must be stored in the user program so that, for example, the number of revolutions since the start can be calculated subsequently.
- The value is reset when the system is switched on and in response to initialization tasks such as set actual value system (**_redefinePosition()**) or homing (**_homing()**).
  The modulo revolution can change via **_redefinePosition()**. For absolute position specifications and stationary axes, the principle of the shortest path is used; for moving axes, corrections are made in the direction of motion. For relative position specifications, the correction is made over the entire position difference.
- There is no special overflow handling for the count value.
  A counter overflow must be taken into account in the user program.

Table 3-26 System variables for determining the modulo revolutions

Variable	State	Meaning
positioningState.commandModuloCycles	See the **StructAxisPositioningState** data type in the System variables parameters manual	Setpoint for modulo revolutions
positioningState.actualModuloCycles	See the **StructAxisPositioningState** data type in the System variables parameters manual	Actual value for modulo revolutions
positioningState.moduloCycles	See the **StructAxisSensorData** data type in the System variables parameters manual	Actual value for modulo revolutions

3.11.10 Preparation of manipulated variables for electric axis

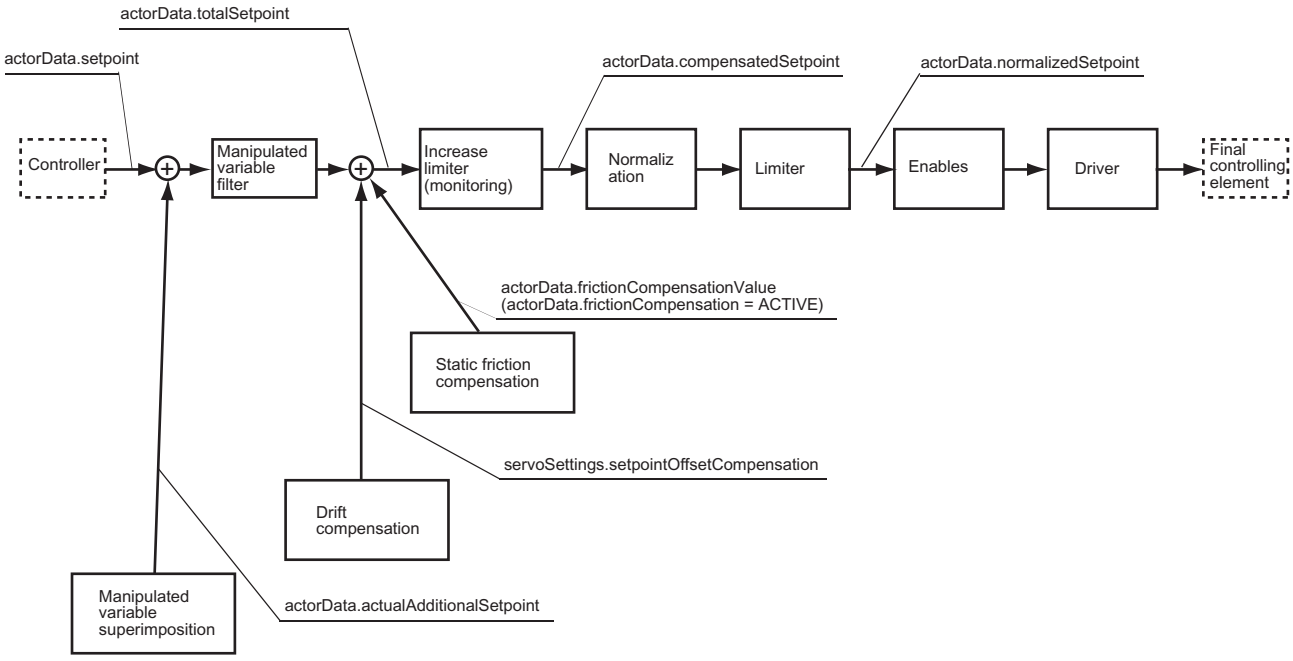


Figure 3-53 Manipulated variable preparation

3.11.11 Manipulated variable superimposition

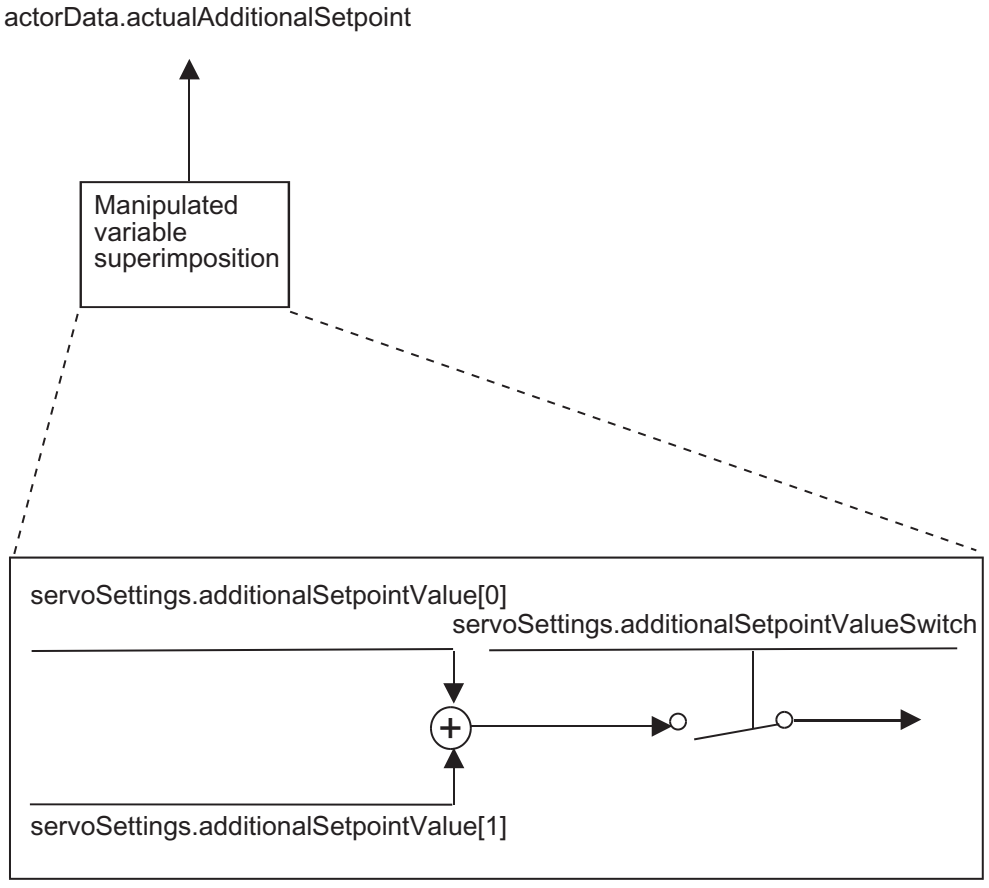


Figure 3-54 Manipulated variable superimposition

Superimposition of manipulated variables is enabled by means of a switch.

Manipulated variable superimpositions remain in effect when the drive is active. The user is responsible for handling the superimpositions.

As of V4.1 SP1, the superimposition of manipulated variables is not active for the FEEDBACK_EMERGENCY_STOP alarm reaction and the **_stopEmergency()** command with stopMode=STOP_WITH_COMMAND_VALUE_ZERO.

3.11.12 Manipulated variable filtering (as of V4.1 SP1)

A man. var. filter can be set as a PT1 filter in the **setpointFilter** configuration data element. This filter acts after the controller and after the precontrol value and the additive manipulated variable value (additionalSetpoint) have been added.

A change in the filter data takes effect immediately.

With the DSC setting, the man. var. filter is only effective for the precontrol.

3.11.13 Drift/offset compensation

The analog output signal of analog-coupled drives can include a **drift**. This can be compensated for by an offset in the axis.

Drift is enabled/disabled using the **DriftEnable** configuration data element. The value is specified in the system variable **servoSettings.setpointOffsetCompensation**.

The value of the system variable **servoSettings.setPointOffsetCompensation** is reset to 0.0 with **_disableAxis()** (or with the alarm response RELEASE_DISABLE). If a value not equal to 0.0 is effective when the axis is switched on, the value needs to be reset before switching on the axis (before issuing **_enableAxis()**).

3.11.14 Static friction compensation

A simple compensation is available for overcoming for the forces of static friction. During startup from a standstill, a DT1 element adds a static friction compensation signal to the manipulated variable.

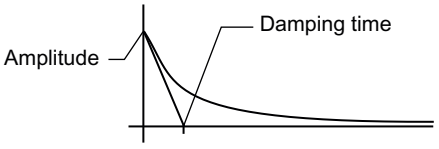


Figure 3-55 Static friction compensation

Static friction is added relative to the velocity setpoint. It only takes effect when processing motion specifications and does not affect force/pressure control.

The standstill identification for static friction compensation can be set separately, as is the case for the amplitude and the decay response. The amplitude and decay response are set in the configuration.

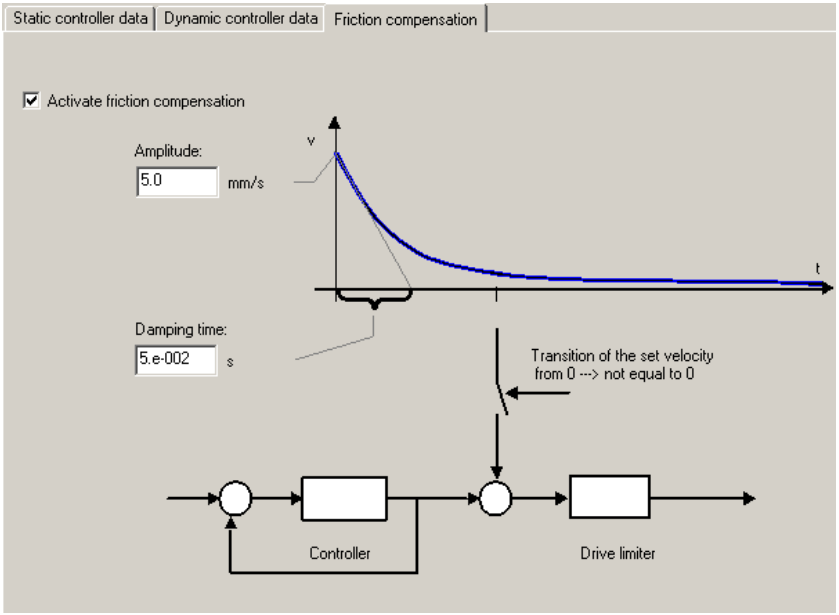


Figure 3-56 Static friction compensation

3.11.15 Backlash on reversal compensation

During the power transmission between a moving machine part and the corresponding drive, a backlash on reversal (play) usually occurs.

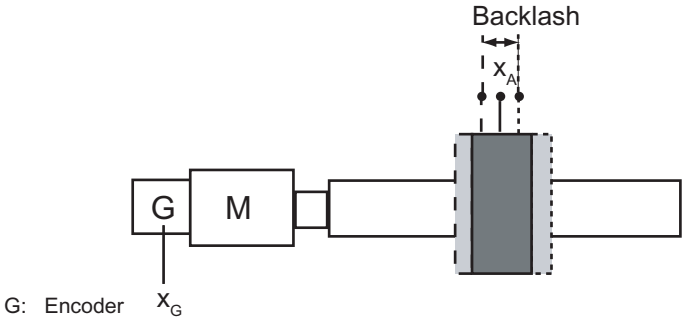


Figure 3-57 Backlash

Despite the backlash on reversal, a clear derivation of the axis position from the encoder position and an exact traversing, positioning and synchronous operation of the axis must be possible. SIMOTION provides the backlash on reversal compensation function for this purpose.

The backlash is specified in **absBacklash.length** or **incBacklash.length**, and the backlash compensation velocity is specified in **absBacklash.velocity** or **incBacklash.velocity**.

Backlash on reversal compensation is effective only for encoders mounted on the motor side and is set specifically for each encoder. A direct measuring system measures the variable directly without any intermediate backlash. Therefore, any existing backlash is compensated with closed-loop control to the direct measuring system.

Backlash type

- Positive backlash
  A positive backlash is set with **absBacklash._type** = POSITIVE or **incBacklash._type** = POSITIVE.
  Setting =POSITIVE means that the mechanical position lags behind the actual encoder value. This is the case, for example, if the ball screw has play and the encoder is attached to the motor (normal condition, default).
  When the direction is reversed, the backlash is applied by the system at the backlash compensation velocity.
  Unless stated otherwise, the POSITIVE setting is assumed in the following.
- Negative backlash
  The setting **incBacklash._type** = NEGATIVE is not supported.
  The setting **absBacklash._type** = NEGATIVE is not supported.

Homing with incremental encoders

In homing, the encoder value is assigned a unique (mechanical) axis position based on the reference signal of a homing mark.

If there is a backlash, the axis must always be traversed from the same side to the synchronization point during homing. The assignment of the actual control position to the mechanical position of the axis is thus unique.

This applies to:

- Homing on encoder zero mark
- Homing on external zero mark
- Homing via actual value setting (setting the actual axis value to a specified value)

Note

The fraction backlash that the axis lags when traversing in the homing direction is irrelevant. Homing yields a unique assignment of the mechanical and displayed axis position relative to the encoder value. When traversing in this direction, the same ratios always result.

Direction reversal and switch-on behavior of incremental encoders

In a direction reversal when **incBacklash._type**=POSITIVE, the motor runs through the backlash range. During this motor motion, the mechanical and thus the displayed actual position of the axis does not change, whereas the encoder value in **sensordata.incrementalPosition** does change. The axis is then traversed by the commanded distance or to the commanded position.

In case of direction reversal, the backlash compensation is independent on the 'homed' status. However, the first motion after the control unit is switched on is run through without backlash compensation.

When the backlash range has been run through completely one time (no matter which direction), the set backlash is then compensated when the direction is reversed - if backlash compensation is enabled. This is independent of the homing status and applies to the relative or, if applicable, absolute traversing of the axis in the non-homed state.

Homing with absolute encoders

The absolute encoder value is assigned a mechanical axis position by defining the absolute encoder offset for homing with absolute encoders or absolute encoder adjustment.

This results in a direction dependency for the absolute encoder, too, since the position of the backlash relative to the encoder value/axis position is relevant when setting the absolute encoder offset or the mechanical axis position.

After the absolute encoder adjustment, backlash on reversal is carried out if the direction is reversed, but not when traversing is continued in the same direction.

If the control unit is switched off/on, the mechanical axis position is assigned/indicated to the actual encoder value via the absolute encoder offset. In the same way as motion restart after absolute encoder adjustment, backlash compensation is not activated after switch-on for motion in the same direction as the absolute encoder adjustment; however, it is activated for motion in the opposite direction.

In the **absBacklash.startupDifference** configuration data element, the direction opposite to the reference direction must be entered for setting the absolute encoder offset. For example, **absBacklash.startupDifference**= NEGATIVE must be set for referencing the absolute encoder offset to a positive direction of motion, since the backlash must be compensated in this case

when immediate travel in the negative direction occurs after switch-on. This is independent of the position of the backlash relative to the actual axis position at the time of switch-on.

Note

Directly after switch-on, the mechanical axis position is displayed correctly only if the position of the backlash at the time of switch-on corresponds to the position of the backlash relative to the mechanical axis position when the absolute encoder offset is set. If this is not the case, the mechanical axis position may deviate from the displayed axis position up to the total amount of the backlash; that is because the control detects the actual encoder value at the time of switch-on but it is not able to establish the position of the backlash without moving the axis.

Direction reversal and switch-on behavior of absolute encoders

With direction reversal and the setting **absBacklash._type**=POSITIVE, the motor runs through the backlash range. During this motor motion, the mechanical axis position and, thus, the displayed actual position of the axis does not change, whereas the encoder value in **sensordata.incrementalPosition** does change (also with the absolute encoder). The axis is then traversed by the commanded distance or to the commanded position.

Backlash compensation for direction reversal is independent of the 'Homed' status (here, the absolute encoder adjustment).

Backlash compensation is not activated for the first motion in both directions once the control has been switched if no absolute encoder adjustment has yet been carried out. When the backlash range has been run through completely one time (no matter which direction), the set backlash is then compensated when the direction is reversed - if backlash compensation is enabled.

Status display

The **sensorMonitoring.passingBacklash** system variable indicates that the backlash is being run through without changing the actual axis value.

Since the travel specification resulting from the specified motion and backlash recovery is superimposed, this indication is not identical to the specification for backlash recovery.

Note

In order not to impair the response times with the backlash, the backlash compensation is started simultaneously with the motion.

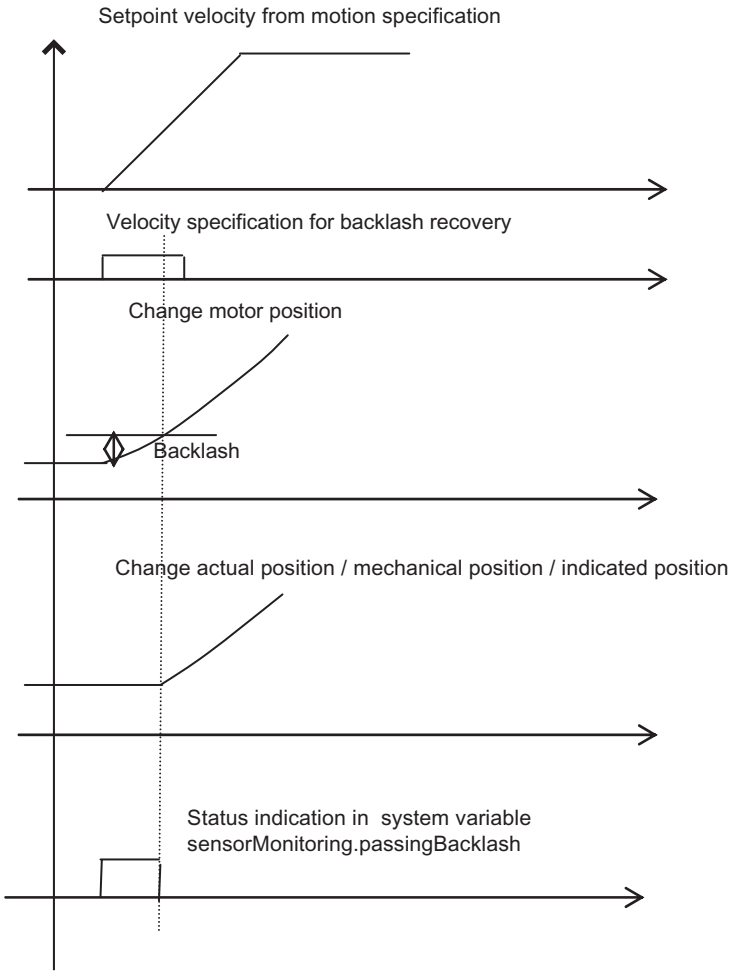


Figure 3-58 Backlash on reversal compensation sequence

3.11.16 Traversing of the positioning axis without position control

The positioning axis can also be traversed without active position control.

Traversing motions with **_move()** or as velocity profiles can be applied on the position axis/ synchronized axis with position control or, alternatively, with a velocity specification only.

Table 3-27 Setting via movingMode parameter in command

Commands	Function
_move()	For traversing with a programmable velocity profile
_runTimeLockedVelocityProfile()	For traversing with a user-definable velocity profile
_runPositionLockedVelocityProfile()	For traversing with a position-related velocity profile

The transition from a position-controlled or open-loop/closed-loop pressure-controlled motion to a motion solely with a velocity/speed specification can take place when the axis is at a standstill or in motion; this is also the case for the transition from a motion with speed specification to a position-controlled motion.

The **decodingConfig.speedModeSetpointZero** configuration data element can be used to specify whether the current velocity is maintained during the switchover from position-controlled or open-loop/closed-loop force-/pressure-controlled operation to operation with velocity/speed specification, or whether the velocity is set to zero at the time of switchover.

The dynamic response parameters and the maximum values for speed specification are derived from the settings for position-controlled axis operation.

The position-related monitoring functions are deactivated. The encoder limit frequency can be exceeded.

It is possible to activate the position-controlled axis with **_enableAxis()** for the velocity specification only.

See also

Setting and canceling the axis enables (Page 313)

Moving (Page 329)

Traversing the axis via velocity specifications (Page 163)

3.11.17 Stepper drives

The special torque characteristics of a stepper motor and the response to overloading should be considered when assigning the stepper motor axis parameters.

Functionality of stepper motor axes

The Axis technology object can be used to implement the following functions:

- Positioning axis without additional encoders
  - Homing with edge of external zero mark
  - Rotation monitoring with external zero mark
- Position axis with additional incremental or absolute encoder
  - The axis works in the same way as with a servo motor with an analog interface +/-10 V.

Behavior of a stepper motor

From a given speed of rotation of the stepper motor (around 500 rpm), the torque produced by the motor drops logarithmically, and approaches zero at a maximum speed (around 3,000 rpm). The specific data can be found in the data sheet for the motor used.

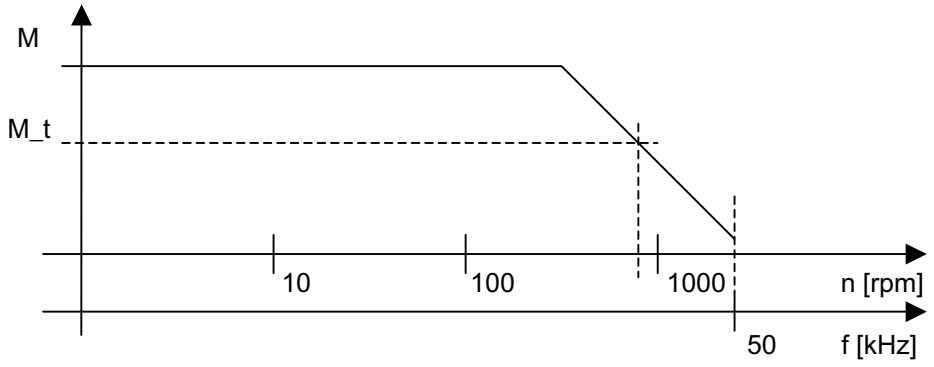


Figure 3-59 Example of a stepper motor torque characteristic

Consequences of operation in the overload range

If the stepper motor is ever unable to produce the requested torque, it loses synchronization with the predefined frequency and its speed drops suddenly. This can lead to standstill.

In this state, motion cannot be resumed unless a setpoint of 0 is entered in the meantime.

For a position axis without additional encoders, the traversing position and, as a result, the synchronization of the axis are lost.

Avoiding operation in the overload range

When the axis is designed, a maximum motor speed should be determined using the torque M_t required by the process. This maximum speed corresponds to the maximum velocity of the axis. The maximum frequency of the stepper motor must not be exceeded.

3.11.18 Encoder signal output (V4.0 or later)

The real axis with the setting **typeOfAxis**=REAL_AXIS_WITH_SIGNAL_OUTPUT can be used to output encoder signals via the SINAMICS TM41 module.

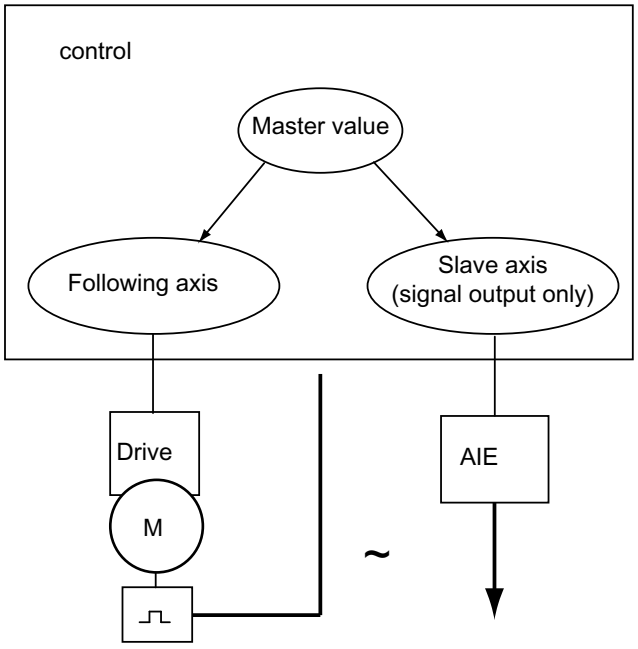


Figure 3-60 Using the axis for encoder signal output

Application

The axis position (a master value) is to be made available to a second control unit as an encoder signal via encoder signal simulation.

The standard message frame 3 is set between control unit and drive. An offset for the zero pulse output can be specified in the application.

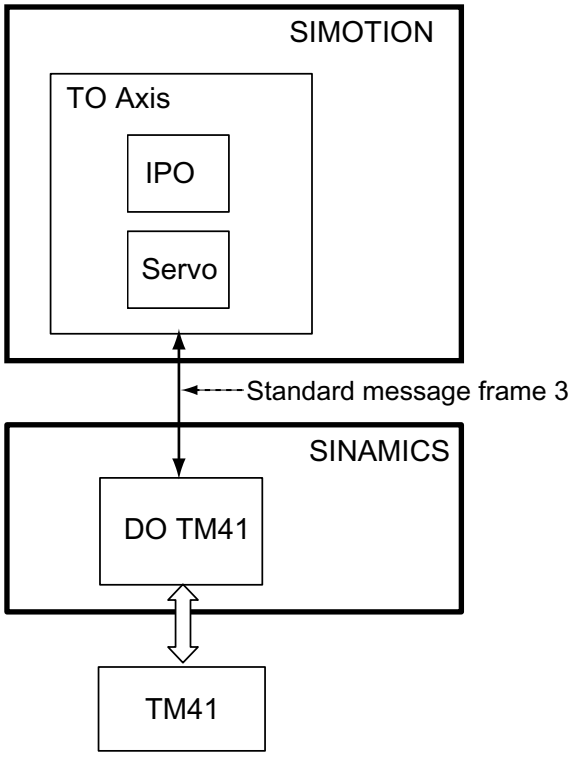


Figure 3-61 Interface of TM41 / DO41 to Axis technology object

In order to rule out lasting position deviations in the output signal, the axis can be provided with a PI controller that is applied to the actual position returned by TM41 / DO41.

Setting as an axis with signal output via the TM41 module only

Setting the axis as an axis with signal output only via the enumerator element **REAL_AXIS_WITH_SIGNAL_OUTPUT in TypeOfAxis**.

The following functions are not practicable or cannot be activated with this setting:

- Active homing is not permitted
- Measurement via digital drive is not permitted
- Compensations are disabled
- Following error monitoring functions are disabled
- Following error monitoring is disabled
- Positioning monitoring is deactivated
- Standstill monitoring is disabled
- Hardware limit monitoring is disabled

Note

You can also find further information under *FAQs* in the *SIMOTION Utilities & Applications* provided with SIMOTION SCOUT.

See also

Setting as a real axis with encoder signal simulation (V4.0 and higher) (Page 59)

3.12 Commissioning the position controller of positioning axes

3.12.1 Overview of commissioning the position controller of positioning axes

Basic procedure

When commissioning the position controller, you should follow a basic procedure, which is outlined below. You can find detailed information in subsequent chapters.

- Enter/check the settings for the controller type, load gear, measuring gear, normalization of the speed setpoint interface, leadscrew pitch, encoders, and valve characteristic (for hydraulic axes)
- Set the speed controller on the drive
- On the SIMOTION control unit, preselect the axis data set (Page 200) whose controller parameters are to be optimized (and, if necessary, preselect the appropriate data set on the drive)
- Set the controller parameters
  The SIMOTION measuring functions (Page 238) for manual optimization and the automatic controller setting (Page 230) are available for this.
- Set the equivalent time constant of the position controller
- Non-volatile storage of parameters on the target device and in the offline project

You can also find a commissioning guide under *FAQs* in the *SIMOTION Utilities & Applications* provided with SIMOTION SCOUT.

See also

Overview of automatic controller setting (as of V4.1 SP1) (Page 230)

SIMOTION measuring functions (Page 238)

Axis control panel (Page 243)

Data sets (Page 200)

3.12.2 Configuration data

Configuration data in the axis data set

The configuration data of a data set is contained in the structure **TypeOfAxis.NumberOfDataSets.DataSet_n** (n: data set number).

Table 3-28 Configuration data in the axis data set

Parameter	Configuration data element
General position control parameters	
Controller type	controllerStruct.conType
VTC Equivalent time of the speed control loop (time constant of the balancing filter). With the MODE_2 setting in the balancing filter, VTC is effective with a maximum value of 16 servo cycle clocks.	• Electric axis Axis data set: dynamicData.velocityTimeConstant • Hydraulic axis controllerStruct.DynamicQFData.qOutputTimeConstant
PTC Equivalent time of position control loop	• Electric axis dynamicData.positionTimeConstant • Hydraulic axis dynamicQFData.positionTimeConstant
Proportional-action position controller with precontrol for electric drives (controller type = PV)	
Balancing filter type	controllerStruct.PV_Controller.balanceFilterMode
Activation of Dynamic Servo Control (DSC)	controllerStruct.PV_Controller.enableDSC
Activation of DSC with spline	controllerStruct.PV_Controller.enableDSCSpline
Kpc Weighting of velocity precontrol	controllerStruct.PV_Controller.kpc
Kv P controller gain	controllerStruct.PV_Controller.kv
Activation of velocity precontrol	controllerStruct.PV_Controller.preCon
PID position controller with/without actual-value-dependent D component for hydraulic drives (controller type = PID/ PID_ACTUAL)	
Balancing filter type	controllerStruct.PID_Controller.balanceFilterMode
Activation of the integrator limitation.	controllerStruct.PID_Controller.enableAntiWindUp
Delay time of DT1 element	controllerStruct.PID_Controller.delayTime
Kd D-component gain	controllerStruct.PID_Controller.kd
Ki I-component gain	controllerStruct.PID_Controller.ki
Kp P-component gain	controllerStruct.PID_Controller.kp
Kpc Weighting of velocity precontrol	controllerStruct.PID_Controller.kpc
Activation of velocity precontrol	controllerStruct.PID_Controller.preCon

Parameter	Configuration data element
Mechanics	
Load gearbox	• Load revolutions Gear.denFactor • Motor revolutions Gear.numFactor

Table 3-29 Configuration data that is not included in the axis data set

Parameter	Configuration data element
Normalization of drive interface (electric drive)	
Maximum speed of rotation (rotary drive)	TypeOfAxis.setPointDriverInfo.DriveData.maxSpeed
Maximum traversing velocity (linear drive)	TypeOfAxis.setPointDriverInfo.LinearMotorDriveData.maxSpeed
Normalization speed (rotary drive)	TypeOfAxis.setPointDriverInfo.DriveData.nominalSpeed
Reference velocity (linear drive)	TypeOfAxis.setPointDriverInfo.LinearMotorDriveData.nominalSpeed
Reference for normalization speed (rotary drive)	TypeOfAxis.setPointDriverInfo.DriveData.speedReference
Reference for reference velocity (linear drive)	TypeOfAxis.setPointDriverInfo.LinearMotorDriveData.speedReference
Mechanics	
Leadscrew pitch	leadScrew.pitchValue
Fine interpolation	
Fine interpolator in the servo cycle clock	Fineinterpolator._type
Encoder	
Encoder parameters	The encoder settings are contained in the encoder data set: TypeOfAxis.NumberOfEncoder.encoder_x (x: Number of the encoder data set)

3.12.3 Example of commissioning a proportional-action controller with precontrol

The procedure for determining the control parameters of an electric axis manually is outlined below using a P controller with precontrol (PV controller) as an example.

This procedure is the same whether or not the DSC setting is used.

Requirements

- The settings have been made for the controller type (PV controller), velocity precontrol, load gear, measuring gear, normalization of the speed setpoint interface, leadscrew pitch, and encoders.
- The fine interpolator should be set to cubic (continuous acceleration) or quadratic (continuous velocity) interpolation in order for the velocity precontrol to provide continuous values in the acceleration and deceleration phases.

- The axis data set whose control parameters are to be determined is active (if necessary, the relevant data set on the drive must also be selected).
- Monitoring and compensation functions are deactivated
  - Drift and friction compensation
  - Dynamic monitoring of following errors
  - Positioning monitoring

Checking the interface to the drive

Before performing the actual axis optimization, you should check whether the interface between the SIMOTION control and the drive has been configured correctly.

The following check is useful for this purpose:

The axis is traversed at constant velocity, and the control errors are recorded with the SIMOTION trace function in SCOUT. For this check, the servo gain factor Kv should be set to a low value (e.g. default value: Kv = 10/s).

-> Control error: **servoData.controllerDifference**

If the axis is operated with 100-percent velocity precontrol, the control error in the steady-state condition must be zero on average during the constant motion phase.

If velocity precontrol is not in effect, the control error in the steady-state condition is calculated as follows:

Control error = set velocity/Kv

If another control error results, then the configuration of the interface between the SIMOTION control and the drive is faulty.

A common cause for the faulty behavior is a discrepancy in the normalization speed configuration between the SIMOTION control and the drive. Check the following setting:

Scenario 1: **TypeOfAxis.setPointDriverInfo.DriveData.speedReference** = MAX_VALUE

In this case: drive normalization speed =
TypeOfAxis.setPointDriverInfo.DriveData.maxSpeed

Scenario 2: **TypeOfAxis.setPointDriverInfo.DriveData.speedReference** = NOMINAL_VALUE

In this case: drive normalization speed =
TypeOfAxis.setPointDriverInfo.DriveData.nominalSpeed

Note

Drives are typically linked to the SIMOTION control with PROFIdrive (via PROFIBUS/PROFINET). The parameters for setting the normalization speed for the SINAMICS, MASTERDRIVE MC, and SIMODRIVE 611U drive systems are listed below.

For SINAMICS, the normalization speed can be accepted by the drive in the axis wizard; as of V4.2, it is adapted during runtime. See Coupling of digital drives (Page 56).

Table 3-30 Parameters for setting the normalization speed

Drive system	Parameter
SINAMICS	P2000
MASTERDRIVES MC	P353
SIMODRIVE 611U	P880

Determining the controller parameters

The following controller parameters of the active axis data set should be determined:

Kv P controller gain	**controllerStruct.PV_Controller.kv**
VTC Equivalent time of the lower-level drive (time constant of the balancing filter)	**dynamicData.velocityTimeConstant**
PTC Equivalent time of the position control loop	**dynamicData.positionTimeConstant**

Two main optimization goals can be differentiated:

1. High dynamic response for continuous movements and synchronous operation groups. The axis is allowed a certain amount of overswing.
2. High dynamic response for discontinuous movements, i.e. fast, time-optimized positioning movements without overswings.

As first step, the automatic speed optimization and subsequently the automatic position optimization (if DSC possible) are performed.

This optimization provides very good results even for the above-mentioned strategy.

The optimization of movements without overswing is described next.

Prerequisite:
The speed controller of the drive has already been set as (nearly) overshoot-free. In principle, this can be achieved by using a low-pass filter as the speed setpoint filter or through the use of a system model (reference model) for the I-component of the speed controller. For more information, refer to the relevant drive documentation.

Determining the servo gain Kv of the P-controller

Disable the velocity precontrol and balancing filter in the configuration data of the axis.

controllerStruct.PV_Controller.kpc = 0
dynamicData.velocityTimeConstant = 0

Position the axis cyclically with a high deceleration rate and trapezoidal velocity profile and trace the system variables for the set and actual positions of the axis with SIMOTION trace in SCOUT. Make sure that the axis reaches the constant velocity phase during this operation. You must also select the deceleration such that the current or torque in the drive is not limited.

Set position: **servoData.symmetricCommandPosition**
Actual position: **servoData.actualPosition**

Starting with Kv = 10, continue increasing the servo gain factor in five-percent increments as long as the actual value does not overshoot or undershoot when entering the target position.

Deduct 10 percent from this maximum Kv value and make this setting on the target device.

Setting the balancing filter

When velocity precontrol is used, the entry behavior during positioning and the control errors in the acceleration and deceleration phases can be influenced by the time constant of the balancing filter (VTC).

VTC: **dynamicData.velocityTimeConstant**

The velocity precontrol must be activated to determine the time constant.

controllerStruct.PV_Controller.kpc = 100
controllerStruct.PV_Controller.preCon = YES

Position the axis cyclically with a high deceleration rate and record the control error with SIMOTION trace in SCOUT. Make sure that the axis reaches the constant velocity phase during this operation. You must also select the deceleration such that the current or torque in the drive is not limited.

Control error: **servoData.controllerDifference**

Continue increasing the time constant of the balancing filter (VTC) incrementally until the axis enters the target position without overshooting or undershooting.

If you have to set a very large time constant to avoid overshooting/undershooting, and if the axis "creeps" to its target position, you may be able to achieve a smoother setpoint curve by using the SMOOTH velocity profile. In this case, you must limit the jerk in the motion command for entry in the target position and then repeat the balancing filter optimization.

If this still does not yield the desired result, you should check the drive setting again. It could be that the drive setting has not yet been optimized to be overshoot-free.

You may want to allow a small overshoot/undershoot upon entry in the target position to enable a faster settling behavior.

For axes with different balancing filter settings in a synchronous operation group, a dynamic response adaptation is desirable, see Dynamic response adaptation (Page 129).

Setting the equivalent time of the position controller

To apply the preassigned axis stop ramp (in the event of an error, for example), the system requires the equivalent time constant of the position controller (PTC).

PTC: **dynamicData.positionTimeConstant**

In this example, this time constant is derived from the time constant of the balancing filter (VTC), due to the use of velocity precontrol (at 100%). The following relation applies:

PTC = VTC

If the velocity precontrol is not activated, the following is true for PTC:

PTC = 1/Kv

Storing the control parameters

If the control parameters have been entered in the corresponding configuration data in the target device, they must now be stored as non-volatile data so that they will still be available the next time the control is powered up. The following online functions are available for this purpose under the target device in SCOUT.

Table 3-31 Online storage functions

Function	Comments
Copy ACTUAL to RAM	Copies the configuration data to the RAM memory on the target device
Copy from RAM to ROM	Non-volatile storage of configuration data on memory card

In order to transfer modified configuration data to the configuration, you must select the **Load configuration data to PG** function under the target device in SCOUT and then save the project.

After you have optimized the position controller, you must reset the compensations and monitoring functions listed under "Requirements", if applicable.

3.13 Command variable calculation

3.13.1 Velocity profiles

Specifications of velocity changes for axes (approaching, stopping, other velocity changes) are applied via **velocity profiles**.

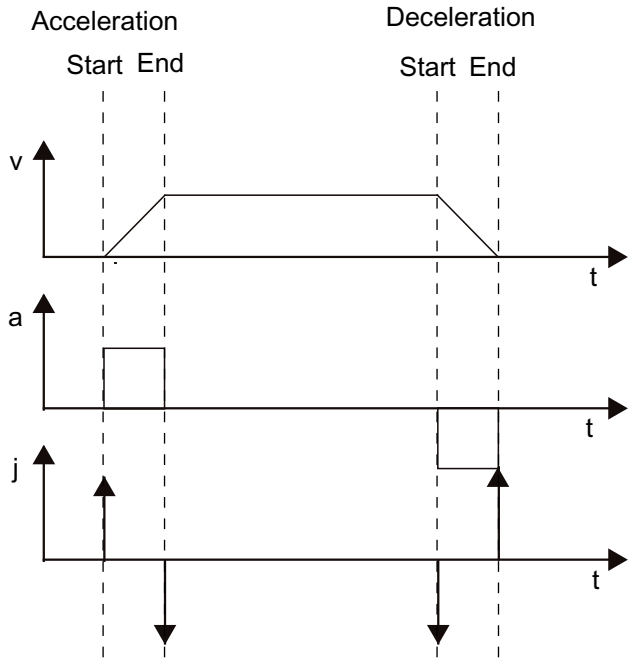
The velocity profile defines the behavior of the axis during startup and when braking, and for velocity changes.

The following velocity profiles are available:

- **Trapezoidal** velocity profile (**TRAPEZOIDAL**)
  The trapezoidal profile is set for constant acceleration and deceleration of motion in a positive and negative direction.
- **Smooth** velocity profile (**SMOOTH**)
  This profile is used for a continuous acceleration and deceleration. The jerk can be set.

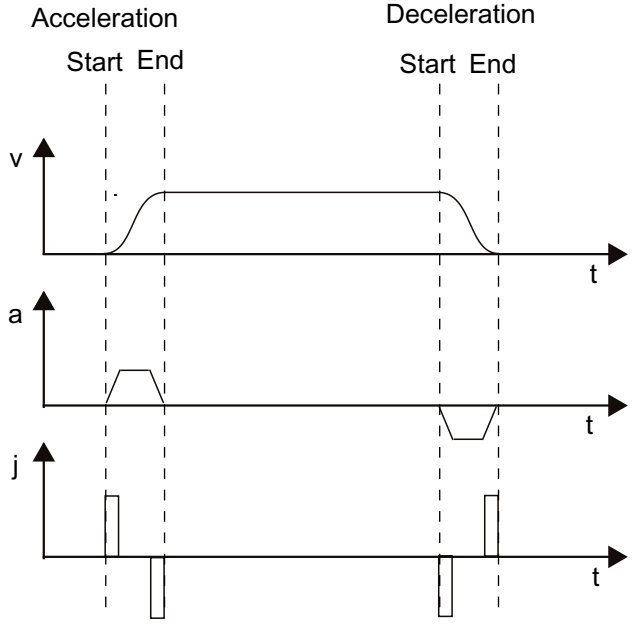
The parameters for profile, velocity, acceleration, and jerk can be assigned in the command, or the **userDefault** value settings are used.

The parameter for the velocity profile is specified via the motion command (velocityProfile) or stored as a default value in the userdefaultdynamics.profile system variable.



v - velocity
a - acceleration
j - jerk
t - time

Figure 3-62 Trapezoidal velocity profile



v - velocity
a - acceleration
j - jerk
t - time

Figure 3-63 Smooth velocity profile

Continuous acceleration curve for smoothing

Prior to SIMOTION V4.4., the following applies to single axis motions and path motions when a continuous acceleration curve is specified: When blending to the motion of the new command on transition, acceleration is zeroed, i.e. the deceleration is canceled via the jerk.

As of SIMOTION V4.4, it is possible to set that the acceleration is not zeroed. See also Positioning with blending (Page 164).

Continuous acceleration curve for reversing

For single axis movements, for path movements, for the synchronization of synchronization movements and for the specification of a continuous acceleration curve, for any required reversing in the reversal point of the movement direction, the acceleration is made to zero. This means the acceleration/deceleration will be removed using the jerk and re-established after the reversal using the jerk.

This does not apply for motion specification via user-defined profiles or for the procedure based on motion vectors (MotionIn interface).

Setting the acceleration to zero when stopping or substituting a motion

When stopping with the **_stop()** and **_stopEmergency()** commands or substituting a motion with the **_pos()** and **_move()** commands, an existing acceleration or deceleration can be set immediately to zero. The existing acceleration or deceleration is not first reduced via the jerk.

The abortAcceleration function parameter is available for the stop commands as of V4.2.1 and the substitution commands as of V4.4.

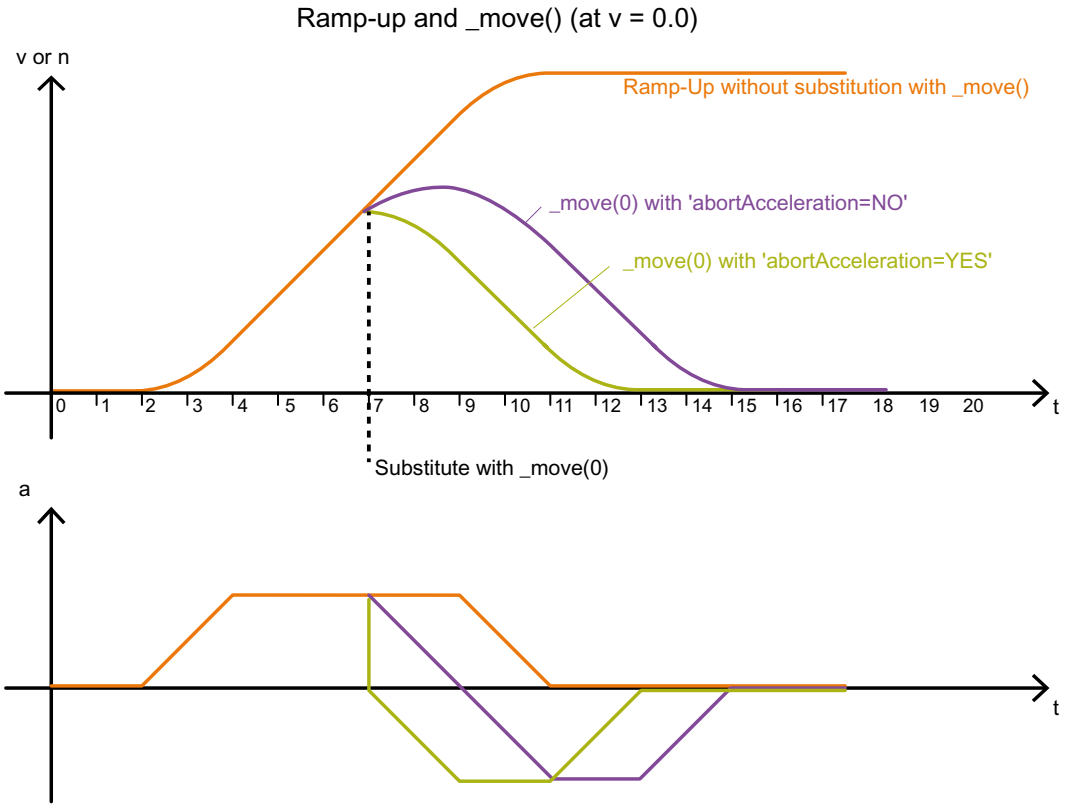


Figure 3-64 Velocity profile on ramp-up with and without abortAcceleration

3.13.2 Defining accelerations and decelerations

The axis acceleration and deceleration parameters can be set to take effect in relation to the direction or state using the **decodingConfig.directionDynamic** configuration data element.

Parameters for direction-related setting

- **positiveAccel:** Acceleration in the positive direction of motion and deceleration in the negative direction of motion
- **negativeAccel:** Acceleration in the negative direction of motion and deceleration in the positive direction of motion

The option to assign parameters for a direction-dependent dynamic response is relevant, for example, for applications such as vertical axes.

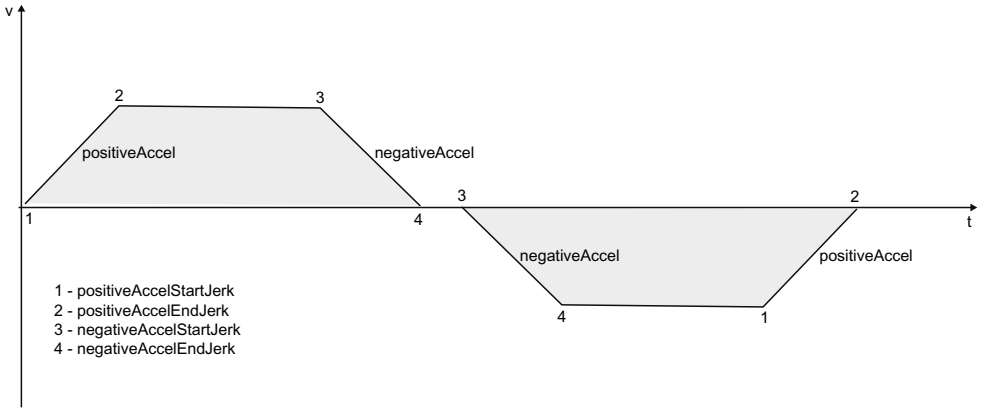


Figure 3-65 Operative points for acceleration and jerk parameters

Parameters for state-dependent setting

- **positiveAccel:** Acceleration of axis motion, irrespective of the direction of motion
- **negativeAccel:** Deceleration of axis motion, irrespective of the direction of motion

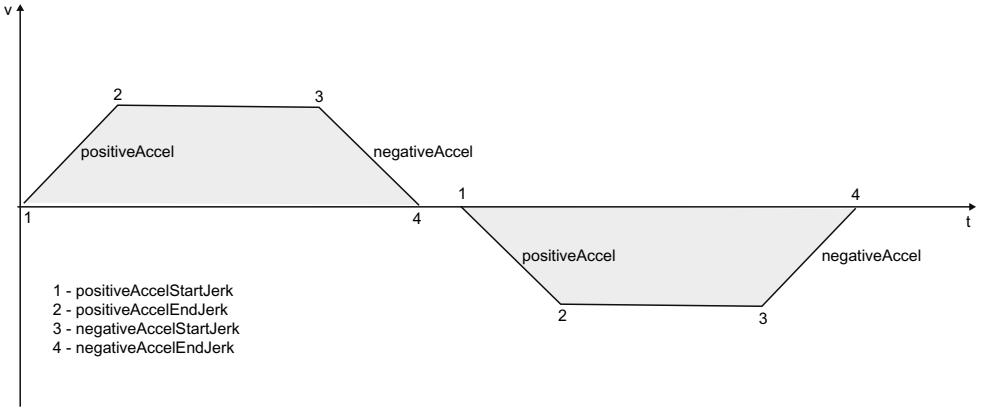


Figure 3-66 Operative points for acceleration and jerk parameters

Values set by the command or set by means of a system variable

- velocity
- positiveAccel
- negativeAccel
- positiveAccelStartJerk
- positiveAccelEndJerk
- negativeAccelStartJerk
- negativeAccelEndJerk

The dynamic response parameters are set with parameters in the motion command or stored as default values in the system variables of the userdefaultdynamics structure.

3.13.3 Override

Factors can be superimposed online on the current traversing velocity or acceleration/ deceleration. The velocity override is applied to the velocity, and the acceleration override is applied to the acceleration and deceleration.

The override values can be set and read via **system variables**.

The override for the **velocity** can be set from **0** to **200%**.

The override for the **acceleration/deceleration** can be set from **1%** to **1000%**.

System variables

- override.velocity
- override.acceleration

3.13.4 Default settings for dynamic response parameters

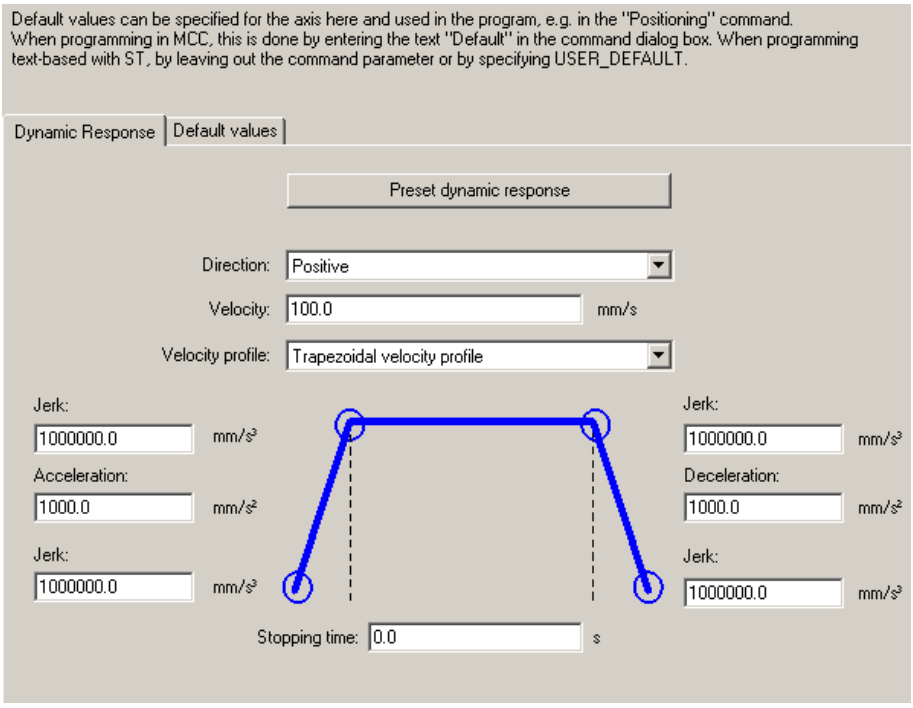


Figure 3-67 Default settings for dynamic response parameters

Stop time

This time becomes active if the default value is used for the time with **_stopEmergency()** and the **stopDriveMode=**STOP_IN_DEFINED_TIME setting.

Dynamic response default setting

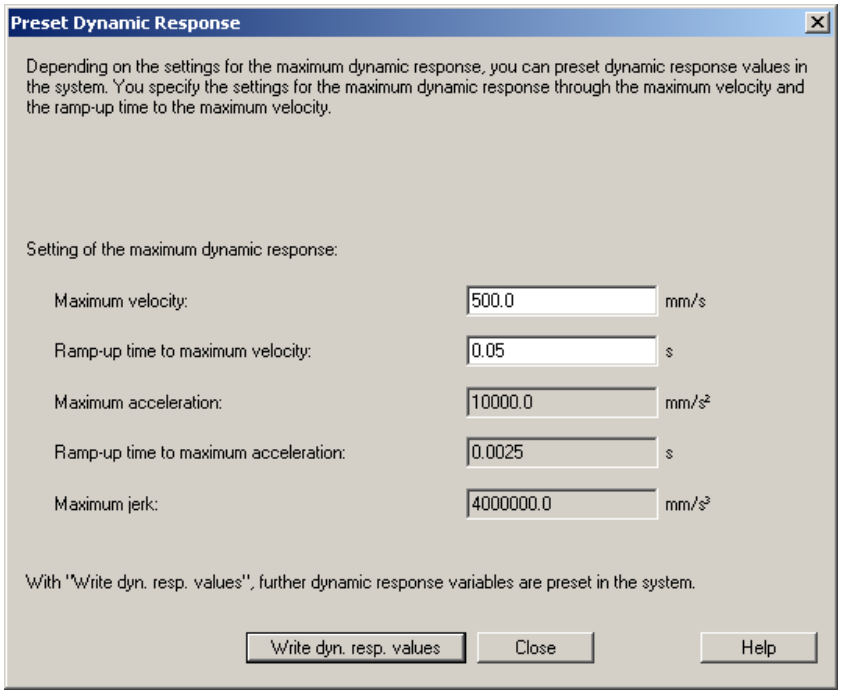


Figure 3-68 Dynamic response default setting

For the selection of the screen form, the **maximum velocity**, **maximum acceleration** and **maximum jerk** fields will be displayed using the setting in the associated configuration date. This provides the **startup time to the maximum velocity** and **startup time to the maximum acceleration**.

If the **maximum velocity** or the **startup time to the maximum velocity** is changed, the dependent values will be recalculated and displayed. The configuration data will be written directly and remain at the changed values also for **close** without **write dynamic values**.

The **startup time to the maximum velocity** will be used as startup time assuming the constant maximum acceleration over this time, i.e. without considering the time for building and removing the acceleration because of the maximum jerk.

Write dyn. resp. values writes corresponding values to the configuration data and adapts system variables, see Dynamic limiting functions (Page 160).

See also

Velocity profiles (Page 153)

Defining accelerations and decelerations (Page 156)

Defaults (Page 84)

3.13.5 Dynamic limiting functions

Maximum values

The properties of the drive and the mechanical system produce the maximum values for velocity, acceleration and jerk. The values are set in the **maxVelocity**, **maxAcceleration**, and **maxJerk** variables.

Technological maximum values

For programming, additional dynamic limit values are available in the **plusLimitsOfDynamics** and **minusLimitsOfDynamics** system variables. These can be read and written in the program.

The technological maximum values are state-dependent, i.e. are not a direction-dependent setting:

- **plusLimitsOfDynamics.velocity**
  Technological limit value for the velocity of the axis independent of the direction of motion
- **plusLimitsOfDynamics.positiveAccel**
  Technological limit value for the acceleration of the axis independent of the movement direction
- **plusLimitsOfDynamics.negativeAccel**
  Technological limit value for the deceleration of the axis independent of the movement direction
- **plusLimitsOfDynamics.positiveAccelJerk**
  Technological limit value for the jerk for establishing the acceleration and removing the deceleration independent of the movement direction
- **plusLimitsOfDynamics.negativeAccelJerk**
  Technological limit value for the jerk for removing the acceleration and establishing the deceleration independent of the movement direction

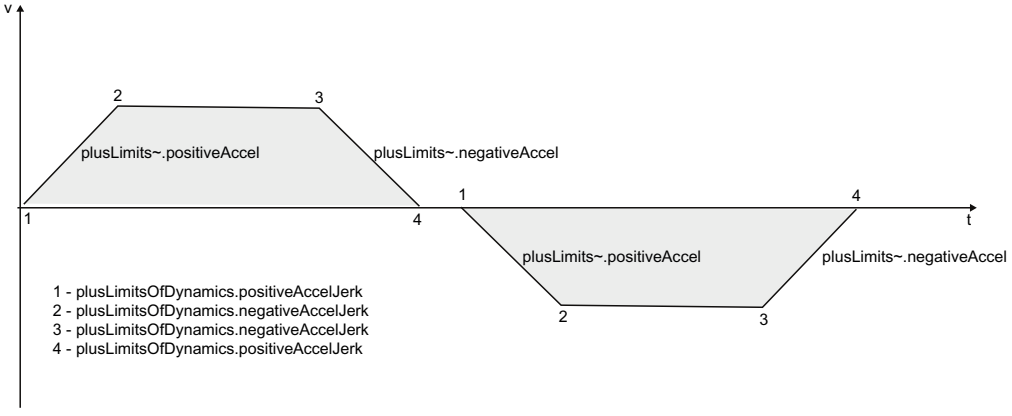


Figure 3-69 Technological maximum values for non-direction-dependent setting

Technological maximum values for direction-dependent setting:

- **plusLimitsOfDynamics.velocity**
  Technological limit value for the velocity of the axis independent of the direction of motion
- **minusLimitsOfDynamics.velocity**
  Technological limit value for the velocity of the axis independent of the direction of motion

- **plusLimitsOfDynamics.positiveAccel**
  Technological limit value for the acceleration of the axis in the positive movement direction
- **plusLimitsOfDynamics.negativeAccel**
  Technological limit value for the deceleration of the axis in the positive movement direction
- **minusLimitsOfDynamics.positiveAccel**
  Technological limit value for the acceleration of the axis in the negative movement direction
- **minusLimitsOfDynamics.negativeAccel**
  Technological limit value for the deceleration of the axis in the negative movement direction
- **plusLimitsOfDynamics.positiveAccelJerk**
  Technological limit for the jerk for establishing the acceleration and removing the deceleration in the positive movement direction
- **plusLimitsOfDynamics.negativeAccelJerk**
  Technological limit for the jerk for removing the acceleration and establishing the deceleration in the positive movement direction
- **minusLimitsOfDynamics.positiveAccelJerk**
  Technological limit for the jerk for establishing the acceleration and removing the deceleration in the negative movement direction
- **minusLimitsOfDynamics.negativeAccelJerk**
  Technological limit for the jerk for removing the acceleration and establishing the deceleration in the negative movement direction

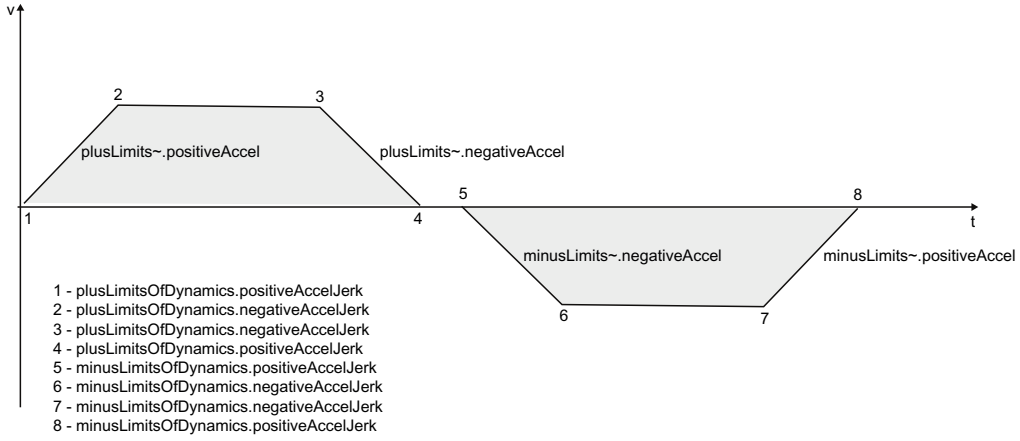


Figure 3-70 Technological maximum values for direction-dependent setting

The minimum from the maximum value of the axis and the set technological maximum value for the associated dynamic response quantity are used.

For system created movements with maximum dynamic response values, e.g. travel to the software limit switch, the minimum from the maximum value and the technological maximum value is used for the corresponding dynamic response quantity.

Active dynamic limitation

- Enter the maximum velocity in **maxVelocity**. This must always be less than or equal to the velocity that is physically possible. The greater the difference between the two, the larger the available control margin.
  Please note that the gear is data-set-dependent. Maximum velocity is calculated per data set. The maximum across all data sets is taken to be the maximum velocity.
- The effective limit is always the minimum of the maximum value and the technological maximum value.
- The **maxJerk** limiting is only monitored for motions in jerk-controlled mode or motions with continuous acceleration.

Dynamic limitation with synchronous operation relationships

If the axis provides a master value for a synchronous operation relationship, the maximum velocity is also limited to less than half the modulo range/IPO cycle clock (maximum possible master value velocity of the axis).

It is the minimum value from the following that applies when calculating the setpoint value for the following axis:

- Maximum velocity (**maxVelocity**) of the following axis
- Maximum technological velocity (**plusLimitsOfDynamics**/**minusLimitsOfDynamics**)
- Maximum master-value velocity of the axis

If a velocity faster than this maximum velocity is specified, the technological alarm 40002 "velocity will be limited" will be generated and the set velocity adapted appropriately.

Dynamic limitation with active ramp calculation in the drive

Please note that when linking with digital drives and with a ramp calculation for the speed active in the drive, additional dynamic limitation may occur, which is not taken account of separately in the motion control and motion monitoring functions performed in the control.

The operating mode is displayed in the drive's PROFIdrive parameter R0930 (PROFIdrive operating mode):

1. Speed-controlled operation with ramp-function generator
2. Position-controlled operation
3. Speed-controlled operation without ramp-function generator

SINAMICS drives:

- Drive object type vector
  Default setting *1. Speed-controlled operation with ramp-function generator*
  The motion dynamic response is, therefore, limited in the drive, if applicable.
- Drive object type servo
  Default setting *3. Speed-controlled operation without ramp-function generator*
  The motion dynamic response is not limited in the drive.

When using vector drives, therefore, you must check that the ramp-up/ramp-down time parameterized for the drive matches the configured axis acceleration/deceleration.

3.13.6 Stopping with preassigned braking ramp

The deceleration set in the **emergencyRampGenerator.maxDeceleration** configuration data element is used to stop an axis with a preconfigured braking ramp.

The current velocity is brought to zero at the assigned deceleration rate.

With position-controlled traversing, the position controller continues to be active during the stopping procedure.

For position-controlled, and thus position-related, traversing, the application point of the actual position value is extrapolated with the current velocity and the equivalent time of the position control loop set in the **dynamicData.positionTimeConstant** configuration data element.

Therefore, the application point can only be calculated correctly if the equivalent time of the position control loop is set correctly.

The preassigned braking ramp is effective in the following cases:

- During stopping with the **_stopEmergency()** command and the **STOP_WITH_COMMAND_VALUE_ZERO** setting
- During the **FEEDBACK_EMERGENCY_STOP** alarm response
- When program simulation is enabled during axis motion using the **_enableAxisSimulation()** command

3.13.7 Traversing the axis via velocity specifications

The axis can be traversed via velocity specifications. A position axis can be traversed either via a velocity setpoint specification or in position-controlled mode.

The following is specified for the axis motion:

- Direction
- Dynamic response parameters
- Optionally, the duration of the constant motion phase

Parameters

- Direction
  The direction of motion is defined using either the sign of the velocity specification or a specific parameter.
- Velocity

- Duration of constant motion phase
  The duration of the constant motion phase is an optional parameter. If the constant motion phase is not specified, the **_move()** command triggers a continuous motion that must be overridden by a **_stop()** command or another motion command.
- Dynamic response parameters

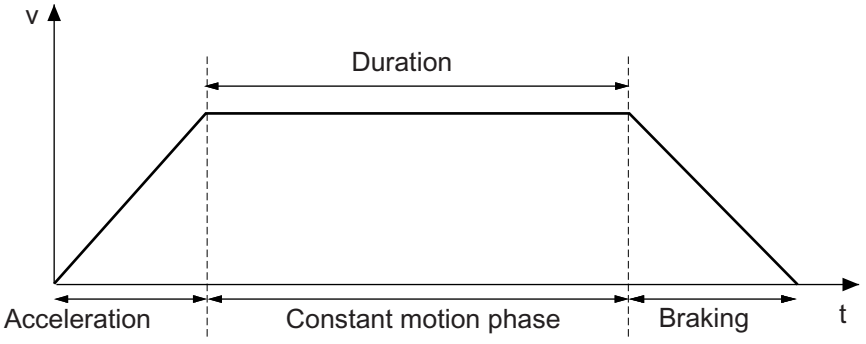


Figure 3-71 Phases for moving with _move()

An acceleration phase or a deceleration phase can be ended with jerk control using the **velocityType=RESULTING** parameter. The resulting velocity is then retained.

See also

Velocity profiles (Page 153)

Defining accelerations and decelerations (Page 156)

3.13.8 Positioning

The axis is moved to a target position via a parameterizable velocity profile. The entry in the target position is monitored.

See also

Positioning and standstill monitoring (Page 102)

3.13.9 Positioning with blending

Blending is a special way of linking the positioning motion programmed in the command to a previous positioning motion. Unlike overriding, the previous motion command is completed up to the target position, where the transition occurs. Blending takes place between two positioning commands. The set velocity specified in the commands for each motion is never violated.

The previous positioning motion is performed at the velocity specified in the command up to the target position. **Exception:** If the velocity of the new positioning motion is less than the set velocity of the previous motion and both motions have the same sign, then the velocity of the previous positioning motion is reduced to the velocity of the new motion by the time the target position is reached.

In versions earlier than SIMOTION V4.4, the dynamic response is planned via two traversing blocks.

As of SIMOTION V4.4, the dynamic response is planned via three traversing blocks (current, next and next but one) for a newly created axis. It is therefore possible to plan a higher velocity if necessary for axis motion in the same direction. The setting is saved in the **DecodingConfig. commandsForAxisDynamics** configuration data element.

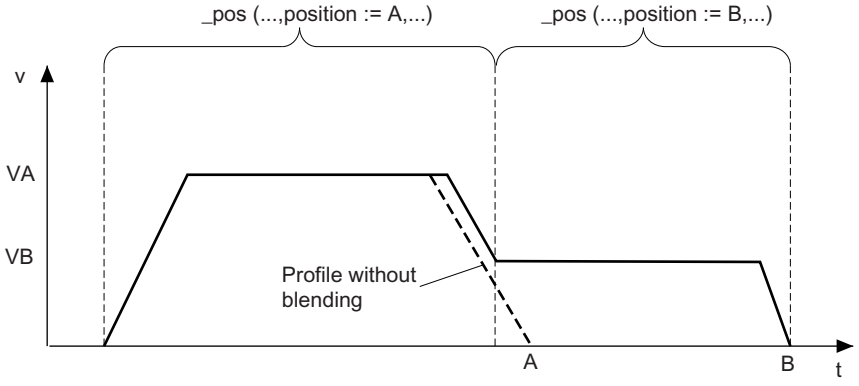


Figure 3-72 Blending when velocity of the subsequent motion is lower

If the direction is reversed, the previous positioning motion is slowed down at the target point, and transition to the new motion takes place immediately.

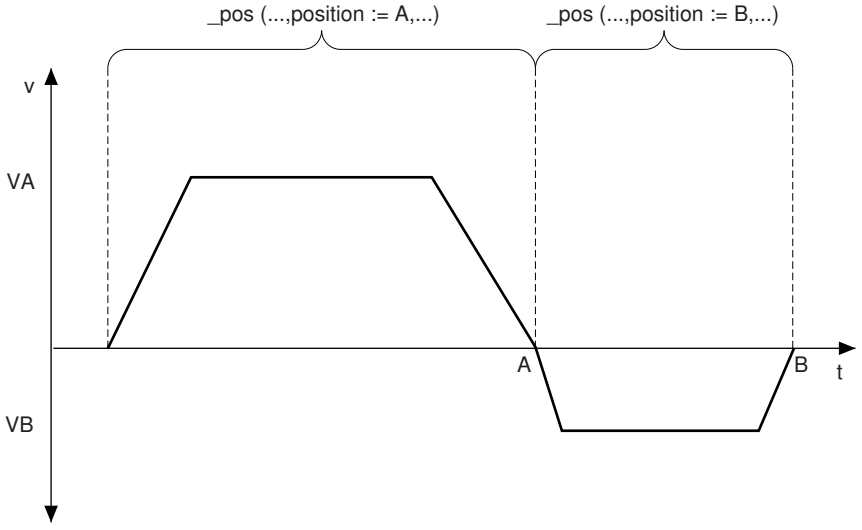


Figure 3-73 Blending when the subsequent motion reverses direction

If the absolute value of the velocity of the new command is greater than the velocity of the current motion, the velocity is increased after the transition to the new command, that is, after the previous target position is reached.

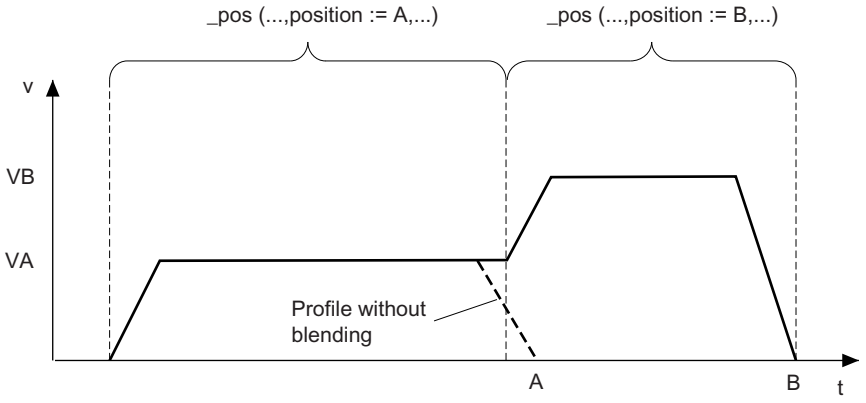


Figure 3-74 Blending when velocity of the subsequent motion is higher

Active blending requires **mergeMode**= NEXT_MOTION or SEQUENTIAL in the command where blending is to take place, plus timely decoding of the command. If the command in which blending is to be carried out is not known to the interpolator at the deceleration starting point of the current motion, the braking ramp is traveled.

The braking distance is always adhered to, even if the path length of the subsequent motion command is shorter than the required braking distance.

Acceleration at the command transitions

Versions before SIMOTION V4.4 are limited by the fact that acceleration is zeroed on blending with programmed, constant acceleration curve (SMOOTH) at the transition to the new command.

As of SIMOTION V4.4, it is possible to use the **DecodingConfig.blendingAcceleration** configuration data element to set that the acceleration at the transition to the new motion command is not zeroed if not necessary for the overall motion. In a newly created axis, WITHOUT_REDUCTION is the default.
When reversing, the acceleration is always zeroed regardless of this setting. Reversing here means that the direction of motion of the axis changes at the blending point.

3.13.10 Superimposed positioning

Superimposed positioning is performed in the superimposed coordinate system of the axis. The target position can be specified as an absolute or relative position in this coordinate system. (For feedback of the target position, see **Superimposed motion**)

The superimposed motion has its own programmable dynamic response parameters. The **userDefault** settings of the dynamic response parameters are identical to the settings of the main motion. Blending conditions for the superimposed motion are ignored.

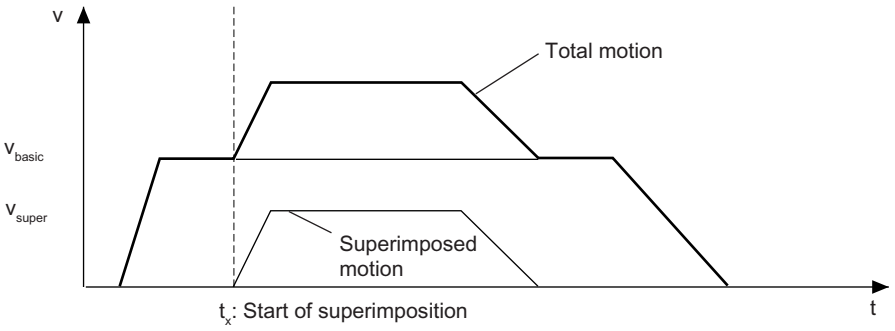


Figure 3-75 Superimposed positioning

See also

Superimposed motion (Page 169)

3.13.11 Traversing with specific motion profiles

In addition to traversing/positioning via system functions, the axis can also be traversed via user-defined profiles/specific profiles.

See also

Overview of traversing with user-defined motion and force/pressure profiles (Page 203)

3.13.12 Traversing according to motion vectors (V3.2 and higher)

The axis can be traversed directly in accordance with the specification on the MotionIn interface.

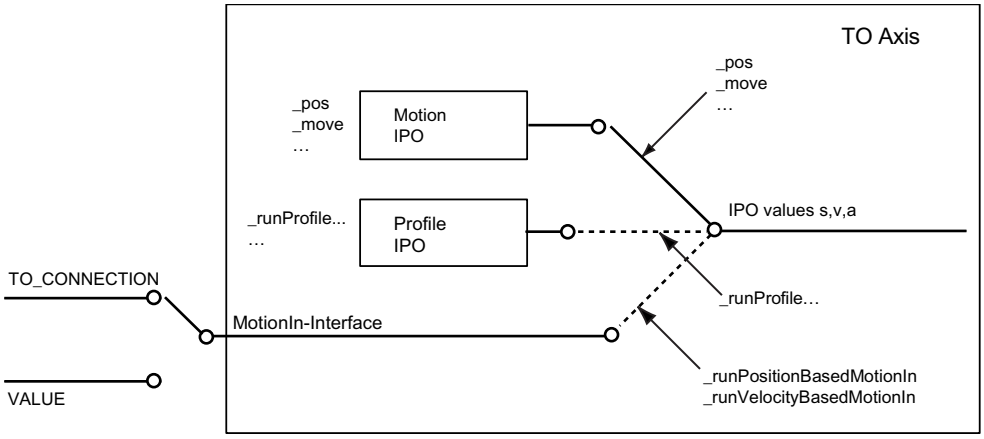
The MotionIn values can be directly picked up by another technology object, e.g., axis, external encoder, adding object, formula object, fixed gear. Alternatively, the MotionIn values can be specified via the default values from the user program.

The **_runPositionBasedMotionIn()** command activates the MotionIn interface with position-reference and **_runVelocityBasedMotionIn()** activates the MotionIn interface with velocity-reference. The MotionIn interface is replaced by motion commands, such as **_move()**, **_pos()**, **_stop()**, and **_stopEmergency()** or it can have an overriding effect itself.

Options for linking drive and position-controlled axes (position axes, following axes) to the motion vector:

- Linking position-controlled axis to motion vector (based on: position)
- Linking position-controlled axis to motion vector (based on: velocity) in position-controlled or velocity-controlled mode
  In this way, a position-controlled axis (position axis) is linked to a drive axis.
- Linking drive axis to motion vector (based on: velocity)
  This links a drive axis to a drive axis.

The MotionIn values can also be directly specified cyclically via the default variables.



TO_CONNECTION … Setpoint specified via TO interconnection
VALUE … Setpoint specified via cyclically active system variable

Figure 3-76 How the MotionIn interface works

Property

- Cyclical inclusion of MotionIn interface values
- Superimposition and overriding motion are possible
- Dynamic response parameters can be predefined in the commands for moving to the MotionIn values and for stopping
- The dynamic response can be limited via configuration data
- The MotionIn interface on the axis is multipoint-capable
- The reference TO is specified in the activation command
- It is then possible to switch to the reference TO

Note

Further information is available under Interconnection of technology objects in the Motion Control Basic Functions Function Manual.

3.13.13 Jerk limitation for local stop response (V3.2 or higher)

Local stop responses (stop responses triggered by alarm responses) may optionally be applied with or without rounding and jerk limiting.

The **decodingConfig.StopWithJerk** configuration data element can be used to specify whether the maximum jerk limitation of the Axis technology object is taken into account.

The jerk is only taken into account for IPO stops.

The jerk is determined from the minimum of the **plusLimitsOfDynamics**/ **minusLimitsOfDynamics** system variables and the **maxJerk** configuration data element.

3.14 Superimposed motion

Superimposed motion is defined with mergeMode=SUPERIMPOSED_MOTION_MERGE.

Superimposed motions are independent motions.

Superimposed motions cancel each other.

Superimposed motions can be stopped and continued separately.

Superimposed motions on a position axis are executed in the superimposed axis coordinate.

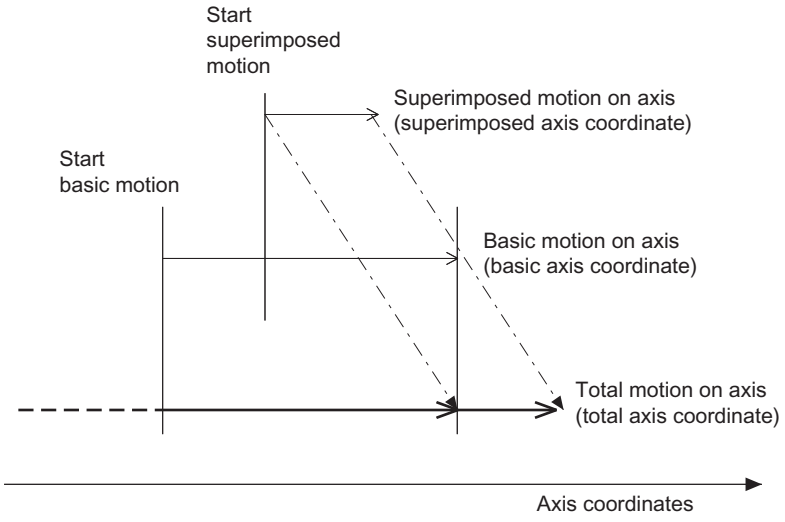


Figure 3-77 Functional diagram for motion transition SUPERIMPOSED_MOTION_MERGE

Commands that allow superimposed motions on the axis include:

_move()
_pos()
_runTimeLockedVelocityProfile()
_runTimeLockedPositionProfile()

Note

Only one superimposed motion can be active on the axis at a given time, e.g. superimposed positioning motion or superimposed synchronous operation.

Note

The dynamic limits always relate to the overall dynamic response. This may result in the superimposed motion having a dynamic response that is not as good as that defined in the specifications.

The superimposed motion is carried out in the superimposed axis coordinate as relative or absolute motion, depending on how it was programmed. Likewise, the basic motion of the axis is executed in the basic axis coordinate as an absolute motion or a relative motion, depending on the programming.

The timing of the feedback of the superimposed axis coordinate to the basic axis coordinate is set in the **decodingConfig.transferSuperimposedPosition** configuration data element; this

setting also specifies when superimposed motions are applied to the basic motion and, thus, when they are overridden. When the criterion becomes effective, the basic motion accepts the superimposed motion component.

Possible settings:

- Only on axis reset with _resetAxis()
- On axis reset and on each overriding command where mergeMode= IMMEDIATELY
- On axis reset, on each overriding command where mergeMode=IMMEDIATELY, and on each axis standstill motionStateData.motionStat=STANDSTILL

In addition, the superimposed axis coordinate/motion is fed back to the basic axis coordinate in the following cases:

- The axis switches into or out of follow-up mode.
- Switchover occurs between speed-controlled, position-controlled, and force-/pressure-controlled operation.
- For force-/pressure-controlled operation

Absolute and/or relative traversing is possible in the basic axis coordinate and in the superimposed coordinate. The total motion of the axis results from adding the position values, velocity values, and acceleration values of the basic motion and the superimposed motion together.

Notes on working with the superimposed axis coordinate

During active homing, a superimposed axis coordinate is always reset.

An active superimposed motion is aborted during active homing.

Display of current axis coordinates

The values/dynamic response vector of the basic axis coordinate and the superimposed coordinate are displayed on the axis.

These coordinates can be read via the following system variables:

- Total axis coordinates:
  positioningState.commandPosition: Set position (total)
  motionStateData.commandVelocity: Set velocity (total)
  motionStateData.commandAcceleration: Set acceleration (total)
- In basicMotion, the values of the basic axis coordinate are displayed on the axis.
  basicMotion.position
  basicMotion.velocity
  basicMotion.acceleration
- In superimposedMotion, the values of the superimposed axis coordinate are displayed on the axis.
  superimposedMotion.position
  superimposedMotion.velocity
  superimposedMotion.acceleration

3.15 Torque limiting via torque reduction

3.15.1 Overview of torque limiting via torque reduction

On the electric axis, torque limiting is available via a torque reduction. The limit value is specified in the command.

This function can be used with a drive axis, a position axis and a following axis. The accuracy depends on the drive used.

To be able to use the function, the PROFIdrive telegram type 10x (apart from telegram 101) must be set. It also requires the use of a drive that supports the function. As of V4.2, 10x telegrams are used with automatic telegram definition.

When the function is called, the desired torque (for rotary and linear axes) or the desired force (for linear axes only) is specified in the corresponding unit or as percentage relative to a reference value (**userDefaultTorqueLimiting.torqueLimit**). 0% means no torque on the drive and 100% means full torque on the drive.

The torque limiting is transferred to the drive as a torque reduction in the PROFIdrive telegram. If, for example, 80% is specified for the torque limiting, SIMOTION calculates the value 20(%) for the torque reduction for the drive and transfers this value to the drive via the PROFIBUS interface.

The limitation acts as an absolute value and thus has the same effect on both positive and negative torque.

If torque limiting is active, the following error monitoring and positioning monitoring functions are deactivated.

Active motion commands and synchronous operation relationships continue to be executed.

The limiting functions can be activated before a motion or simultaneously with a motion and can be switched by reissuing the command.

The position-related monitoring can remain active even with active torque limiting. For this purpose, there is the **TypeOfAxis.ServoMonitoring.motionMonitoringWhenTorqueLimiting** configuration data as of V5.1. When activated, this ensures that the position monitoring, positioning monitoring and/or following error monitoring that was active before the torque limiting remains active during the limiting.

As a result of torque limiting, for example, a greater difference between actual and target values can build up on the position-controlled axis, which can cause the axis to accelerate (in order to reduce the difference) even if the velocity calculated by the interpolator is now lower.

Please note that if an axis is ready as a result of active torque limits/torque reduction, the control difference to the position reference value is initially canceled and then the **_stop()** command is terminated, despite a position-controlled stop motion. Alternatively, you can enter a speed-controlled **_stop()** command.

If, for example, torque limiting is not required during the acceleration phase, the function must not be activated until after the acceleration phase; otherwise, the acceleration must be reduced.

Value specifications and transferring from the drive

The maximum values **SetpointDriverInfo.DriveData.maxTorque** and **SetpointDriverInfo.LinearMotorDriveData.MaxForce** are the reference values for the transmission of the obtained torque reduction to the drive and must be entered in the drive and in SIMOTION in accordance with the motor values.
With SINAMICS, the parameters are present in P1520, P1521, and P2003:
~.DriveData.maxTorque = P1520 = |P1521|
If the technology data block (Page 179) is interconnected, the following applies: **~.DriveData.maxTorque** = P2003

As of V4.2

The above still applies for the setting **SetpointDriverInfo.DriveData.TorqueReference** = MAX_VALUE or **SetpointDriverInfo.LinearMotorDriveData.ForceReference** = MAX_VALUE.

For SINAMICS and round-frame motors, and for the setting **SetpointDriverInfo.DriveData.TorqueReference** = NOMINAL_VALUE and **SetpointDriverInfo.DriveData.SpeedReference** = NOMINAL_VALUE, the following applies:

- **~.DriveData.nominalSpeed** = P2000
- **~.DriveData.MaxSpeed** = P1082
- **~.DriveData.nominalTorque** = P2003
- **~.DriveData.MaxTorque** = P1520

For SINAMICS and linear motors, and for the setting **SetpointDriverInfo.LinearMotorDriveData.SpeedReference** = NOMINAL_VALUE and **SetpointDriverInfo.LinearMotorDriveData.ForceReference** = NOMINAL_VALUE, the following applies:

- **~.LinearMotorDriveData.nominalSpeed** = P2000
- **~.LinearMotorDriveData.MaxSpeed** = P1082
- **~.LinearMotorDriveData.nominalForce** = P2003
- **~.LinearMotorDriveData.MaxForce** = P1520.

By default, these settings are transferred by the drive during the assignment of the axis to a SINAMICS drive by the system as of V4.2.

By default, the settings for torque granularity (see below as well) are transferred by the system from the drive:

~.driveData.torqueReductionGranularity = P1544

~.LinearMotorDriveData.ForceReductionGranularity = P1544

Enabling/disabling

The limiting function is disabled by default and must be specifically activated.

The torque limiting is activated using the **_enableTorqueLimiting()** command and has the following effect:

- The reduced maximum torque limit is in effect immediately
- The following error and positioning monitoring are disabled

With the function parameter **torqueLimitType**=USER_DEFAULT (system default), the default torque limit value set in the **userDefaultTorqueLimiting.torqueLimit** system variable is used. This default can be modified by entering a value in the **torqueLimit** parameter. The value of the parameter is interpreted as a percentage.

With the setting **torqueLimitType**=DIRECT, the value is specified as torque or force based on the **torqueLimitUnit** parameter. Depending on the **torqueLimitUnit** parameter, this value refers to the load or the motor side. For more information, see Conversion of torque/force (Page 176).

The torque reduction is not disabled with **_disableAxis()**; it must be explicitly disabled via a command.

It is disabled/canceled

- With **_disableTorqueLimiting()**
- In the case of **_resetAxis()**, **_restartAxis()**, etc.
- In the case of a transition to STOP

Suppression of the "Speed alignment" fault message of the drive

If torque reduction is enabled and the value is not equal to 0, the fault message of the drive (discrepancy between actual speed and the speed setpoint) is suppressed.

If the fault message is also to be suppressed when the torque reduction is disabled or when the torque reduction is enabled and the value is equal to 0, bit 8 in STW2 can be set for message suppression using the **_setAxisStw()** function.

The setting via the **_setAxisStw()** function has a storing effect, i.e. it is kept during and after the period that a torque reduction is activated.

Special features

- The commands for torque limiting (**_enableTorqueLimiting()**) and travel to fixed endstop (**_enableMovingToEndStop()**) cannot be active simultaneously.
  The transition from _enableTorqueLimiting() to **_enableMovingToEndStop()** is permitted (and has an override effect).
  The transition from **_enableMovingToEndStop()** to **_enableTorqueLimiting()** is not permissible since the setpoint must be clamped when retaining the torque in the fixed endstop.
- The **_stopEmergency()** command:
  If a motion is ended with position control (movingMode = POSITION_CONTROLLED) with the **_stopEmergency()** command, the stop ramp is generated from the current set position in stop modes STOP_IN_DEFINED_TIME, STOP_WITH_DYNAMIC_PARAMETER, and STOP_WITH_MAXIMAL_DECELERATION.
  If a following error has built up at this time due to active torque limiting, it is removed, and as a result, the axis does not come to an immediate standstill.
  For this reason, if you are using torque limiting, you should stop the motion with speed control (movingMode = SPEED_CONTROLLED). Alternatively, you can select the WHEN_COMMAND_VALUE_ZERO stop mode for stopping the motion with position control.
  An active torque reduction (including when moving to a fixed end stop) is retained.
  Exception:
  The **_stopEmergency()** command with stopDriveMode = STOP_WITH_COMMAND_VALUE_ZERO disables the torque reduction, and the travel to fixed endstop command is aborted.
- When torque limiting is active, the following error monitoring is disabled.
  As a result of torque limiting, for example, a greater difference between actual and desired values can build up on the position-controlled axis, which can cause the axis to accelerate (to reduce the difference) even if the velocity calculated by the interpolator is now lower.
  If, for example, torque limiting is not required during the acceleration phase, the function must not be activated until after the acceleration phase; otherwise, the acceleration must be reduced.
- The torque limits B+ / B- and the torque reduction must not be active simultaneously.

Status displays

Table 3-32 Status display variables

Variable	Meaning
TorqueLimitingCommand.State	
	Indicates the torque limiting state. ACTIVE: Torque limiting activated INACTIVE: Torque limiting deactivated
TorqueLimitingCommand.torqueLimitingState	
	Here, it is indicated if the drive is running at the torque limit. **Note:** This state is derived from the PROFIdrive profile status word 'MeldW' (PZD 5), bit 1 (M < Mx) of the drive.

Assignment of the resolution in the drive

Table 3-33 System data

SIMOTION system data	Meaning
userDefaultTorqueLimiting.torqueLimit	
	This data element specifies the user default setting of the torque limit value for the torquelimit function parameter in the _enableTorqueLimiting() command. This value is accessed when the command contains the USERDEFAULT setting. System default: 10.0

Setting the resolution in SIMOTION

The resolution of the torque reduction to the drive can be set, as of V4.0,
in **driveData.torqueReductionGranularity** or
in **linearMotorDriveData.forceReductionGranularity**.
In terms of value transfer, also consult the above section *Value specifications and transferring from the drive*.

Possible settings:

- BASIC (default, 1% resolution)
  64H in PZD (PROFIdrive telegram) is a torque reduction of 100%.
  - SINAMICS
    Setting required: P1544 = 16384.0
  - SIMODRIVE 611U
    Standard setting: P0881 = 16384.0
- STANDARD (~0.006% resolution)
  4000h in PZD (PROFIdrive telegram) is a torque reduction of 100%.
  - SINAMICS
    Standard setting: P1544 = 100.0
  - SIMODRIVE 611U
    Setting required: P0881 = 100.0

Note

Be sure to set the values in the drive consistently.

See also

Enabling and disabling torque limiting (Page 333)

3.15.2 Conversion of torque/force

Speed-controlled axes and rotary axes

When the **_enableTorqueLimiting()** function is programmed, there is always a torque specified in Nm, kNm or MNm.

If the TORQUE setting is specified when calling the function with the **torqueLimitUnit** function parameter, the specified torque refers to the motor. The gear ratio is not taken into account.

If the DEFAULT_UNIT setting is selected in the function parameter **torqueLimitUnit**, the torque refers to the load side and the gear ratio is taken into account. The following conversion formula applies:

$M_{Load} = M_{Motor}$ x (motor revolution/load revolution)

Linear axes with standard motor

In the **_enableTorqueLimiting()** function, the following can be optionally specified: a torque in Nm, kNm or MNm in relation to the motor, or a force in N, kN or MN in relation to the load side.

If the TORQUE setting is specified when calling the function with the **torqueLimitUnit** function parameter, the programmed value is interpreted as torque in relation to the motor. The gear ratio, leadscrew pitch, and efficiency of the leadscrew are not taken into account.

If the DEFAULT_UNIT setting is selected in the function parameter **torqueLimitUnit**, the programmed value is interpreted as force in relation to the load side. The gear ratio, leadscrew pitch, and efficiency of the leadscrew are taken into account in this setting. The following conversion formula applies, whereby **S** is the leadscrew pitch (adjustable in **leadscrew.pitchval**) and η is the efficiency of the leadscrew (adjustable in **leadscrew.efficiency**):

$F = M_{Motor}$ x 2 x π x ($\eta_{Leadscrew}$ / S) x (motor revolution/load revolution)

Linear axes with linear motor

There is always a force in N, kN or MN specified when programming.

3.16 Travel to fixed endstop

The **_enableMovingToEndStop( )** function activates the monitoring of **travel to fixed endstop** in parallel with an axis motion activated by a motion command and the retention of a clamping torque once the fixed endstop has been reached. The operation is also referred to as **clamping**.

This function requires that torque reduction be supported by the digital coupled drive.

Positioning with an immediate step enabling condition must take place first (position-controlled traversing).

During configuration, a setting can be made in **clampingMonitoring.recognitionMode** to define which of the following methods is to be used for detecting that the fixed stop has been reached:

- The following error limit is exceeded
- The torque limit is exceeded (evaluation of the **| M | < M**x message bit in the PROFIdrive telegram / status word (message word bit1 for SIEMENS telegram 101, 102, 103, 104, 105, 106, 125 and 126))
  As of V4.2, 10x telegrams are used with automatic telegram definition.

If the criterion is met, the fixed endstop status is regarded as having been reached, the **movingToEndStop.clampingState**= ACTIVE system variable is enabled, and the clamping tolerance is activated. Following error monitoring is switched off as soon as the command becomes effective. The setpoint at the input of the position controller is maintained. Provided that the position of the fixed endstop does not change, the difference between the setpoint and actual value remains the same.

Note

The position-related monitoring can remain active with TypeOfAxis.ServoMonitoring.motionMonitoringWhenTorqueLimiting even with active torque limiting.

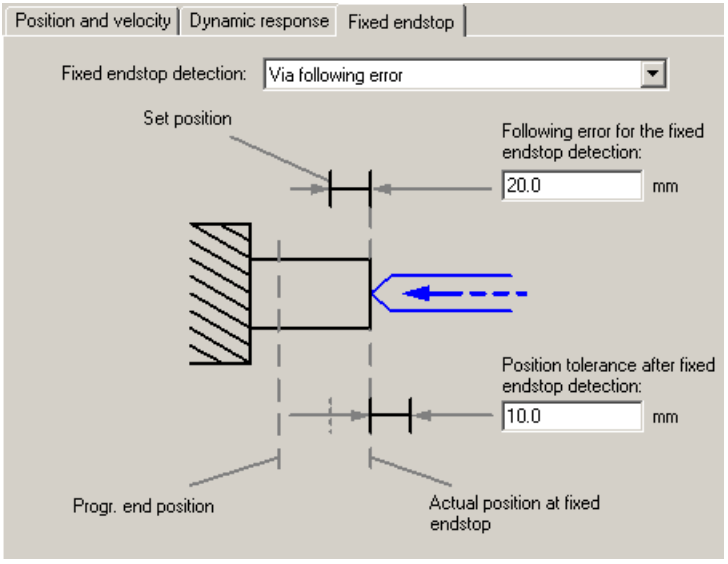


Figure 3-78 Function and parameters for travel to endstop with detection of fixed endstop by means of following error

The motion command is aborted with an error message; closed-loop control remains active. The setpoint at the input of the position controller is clamped. For new motion commands, the clamping set position is used as the start position. Motion commands in the direction of clamping are aborted. Motion commands in the opposite direction to the clamping direction are executed, thereby reducing the torque.

Travel to fixed stop is canceled when the clamping tolerance window is exited.

Issuing the **_enableMovingToEndStop()** command again can cause the torque to switch over, even if clamping is active.

The following commands cancel the function:

- **_resetAxis()**
- **_disableAxis()**
- **_disableMovingToEndStop()**
  - Up to and including V4.1 SP3: Cancel if outside the clamping window
  - As of V4.1 SP3 HF4: Cancel always
- **_cancelAxisCommand()** as of V4.1 SP3 HF4

Non-stepped torque transitions and torque retention over a defined time period can be implemented in the user program, as can specification of torque profiles.

Note

If the clamping torque is increased during clamping (by a new **_enableMovingToEndStop()** command), then this torque might not be achieved. Position control is active, but no more setpoints are generated for a motion in the direction of clamping.

The **_disableMovingToEndStop()** command deactivates the **Travel to fixed endstop** function. The **_disableMovingToEndStop()** command has no effect while clamping is active. The **_disableMovingToEndStop** command can be issued in parallel to position-controlled motion.

If a breakaway of the fixed endstop occurs, the position controller abruptly reduces this difference. This response is used for monitoring purposes. Clamping monitoring saves the current actual value that applies at the time of clamping. The clamping tolerance window then wraps around this value (see the figure above). This is why the clamping tolerance value should be lower than the setpoint/actual value difference at the point when clamping is detected. An error is only detected if the axis leaves the clamping tolerance window during abrupt reduction of the setpoint/actual value difference. In this case, a 50108 alarm is triggered.

If the clamping tolerance window is exited, **alarm 50108: Clamping monitoring error** is triggered and the **travel to fixed stop** is aborted.

In the command, the limiting force for the clamping value is set in [N] for linear axes, while for rotary axes, the limiting torque is set in [Nm].

If the **_enableMovingToEndStop()** command is active and the fixed stop has not been detected, the system behaves the same as with active torque limiting.

The **_enableMovingToEndStop()** and **_enableTorqueLimiting()** commands cannot be active simultaneously.

The transition from "torque limiting" (**_enableTorqueLimiting()**) to "travel to fixed stop" (**_enableMovingToEndStop()**) is permitted (and has an override effect), but the reverse is not permitted.

System variables

Table 3-34 System variables for travel to fixed stop

Variable	State	Meaning
MovingToEndStopCommand.state	ACTIVE / INACTIVE	Command status
MovingToEndStopCommand.clampingState	ACTIVE / INACTIVE	Clamping torque reached status

Note

In the case of travel to fixed stop with fixed stop detection by means of the following error, the entry for the position tolerance applicable to the fixed stop detection should be significantly less than the entry for the following error for the fixed stop detection.

Note

If the "travel to fixed stop" function is to be used in combination with the RecognitionMode=CLAMP_WHEN_TORQUE_LIMIT_REACHED setting, the following setting must also be made on the 611U and in SINAMICS:

- 611U
  Set parameter P1426 ("tolerance band for nset = nact message") to the maximum speed of the drive (P880).
- SINAMICS
  Set parameter P2163 ("Speed 4 threshold value") to the maximum possible value.
- Axis with digital drive link and no setting of a 1xx telegram (i.e. telegram without torque reduction and without actual torque identification)
  The "travel to fixed stop" functionality is only possible by means of the following error criterion.

You will also find further information about travel with torque limiting and travel to fixed stop in the corresponding drive documentation.

See also

Overview of torque limiting via torque reduction (Page 171)

3.17 Technology data

Technology data can be specified cyclically from the controller to the drive or read cyclically by activating the technology data block.

The technology values are needed to implement winder functionality with SIMOTION, for example.

The setting of the motor type determines whether the physical quantity is force or torque (round-frame motor: torque, linear motor: force).

For more information, refer to *'LREAL' interconnection interface type* in the **Motion Control Basic Functions** manual.

The **additive data block 1** contains:

- Controller → drive:
  - Additive torque setpoint: 16 bits
    (display of the value in the **additiveTorque.value** system variable)
  - Positive torque limit (B+): 16 bits
    (display the value in the **torqueLimitPositive.value** system variable)
  - Negative torque limit (B-): 16 bits
    (display of the value in the **torqueLimitNegative.value** system variable)
- Drive → controller:
  - Actual torque value: 16 bits
    (display of the value in the **actualTorque.value** system variable)

As of SIMOTION V4.2, the technology data block is set up and connected automatically.

The technology data is appended as INT to the In/Out drive protocol. This is activated in the **technologicalData.enable** configuration data element.

The logical addresses for the technology data are displayed in the **technologicalDataOutInfo** and **technologicalDataInInfo** configuration data elements.

- They are set when the telegram is specified in the axis wizard. (When you set the axis telegram in HW Config, the relevant data must be created via the PZD area.)

The values B+, B- and additive torque setpoint

- Are indicated and entered according to their associated units.
- Are normalized to the specified maximum values (maxTorque, maxForce) before they are transferred to the drive and are sent to the drive as % values.
- Can be sent to the drive as values from 0 to +/- 200%.

The positive and negative torque limits (torqueLimitPositive (B+), torquelimitNegative (B-)) are sent as a weighting, rather than as a reduction.

The values are sent with reference to the 100% value:

- 4000H corresponds to 100%
- 7FFFH corresponds to 200%

Negative values are possible.

Note

The **SetpointDriverInfo.DriveData.maxTorque** or **SetpointDriverInfo.LinearMotorDriveData.MaxForce** configuration data and the P2003 parameter on the drive serve as normalization basis on the axis. The values must be equal.

See also Chapter Overview of torque limiting via torque reduction (Page 171)
Section *Specification of the values and transfer from the drive*.

Creating technology data block 1 (e.g. with SINAMICS drive):

The drive has already been configured.

Telegram extension (as of V4.2)

As of V4.2, with SINAMICS drives and the **Use symbolic assignment** SCOUT setting (default setting) activated, telegram extension is set automatically by the system.

Telegram extension in HW Config (up to V4.1 SP1)

- Set standard telegram for the drive, then add to the PZD via Details:
  - Input: +1 (PZD)
  - Output: +3 (PZD)
- Save HW Config

Telegram extension in SCOUT (as of V4.1 SP1)

- In the project navigator, select the drive device (e.g. "SINAMICS_Integrated") and open the configuration:
  Project - CPU (e.g. D435) - drive device (e.g. SINAMICS_Integrated) - configuration
- Mark the line of the drive, e.g. "Drive_1"
- Click the "Insert line" button and select "Telegram extension"
- Set one word for the input data and three words for the output data
- Click the "Transfer to HW Config" button and confirm the comparison with HW Config
- Click the "Configure telegram" button
- For "PROFIBUS receive direction" and "PROFIBUS send direction", enter the telegram extension according to the table below at:
  Project - CPU (e.g. D435) - drive device (e.g. SINAMICS_Integrated) - Drives - Drive_x - Communication - PROFIBUS

Table 3-35 BICo interconnections

PROFIdrive telegram - receive direction		
	1st additional PZD (M_Add)	P1511
	2nd additional PZD (B+)	P1522
	3rd additional PZD (B-)	P1523
PROFIdrive telegram - send direction		
	1st additional PZD (M_act)	E.g. R0080 (actual torque value)

The values are normalized in the drive via P2003.

NOTICE
Scaling of additional torque 1 Parameter P1512 "Additional torque 1 scaling" must be interconnected so that additional torque (M_Add) is transferred correctly. Values from 0 to 100% can be set. A scaling factor of 100% corresponds to the transfer of the value configured in M_Add.

Activation in SIMOTION

The technology data block is created when the axis is configured.

The technology data block can only be assigned correctly by the system for active symbolic assignment, manual telegram setting and deactivated automatic telegram extension when the correct BICO interconnection for the technology data block is available on the drive side.

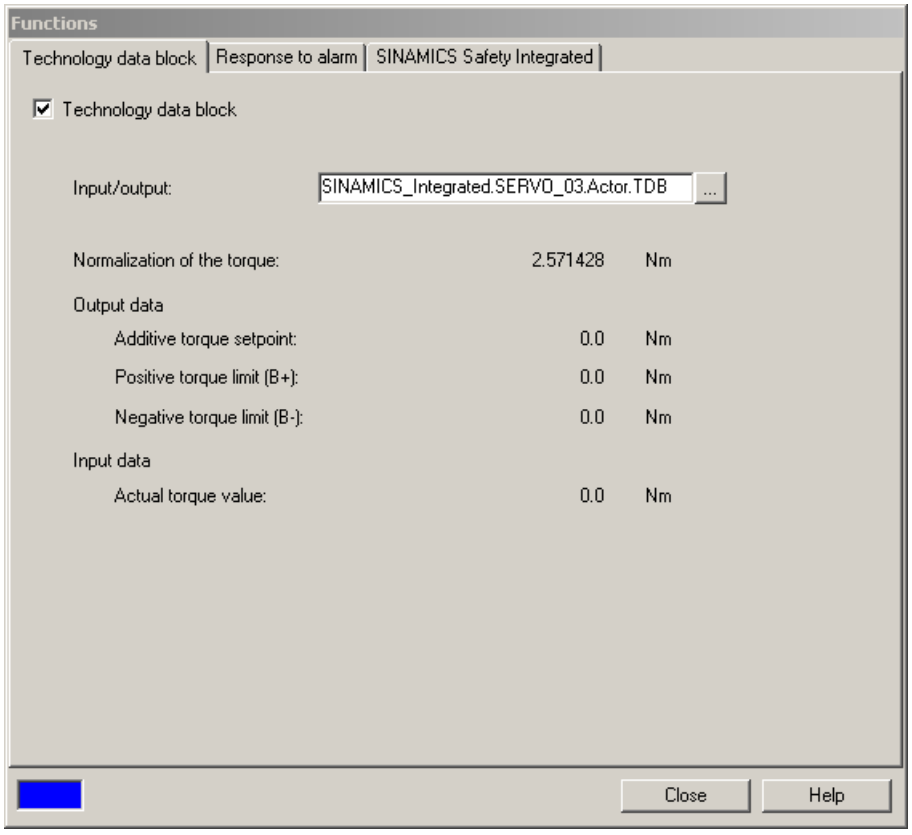


Figure 3-79 Activating a technology data block

Alarms

20005 "Logical address of technology data block driver not found"

The limiting values and additive values are initialized/transferred regardless of the activation and error status of the axis.

3.18 Torque limiting B+/B- (V3.2 and higher)

The torque limits B+/B- can be cyclically specified on the electric axis downstream of the speed controller using the expanded drive message frame to the drive.

As a prerequisite, the technology data block (Page 179) must have been activated on the electric axis. The torque limits B+ and B- are set downstream of the speed controller.

By specifying the upper limit B+ and the lower limit B-, it is possible to enter a specific range that does not have to be symmetrical with respect to the zero position.

The limit values can be predefined on the axis directly by means of cyclically effective system variables. These values can be preset.

The limit values can also be defined on the axis by means of interconnection with other technology objects.

The B+ value and the B- value can be interconnected by means of an LREAL interface or an LREAL connector (using a formula object or a controller object, for example).

See also:

- **SIMOTION Supplementary Technology Objects** function description
- **Motion Control Basic Functions** Manual, Sections *Interconnection using general interconnection dialog box* and *'LREAL' interface type*

Note

If the manipulated variable is inverted on the Axis technology object, then B+/B- are also reversed on the Axis technology object. You must take this into account when specifying the torque limits B+/B-.

Units

The unit settings are used.

The physical quantity depends on the type of motor:

- For rotary motors: Torque [Nm]
- For linear motors: Force [N]

Commands and effectiveness

The limiting is deactivated by default and has to be explicitly activated.

The torque limits are activated using the **_enableAxisTorqueLimitPositive()** / **_enableAxisTorqueLimitNegative()** commands.

The command specifies whether the limits relate to the interconnected value, the system variable or a value defined on the command itself. If they relate to the system variable, then this is applied cyclically; in other words, a change in the system variables takes effect immediately.

If torque limitation B+/B- is activated using the above commands, this will deactivate the following monitoring and limiting functions:

- Following error monitoring
- Velocity error monitoring via reference model
- Time-outs for positioning monitoring and standstill monitoring

The torque limits are not disabled with **_disableAxis()**; they must be disabled explicitly using a command.

The torque limits are deactivated or aborted using the following commands:

- **_disableAxisTorqueLimitPositive()** / **_disableAxisTorqueLimitNegative()**
- **_resetAxis()**, **_restartAxis()**, etc.
- In the case of a transition to STOP

If the function is not activated, the maximum torque and maximum force are output via the interface in accordance with **maxTorque** and **maxForce**.

With activated torque limiting B+ / B- (**_enableAxisTorqueLimitPositive()** / **_enableAxisTorqueLimitNegative()**), as of V4.2, with setting **TypeOfAxis.SetPointDriverInfo.DriveData.torqueReference**, the = MAX_VALUE maximum of the set value in **~.DriveData.maxTorque** or **maxForce** is transferred to the drive.

In the versions <V4.2 up to 200% of this value are possible with activated torque limitation.

If the torque limiting function B+ / B- is not activated, the maximum value set in **~.maxTorque** is transferred in V4.2 as well as in versions <V4.2.

When using the technology data block and with the setting **~.torqueReference** = MAX_VALUE in **~.maxTorque** , the 100% reference value for torque transmission to the drive must be set, i.e., for the value corresponding to p2003 (reference torque), a greater maximum torque is possible in the drive, in accordance with the value in p1520 (maximum torque).

The separation of reference and maximum torque has been introduced in the reference variables with V4.2.

In projects with V4.2 or higher, the new configuration data **~.torqueReference** is set to MAX_VALUE. As such, only the reference torque value can be set as maximum torque limitation without changing projects.

Remedy:

If, in the case of activated torque limiting, a value greater than the reference value in p2003 is transferred to the drive**, ~.torqueReference** = NOMINAL_VALUE needs to be set, in **~.nominalTorque** the value corresponding to p2003 and in **~.maxTorque** the value corresponding to p1520 must be entered.

With the deactivated torque limiting function B+ / B, the set value in **maxTorque** is transferred to the drive.

For new TOs created in V4.2, **~.torqueReference** = NOMINAL_VALUE is set, as well as for the default set adaption the reference values for **~.nominalTorque** and **~.maxTorque** for runtime are transferred from the drive.

The position-related monitoring can remain active even with active torque limiting. For this purpose, there is the **TypeOfAxis.ServoMonitoring.motionMonitoringWhenTorqueLimiting** configuration data as of V5.1. When activated, this ensures that the position monitoring, positioning monitoring and/or following error monitoring that was active before the torque limiting remains active during the limiting.

System variables

Table 3-36 System variables for torque limitation

Variable	Meaning
torqueLimitPositiveIn	Value, state and validity of the interconnected positive limitation
torqueLimitPositiveInDefault	Default
torqueLimitNegativeIn	Value, state and validity of the interconnected negative limitation
torqueLimitNegativeInDefault	Default
torqueLimitPositive	Shows the programmed limit value
torqueLimitNegative	
torqueLimitPositiveCommand	Command status / function status
torqueLimitNegativeCommand	
motionMonitoringWhenTorqueLimiting	The position-related monitoring is active during active torque limiting.

Suppression of the "Motor blocked" fault message of the drive

Message: **F07900 (N, A) drive: Motor blocked / speed controller at its endstop**

Cause: The motor has been operating at the torque limit for longer than the time specified in p2177 and below the speed threshold set in p2175.

If the fault message is to be suppressed, the **_setAxisStw()** function can be used once to set bit 8 in ctrl word2 for message suppression.

The setting via the **_setAxisStw()** function has a storing effect, i.e., it is kept during and after the period that a torque reduction is activated.

The p2175 (0.0) parameter to suppress the fault message on the drive.

See also

Coupling of digital drives (Page 56)

Technology data (Page 179)

3.19 Additive set torque (V3.2 and higher)

An additive set torque can be transmitted to the drive on the axis.

This requires that an axis with digital drive link is used and that the technology data block (Page 179) is activated.

The additive setpoint torque can be set on the axis directly using a cyclically effective system variable (**defaultAdditiveTorque**). This value can be preset.

The additive setpoint torque can also be defined on the axis by means of interconnection with other technology objects.
The additive torque can be interconnected by means of an LREAL interface or an LREAL connector (using a formula object or a controller object, for example).

See also:

- **SIMOTION Supplementary Technology Objects** function description
- **Motion Control Basic Functions** manual, *Interconnection using general interconnection dialog box* and *'LREAL' interface type* chapters

The function is activated via the **_enableAxisAdditiveTorque()** command. Without an activation, the transmitted additive torque is = 0.

The command specifies whether the limits relate to the interconnected value, the system variable or a value defined on the command itself. If they relate to the system variable, then this is applied cyclically; in other words, a change in the system variables takes effect immediately.

The **additiveTorque** configuration data element can be used to determine whether the last valid value or the replacement value is used if an interconnection value is activated but is invalid. After system start-up, the last valid value is 0.

The additive set torque can also be specified directly via the replacement value.

The replacement value can be preset.

This value takes effect immediately after a change if the function is active.

For more information, refer to *'LREAL' interconnection interface type* in the **Motion Control Basic Functions** manual.

Unit

The unit setting is used.

The physical quantity depends on the type of motor:

- For rotary motors: Torque [Nm]
- For linear motors: Force [N]

Commands

_enableAxisAdditiveTorque() enables the additive torque.

The additive set torque is not disabled with **_disableAxis()**; instead it must be explicitly deactivated via a command.

It is disabled/canceled

- with **_disableAxisAdditiveTorque()**
- In the case of **_resetAxis()**, **_restartAxis()**, etc.
- In the case of a transition to STOP

System variables

Table 3-37 System variables for the additive set torque

Variable	Meaning
additiveTorqueIn	Value, state and validity of the additive set torque
defaultAdditiveTorque	Default
additiveTorque	Value that is sent to the drive (before normalization)

Suppression of the "Motor blocked" fault message of the drive

See Torque limiting B+/B- (V3.2 and higher) (Page 182).

See also

Coupling of digital drives (Page 56)

3.20 Force/pressure control

3.20.1 Overview of force/pressure control

Force/pressure control requires at least the **position axis technology**. If a hydraulic axis is used as a hydraulic drive axis, force/pressure control is also possible here.

Pressure control and force control are available on the position axis. The actual pressure or force value is measured using an external sensor. Several force/pressure sensors can be connected to the axis. The actual force/pressure values can be smoothed with a **PT1 filter**. A **PT2 filter** is available at the output of the pressure control.

The manipulated variable on the drive is still the speed or velocity setpoint.

Monitoring for maximum force, control deviation, force/pressure entry (see positioning monitoring) and force/pressure end value is performed on the actual value side. Limiting functions on the setpoint side include speed setpoint limiting (manipulated variable limitation) and anti-windup (I-component) with manipulated variable limitation.

Conditions can be specified in a specific activation command **_enableForceControlByCondition()** for switchover from position-controlled motion to force/pressure controlled status. The force/pressure setpoint or profile is specified using commands.

The commands for force/pressure control can also be programmed on the virtual axis. Commands that cannot be executed, such as activate force/pressure controller, position-related profile, or data set commands generate a technological alarm, **Technological alarm: 030015: Technology not configured**.

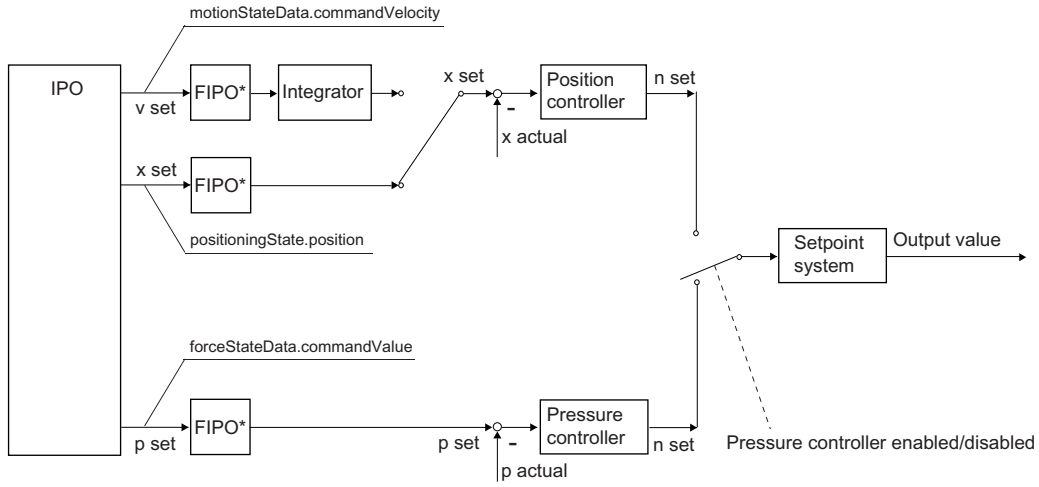


Figure 3-80 Overview of control structure for force/pressure control

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

You can call signal flow diagrams under the Axis object in the project navigator. The signal flow diagram illustrates the signal path and enables major items of configuration data and important system variables to be parameterized using the signal flow.

Possible operating modes with force/pressure control

- **Force/pressure control active, setpoint mode**
  Actual force/pressure value is controlled to force/pressure setpoint.
  Monitoring and limiting functions are active.
- **Force/pressure control active, follow-up mode**
  Force/pressure setpoint is controlled to actual force/pressure value.
  Force/pressure monitoring and limiting functions are not active in follow-up mode.
- **Program simulation**
  (analogous to positioning mode)
  Force/pressure setpoints are calculated but not output.
  Actual force/pressure value is set to force/pressure setpoint.
- **Velocity-limited mode**
  The closed-loop force/pressure control or open-loop force/pressure control is the master. The velocity is limited in parallel.

Remark

While the force/pressure controller is active, the position values of the IPO are corrected.

Switchover between position-controlled motion and force/pressure-controlled motion is performed by the application.

Fine interpolation

The force/pressure values undergo linear fine interpolation (V4.1 SP1 and later).

See also

Force/pressure control with hydraulic speed-controlled axes with Q valve only (Page 305)

Technology data (Page 179)

3.20.2 Configuration of the actual force/pressure value sensors

More than one pressure or force sensor can be configured.

The various sensors can be of different types and have different interfaces. The sensor for the transition criterion can be specified in the switchover command.

There can be up to eight (8) sensors for the force/pressure value on the axis and up to eight (8) binary inputs on the axis, for example, for switchover criteria.

Because the sensor can be accessed via the logical address, the same sensor can therefore be used on several axes. The sensor data must be set separately on each axis.

In V4.0 or higher, a calculated value from the user program can also be specified as the actual sensor value via the cyclically effective **forceActualValueSetting.value** system variable. This requires the setting **additionalSensorType**= SET_ACTUAL_VALUE.

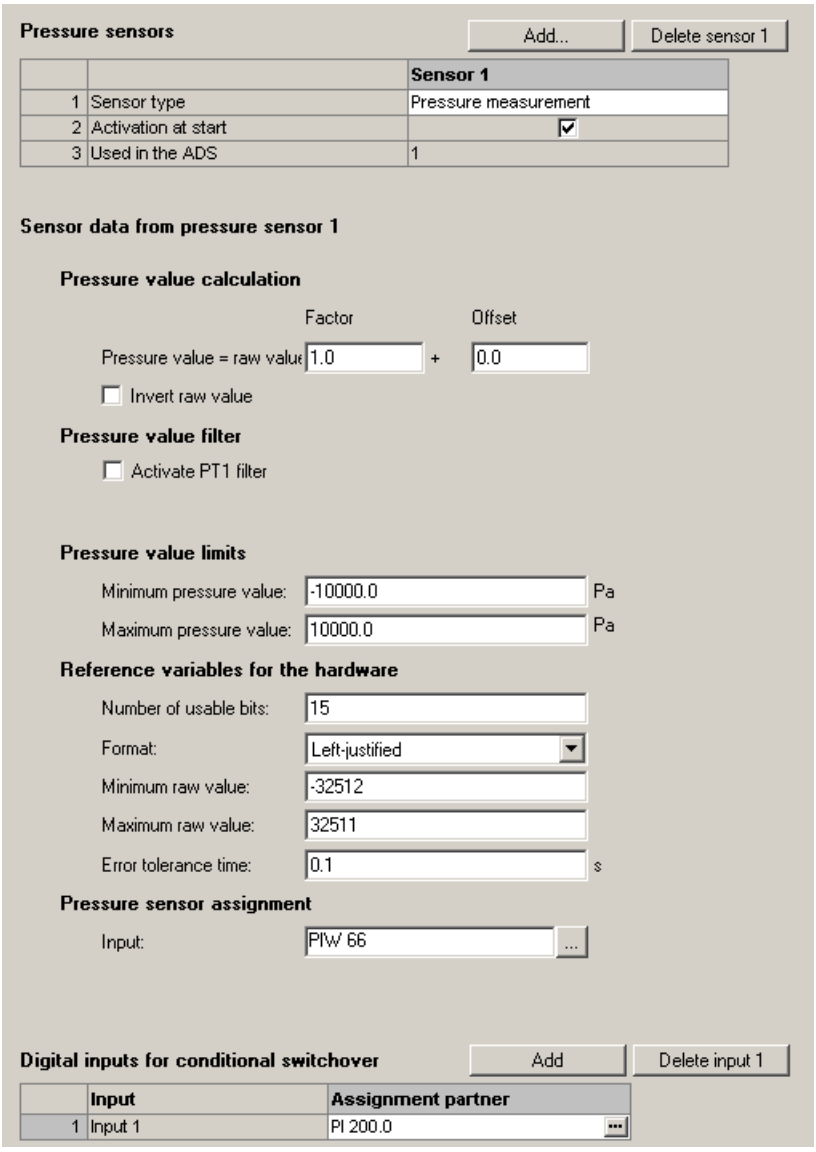


Figure 3-81 Setting the force/pressure sensor

The force/pressure sensor interface can be activated and deactivated on the axis using the **_enableAxisInterface()** and **_disableAxisInterface()** commands.

The Activated/Deactivated force/pressure sensor interface status is displayed in the **additionalSensorData.additionalSensorData[i].input** system variable.

3.20.3 Controller for force/pressure control

The force/pressure control is carried out via an adjustable PID controller.

The controller has the same structure for force/pressure control and for force/pressure limiting. One controller for both tasks is available; the controller parameters for each task can be accessed by switching the data set.

The I component of the controller is implicitly set during enabling. When the data set is changed over, the system sets the I component so that a continuous setpoint curve is obtained.

When pressure limiting is enabled, the most recent position setpoint is maintained during the switchover to pressure control; otherwise, the actual value is applied. The force/pressure setpoint is set the same as the actual force/pressure value at the switchover time.

The initial value of the force/pressure controller can be limited by an upper value (**forceControllerData.outputLimits._max**) and a lower value (**forceControllerData.outputLimits._min**). It is thus possible to react in one direction only, e.g. to maintain pressure, to move in one direction only in the event of overpressure, to not actively build up pressure.

If the interpolator cycle clock is not the same as the position control cycle clock, a linear fine interpolation is carried out for the force/pressure setpoint. The force/pressure setpoints are limited to a maximum value.

If the hydraulic functionality of the axis is used, specific factors can be predefined for the sliding friction and the additive offset for the force/pressure control.

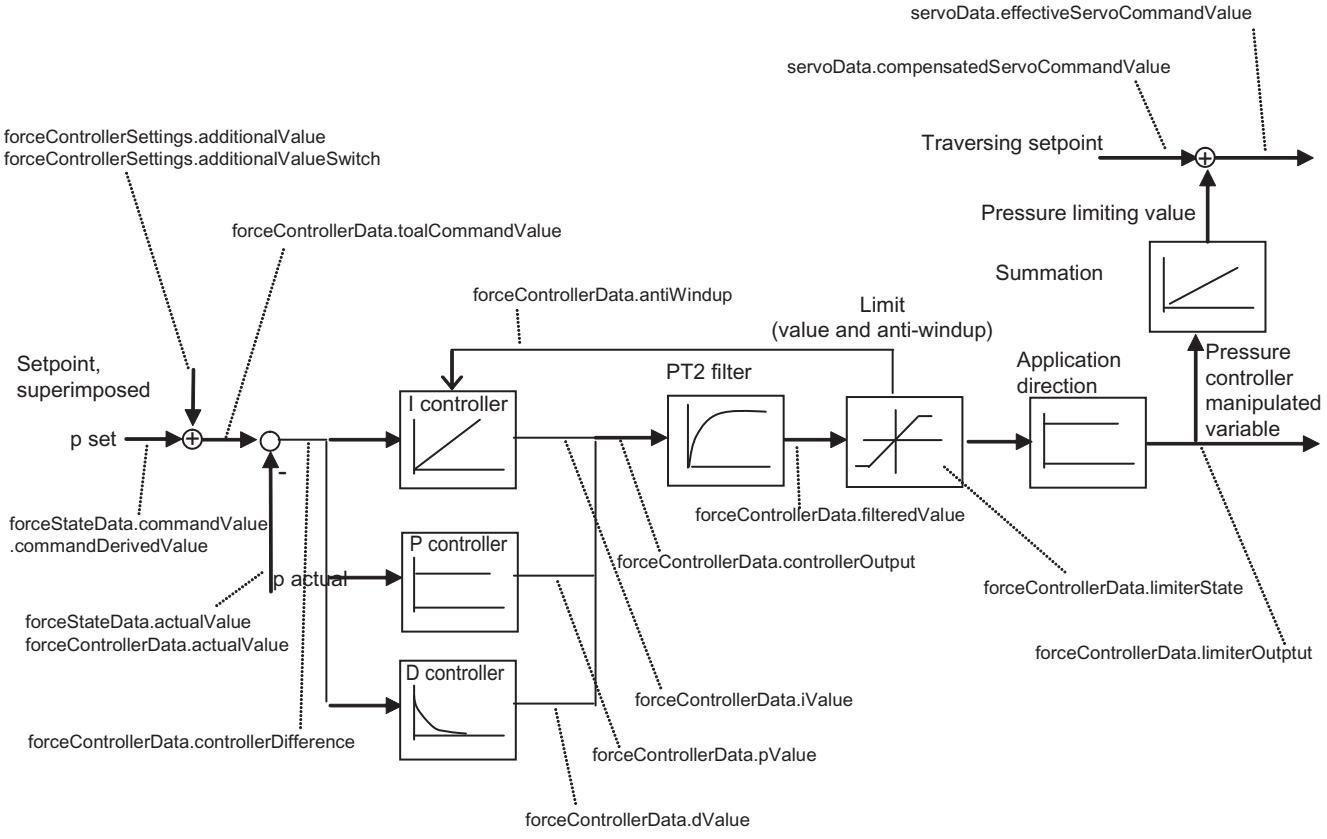


Figure 3-82 Overview of pressure controller/pressure limiting controller

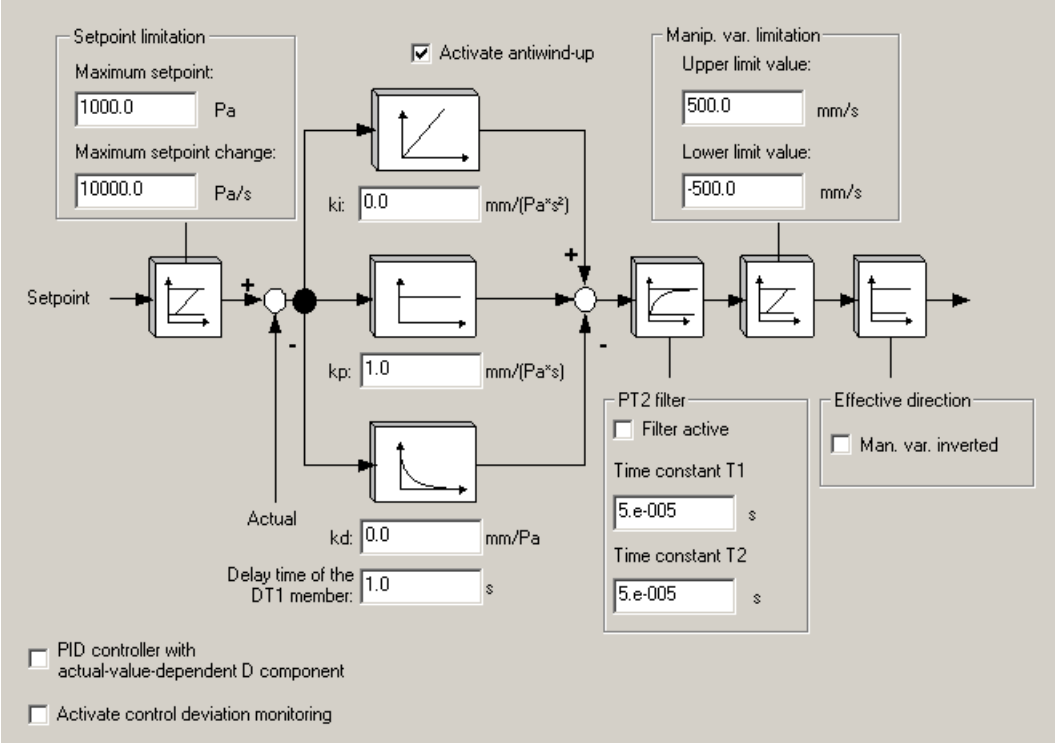


Figure 3-83 Controller setting

Reducing the I component via the user program (as of V4.1 SP1):

It is now possible to reduce the I component of the pressure controller with a PT1 filter (setting in **PID_Controller.iValueFeedbackTimeConstant** configuration data element) using the write-accessible **forceControllerSettings.integratorMode** system variable from the user program.

This enables faster reduction of the I component, for example, following the switchover from pressure limiting to pressure control.

See also

Compensations that are active only on the axis with hydraulic functionality (Page 298)

3.20.4 Monitoring functions / limiting functions / emergency strategies with active force/pressure control

The actual force/pressure value is monitored for a maximum value or limit value. The control deviation is monitored for a maximum value.

Based on a force/pressure entry window, you can monitor correct termination of a force/pressure profile (force/pressure entry monitoring) – refer to positioning monitoring during positioning.

You can also monitor observance of a constant force/pressure setpoint within a tolerance band (force/pressure end value monitoring) – see standstill monitoring on a position-controlled axis.

Travel to software limit switch:

When traveling to the software limit switch, pressure limitation is deactivated.

You must ensure that the maximum deceleration values are set correctly so that the setpoints of the software limit switches are not overtraveled when the system switches automatically from pressure-limited operation to position-controlled operation.

Limiting functions are available for:

- Force/pressure setpoint
- Force/pressure controlled variable
- Actual force/pressure value

Note

To allow an immediate stop with active force/pressure control, e.g. via the **_stopEmergency()** command (stopDriveMode=WITH_COMMAND_VALUE_ZERO), the control switches to pressure limitation and, with pressure limitation active for the force/pressure setpoint, moves to velocity 0 using the specified velocity profile for active position control.

If the STOP_EMERGENCY alarm response occurs for active pressure control, a switch is made to position control using pressure limitation. The pressure setpoint is active at the switching point. If in this case, the actual pressure value is higher than the pressure setpoint or limit value, the actual pressure value will be removed appropriately.

See also

Dynamic limiting functions (Page 160)

3.20.5 Activating the force/pressure control

Force/pressure control is activated by commands from the application.

SIMOTION supports different activation modes.

- **Direct activation**
  Direct activation is achieved using the axis-enable command (**_enableAxis()** or **_enableQFAxis()**) (**forceControlMode** parameter = ACTIVE). When enabled, the last actual force/pressure value is taken as the force/pressure setpoint. The position controller must also be enabled.
  The direct activation of the pressure control is possible in the stopped state (**motionStateData.stillStandVelocity** = ACTIVE).
- **Direct deactivation**
  Direct deactivation occurs via the command to reset the axis enable or the command for axis enable without the parameter for force/pressure control.
- **Automatic activation with condition**
  There is a separate command for automatic activation. The conditions are verified and the switchover takes place in the position control cycle clock. The axis is switched over to a pressure-time profile. The conditions (force, pressure, position, time, and input) are specified in the command and can be linked in the command in several stages. The conditions can also be changed before they occur by reissuing the command. The force, pressure, position, and time are stored at the time of switchover. The values are available through system variables.

Switching from position control to force/pressure control:

To obtain a continuous manipulated variable curve for switching from the position control to force/pressure control, the I component of the process controller will be initialized so that the new manipulated variable from the process controller corresponds to the estimated manipulated variable from the position controller.

The estimated manipulated variable is the last manipulated variable of the position controller plus the filtered actual acceleration multiplied with the clock time.

Switching from force/pressure control to position control:

To obtain a continuous manipulated variable curve for switching from force/pressure control to force/pressure control, the setpoint generation and the position controller will be restarted. The current actual value measured by the encoder modified with the estimated following error is used as starting value for the set position.

The estimated following error is calculated from the current manipulated variable multiplied by the replacement time constant of the position control loop PTC (**typeOfAxis.NumberOfDataSets.DataSet_1.DynamicData.positionTimeConstant** configuration data item).

The position controller then outputs in the first cycle clock after the switching the same manipulated variable as the force/pressure controller before the switching.

Note

In certain situations, the reference to the current manipulated variable can cause problems. This is the case, for example, when an offset compensation is active. The PTC for the switching time should then be set to zero. PTC can be switched with the data set.

3.20.6 Force/pressure setpoint specification

Force/pressure setpoints can be specified as follows:

- Directly as a value
- Via a time-related force/pressure profile
- Via a position-related force/pressure profile

The derivation value for the transition to the setpoint, or starting value can be specified in the command. The last force/pressure setpoint is maintained at the end of the profile.

Even with direct setpoint value specification, the command requires different runtimes for its execution on account of the possible transition to the pressure setpoint (pressure entry monitoring). Synchronous and asynchronous command execution can be set.

With direct force/pressure setpoint specification, the setpoint is always active immediately. **mergeMode** can be set in the force/pressure profile commands (sequential execution).

In position-related profiles the absolute position reference of the profile applies at the starting point. During ramp up to the profile and ramp down from the profile, the **additionalSensorData.derivedValue** refers to derivation of the setpoint over time.

An error message is output for starting at a position outside the profile.

3.20.7 Commissioning procedure for force/pressure control

Procedure for commissioning:

1. Commission the drive.
2. Commission the position controller for positioning mode.
3. Set the force/pressure controller. To do this, you can define setpoints for the force/pressure controller via the additive force/pressure setpoint using a function generator.

3.20.8 Force/pressure control with velocity limiting

On the position axis, velocity limiting (flow rate limiting for hydraulic functionality) can be activated in parallel with the force/pressure control. Velocity limitation is effective within pressure control.

It is activated in the same way, for example, as force limiting with limiting value setting or limiting profile enabling. Explicit deactivating is possible by means of a specific command.

If the velocity/flow rate limit is reached, the flow rate is limited, and the I component is stopped in the force/pressure controller.

Like torque limiting, velocity limiting can be activated in parallel with the motion commands.

Velocity limiting is enabled by means of the following commands:

- Activate position-related limiting profile
  _enablePositionLockedVelocityLimitingProfile()
- Activate time-related limiting profile
  _enableTimeLockedVelocityLimitingProfile( )
- Activate limiting value
  _enableVelocityLimitingValue()

Velocity limiting is disabled by means of the following command:

- **_disableVelocityLimitingValue()**

In the event of an error, the velocity limiting remains inactive, the force/pressure control is replaced by the force/pressure limiting and the velocity limiting is disabled.

The velocity limiting direction can be activated by means of the **velocityLimitingDirection** parameter (as of V4.1 SP1).

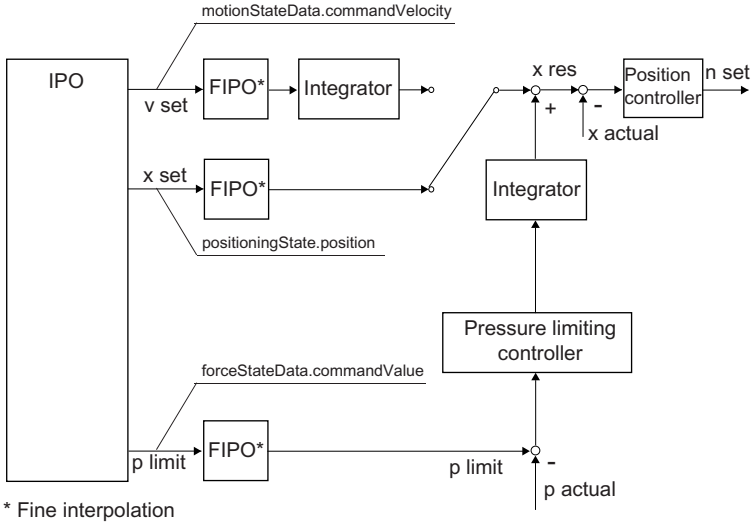
The set direction is indicated in **velocityLimitingCommand.velocityLimitingDirection**.

3.21 Force/pressure limiting

3.21.1 Overview of force/pressure limiting

The force/pressure limiting functionality requires **position axis technology**. If a hydraulic axis is used as a drive axis, force/pressure limiting is possible here, too.

The position controller is active as a lower-level controller if force/pressure limiting is active.



Overview of control structure for force/pressure limiting

With force/pressure limiting, the force/pressure controller is not activated until the force/pressure limit is exceeded ($\mathbf{p}_{act} > \mathbf{p}_{limit}$). It is limited to positive force/pressure values.

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

The force/pressure limiting is switched on using commands, e.g.:

- **_enableTimeLockedForceLimitingProfile()**
- **_enableMotionInPositionLockedForceLimitingProfile()**
- **_enablePositionLockedForceLimitingProfile()**
- **_enableForceLimitingValue()**
- **_enableForceLimitingByCondition()**

Time-related monitoring is deactivated.

The force/pressure limiting remains active in the event of an error, except for the RELEASE_DISABLE and OPEN_POSITION_CONTROL error reactions.

Position-related monitoring (e.g. following error monitoring or positioning monitoring) are deactivated when pressure control is activated or via the pressure limitation command.

You can configure the monitoring functions to remain active with external pressure/force limiting in the **servoMonitoring.motionMonitoringWhenExternalForceLimiting** configuration data element.

A PT2 filter on the force/pressure limiting controller output (as of V3.2) will prevent abrupt signal changes. The filter time constants can be modified online and take effect immediately.

The pressure limiting value is not subject to fine interpolation if **_enableForceLimitingValue()** or **_enable...LimitingProfile()** with derivativeLimitingMode=WITHOUT_LIMITING is set.

The status of the pressure limiting is displayed via the system variable **forceControllerData.limitingState**. The requirement for updating the status variable is that the pressure controller must be in operation (kp must not be equal to 0).

See also

Force/pressure limiting with hydraulic speed-controlled axes with Q valve only (V4.0 and higher) (Page 306)

3.21.2 Positioning with active force/pressure limiting (V3.2 and higher)

With position-related motions and active pressure limitation, cyclic continuation of the interpolator position and velocity can be set.

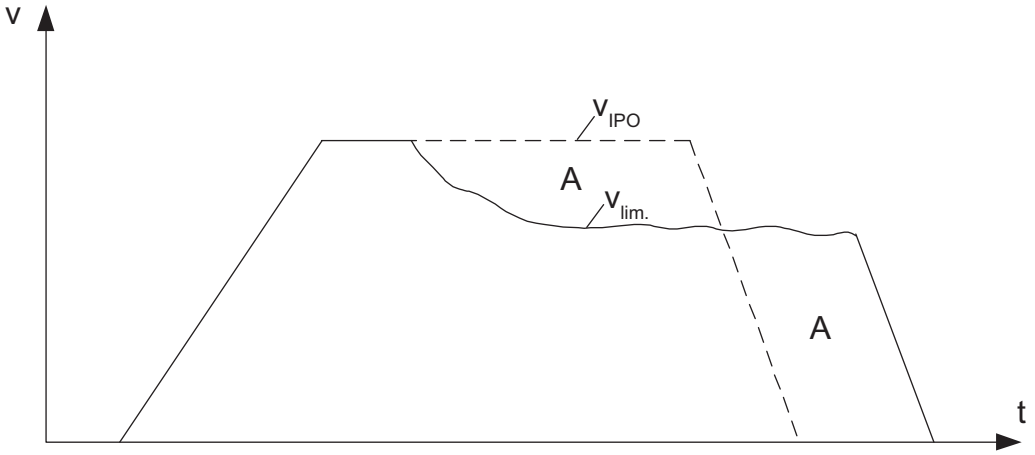


Figure 3-84 Position-related motion with pressure limitation and cyclic continuation

This example shows a v/t profile in which the velocity is reduced by the pressure limitation. Cyclic continuation allows continuous interpolation to the end point, starting from the current position and, optionally, the velocity.

The **decodingConfig.cyclicSetUpInForceLimiting** configuration data element defines how the continuation should take place.

With the setting **decodingConfig.cyclicSetUpInForceLimiting** = POSITION_BASED, cyclic continuation at the current set position and the non-reduced velocity of the interpolator takes place.

With the setting **decodingConfig.cyclicSetUpInForceLimiting** = POSITION_AND_DYNAMIC_BASED, cyclic continuation at the current position and velocity takes place.

Recommended setting: **decodingConfig.cyclicSetUpInForceLimiting** = POSITION_BASED

Braking phase initiated via software limit switch

Cyclic continuation stops from the time at which the braking phase is initiated because the software limit switch is reached. The drive shuts down at maximum deceleration. The pressure limitation remains active.

At standstill, the axis switches from position control to velocity control.

Behavior during positioning/moving to the target position

- Reaching the target position and actual position within the positioning window
  - The target point is reached with respect to the setpoint with cyclic continuation
  - The axis is positioned within the positioning window
  - MOTION_DONE is set and the command is ended as EXECUTED
- Reaching the target position within the positioning window
  - The target point is reached with respect to the setpoint with cyclic continuation
  - The axis is not positioned within the positioning window
  - MOTION_DONE is not set and the positioning monitoring alarm is suppressed due to active pressure limitation
  - Standstill monitoring is activated, and an alarm is triggered if the standstill window is overshot, which cancels the command if necessary
- As a result of pressure limitation, the target position is not reached even through cyclic continuation
  - The target point is not reached with respect to the setpoint even with cyclic continuation
  - No alarm is triggered
  - The command is not ended

Note

Cyclic continuation is not practicable with active torque limitation. In this case, the torque limitation takes place in the drive, and the setpoints limited by the drive are not known to the control.

3.21.3 Increase limiting for pressure profiles and pressure limiting (V3.2 and higher)

Increase limitation can be specified for **_enable...ForceLimiting()** commands.

Determining the output value for the increase limitation

When determining the output value, it is necessary to ascertain whether the pressure limitation is already active when the **_enable...ForceLimiting()** command is activated.

Note

If you prefer to specify the output value directly:

- Use **_enableForceLimitingValue()** to set the output value

- Wait until the value is reached

- Use a new command to ramp up to the target value

A programmed increase limitation can then be used to move to new values.

Initial activation of the pressure limitation

- Output value when pressure sensor not present is FValue=0.0
- If pressure sensor is activated, the output value is the measured actual pressure value
- When pressure control is triggered by means of pressure limitation, the output value is the pressure setpoint in the IPO

New pressure limitation command with pressure limitation already activated

The output value is the limit value in the IPO.

3.22 Data sets

3.22.1 Data set overview

The configuration data belonging to the axis data set (ADS) can be seen on the **Axis data sets** tab in the Configuration dialog for the axis.
Details of the parameters can be found in the TP Cam Configuration Data Reference List.

ADS contain axis configuration data for the axis, especially that pertaining to the servo (controller data, encoder data, process data, data for axis gearing, and data for force/pressure control, for example).

Data sets must be defined and activated as a group because some data, such as controller data, can only be activated simultaneously in groups to ensure consistency of the controller and function.

3.22.2 Data set switchover / encoder switchover

Data sets can be switched over in the runtime application. This switchover is also possible within one IPO cycle clock (e.g., to activate new controller data or to switch over to a second encoder).

One axis can use several encoders for the position control, but only one encoder at a time.

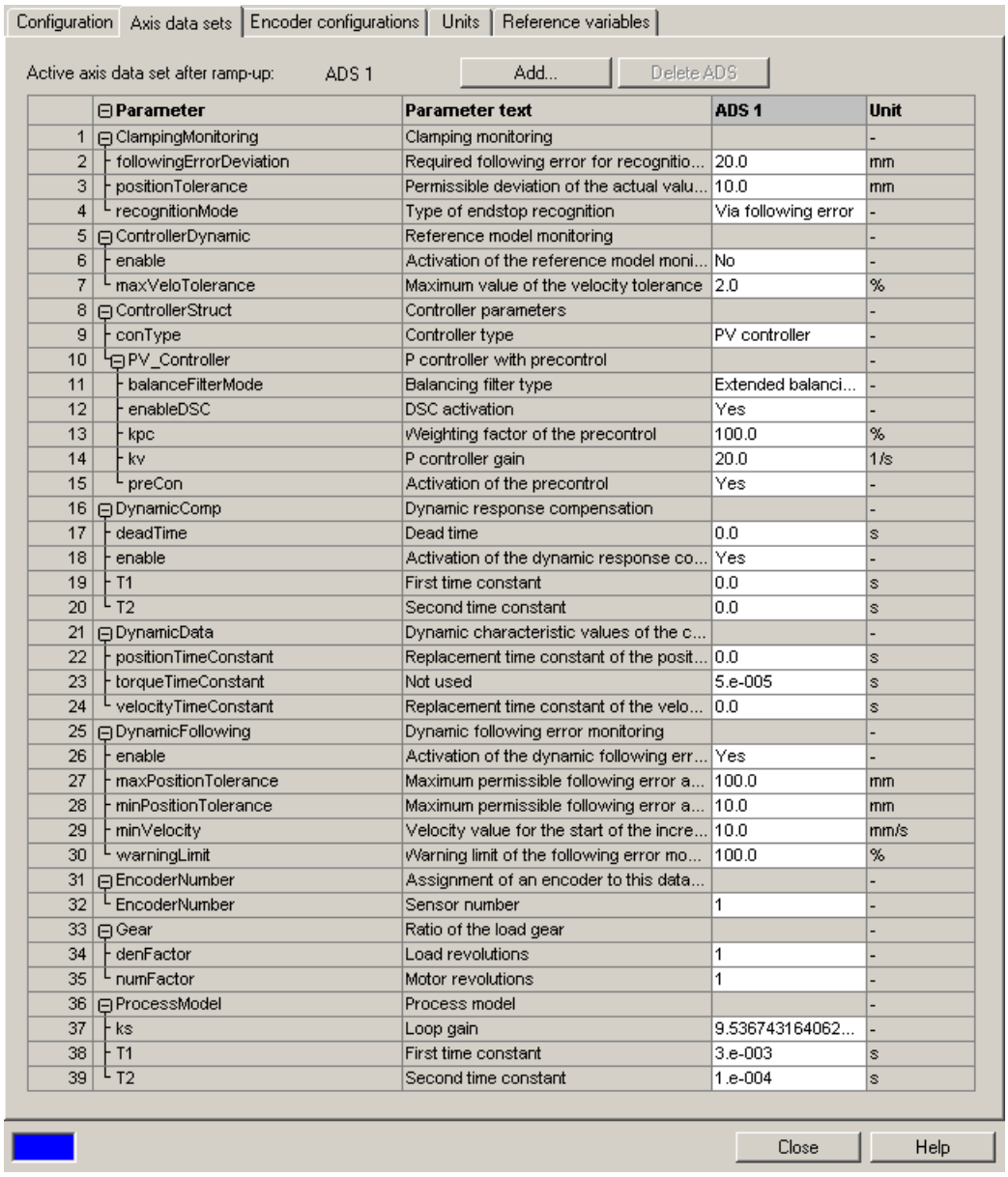
One data set and one encoder are assigned to each axis when it is set up. If you need additional data sets or encoders, they must be added. Data sets can be added in the Axis technology object under **Configuration** on the **Axis data sets** tab.

In V4.0 and higher, the **smoothingTimeByChangeDifference** configuration data element can be used to configure a switchover smoothing filter. This switchover smoothing filter is active for all status transitions/switchovers in which an offset in the manipulated variable can occur due to the switchover. Gear switchover in the data set is not smoothed.

The data sets must have the same control structure and must not contain any change that requires a restart. For example, if data set 1 has DSC, data set 2 must have DSC too, or if following error monitoring is disabled for data set 1, it must also be disabled for data set 2. Data sets with different structures cannot be set up.

Data sets are specified and selected using their numbers.

You can also define which data record is to be loaded for the technology object during CPU power up.



Configuration | Axis data sets | Encoder configurations | Units | Reference variables

Active axis data set after ramp-up: ADS 1 Add... Delete ADS

	Parameter	Parameter text	ADS 1	Unit
1	ClampingMonitoring	Clamping monitoring		-
2	followingErrorDeviation	Required following error for recognitio...	20.0	mm
3	positionTolerance	Permissible deviation of the actual valu...	10.0	mm
4	recognitionMode	Type of endstop recognition	Via following error	-
5	ControllerDynamic	Reference model monitoring		-
6	enable	Activation of the reference model moni...	No	-
7	maxVeloTolerance	Maximum value of the velocity tolerance	2.0	%
8	ControllerStruct	Controller parameters		-
9	conType	Controller type	PV controller	-
10	PV_Controller	P controller with precontrol		-
11	balanceFilterMode	Balancing filter type	Extended balanci...	-
12	enableDSC	DSC activation	Yes	-
13	kpc	Weighting factor of the precontrol	100.0	%
14	kv	P controller gain	20.0	1/s
15	preCon	Activation of the precontrol	Yes	-
16	DynamicComp	Dynamic response compensation		-
17	deadTime	Dead time	0.0	s
18	enable	Activation of the dynamic response co...	Yes	-
19	T1	First time constant	0.0	s
20	T2	Second time constant	0.0	s
21	DynamicData	Dynamic characteristic values of the c...		-
22	positionTimeConstant	Replacement time constant of the posit...	0.0	s
23	torqueTimeConstant	Not used	5.e-005	s
24	velocityTimeConstant	Replacement time constant of the velo...	0.0	s
25	DynamicFollowing	Dynamic following error monitoring		-
26	enable	Activation of the dynamic following err...	Yes	-
27	maxPositionTolerance	Maximum permissible following error a...	100.0	mm
28	minPositionTolerance	Maximum permissible following error a...	10.0	mm
29	minVelocity	Velocity value for the start of the incre...	10.0	mm/s
30	warningLimit	Warning limit of the following error mo...	100.0	%
31	EncoderNumber	Assignment of an encoder to this data...		-
32	EncoderNumber	Sensor number	1	-
33	Gear	Ratio of the load gear		-
34	denFactor	Load revolutions	1	-
35	numFactor	Motor revolutions	1	-
36	ProcessModel	Process model		-
37	ks	Loop gain	9.536743164062...	-
38	T1	First time constant	3.e-003	s
39	T2	Second time constant	1.e-004	s

Close Help

Figure 3-85 Configuration of axis data sets in SIMOTION SCOUT

The individual data sets appear next to each other on the **Axis data sets** tab.

Configuration data which must be the same for all data sets can only be entered in ADS 1. A change in ADS 1 is only mapped in the other ADS if the configuration data has to be set identically for all ADS.

If parameters for axis data sets undergo changes in the expert list, consistency conditions may be violated (check during download). This may happen with controller settings, for example. In this case, the line and cell are highlighted on the **Axis data sets** tab.

Only variables which are effective immediately can be changed online.

The parameter assignment dialog boxes for the Axis technology object (e.g. Control) display the configuration data for one data set at a time. A function bar at the bottom of the dialogs features a combo box 

with the option to switch over the axis data set. This switches over the data set displayed (not the active data set).

Note

Please also note the tooltips above the parameter fields in the individual screen forms for the axis. These show the name of the variable, from which you can also tell whether the variable is assigned to a specific axis data set.

The number of data sets is specified with the structural elements for the **NumberOfDataSets** configuration data element. Up to 16 data sets are possible.

The data set changeover mode is specified in the **NumberOfDataSets.changeMode** parameter.

Table 3-38 Parameter NumberOfDataSets.changeMode

NEVER	No data set changeover takes place.
IN_STANDSTILL	Data set changeover takes place when the standstill signal is applied.
IMMEDIATELY	Data set changeover takes place immediately.
IN_POSITION	Data set changeover takes place when the positioning window is reached.

Switchover to another encoder via the data set switchover

The data set shows which encoder is used with this data set. You can define up to 8 different encoders. All configured measuring systems (encoders) are internally active, and the measured values are updated cyclically.

The encoder contained in the data set is specified and selected using its number (EncoderNumber).

The basic settings for encoders are made under **Configuration** on the **Encoder configuration** tab.

Note

Please also note the tooltips above the parameter fields in the individual screen forms for the axis. These show the name of the variable, from which you can also tell whether the variable is assigned to a specific encoder.

Activating data sets

Data set commands can be used to read, write, and activate data sets on the axis.

The configuration data of a data set are contained in the structure **TypeOfAxis.NumberOfDataSets.DataSet_n** (n: data set number). The display depends on the technology set on the axis.

Data sets must be defined and activated as a group because some data, such as controller data, can only be activated simultaneously in groups to ensure consistency of the controller and function.

- **_getAxisDataSetParameter()** reads a data set on the axis
- **_setAxisDataSetParameter()** overwrites a data set on the axis
- **_setAxisDataSetActive()** sets the data set specified in the function parameter to active

Note

If the active measuring system is switched via a data set changeover, the **_setAndGetEncoderValue()** system function should be used to synchronize the two measuring systems before the switchover takes place. This will prevent unwanted compensating movements of the position controller if differences in position are identified.

See also

Activating data sets (Page 346)

Writing a data set (Page 347)

Reading a data set (Page 347)

Writing the force/pressure-specific data of a data set (Page 347)

Reading the force/pressure-specific data of a data set (Page 348)

Writing the data of the data set (hydraulic functionality only) (Page 348)

Reading the force/pressure-specific data of the data set (hydraulic functionality only) (Page 348)

3.23 Traversing with user-defined motion and force/pressure profiles

3.23.1 Overview of traversing with user-defined motion and force/pressure profiles

In addition to traversing/positioning via system functions, the axis can also be traversed directly via user-defined profiles.

The curve specification of a cam is used as profile function. The cam can be set using the SIMOTION SCOUT engineering system (CamEdit, CamTool) or, alternatively, by means of the application.

A system command is used to assign the appropriate technological variables of the axis to the domain and range.

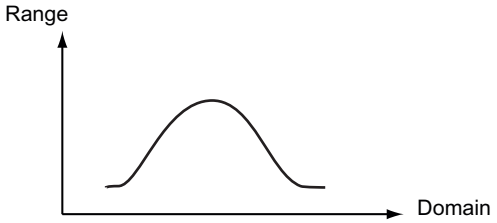


Figure 3-86 Application of cam/technological profiles

A defined start is also possible within a profile.

Application:

Specific adaptation and optimization of traversing specifications.

Additional information about cams can also be found in the SIMOTION CamTool Manual and in the SIMOTION SCOUT online help.

Table 3-39 Possible assignments

Designation	Definition range	Value range	Command
Position-related velocity profile	Position [1) 2)]	Velocity specification	_runPositionLockedVelocityProfile()[1)] _runMotionInPositionLockedVelocityProfile() [2)]
Time-related velocity profile	Time	Velocity specification	_runTimeLockedVelocityProfile()
Time-related position profile	Time	Position	_runTimeLockedPositionProfile()
Time-related force/pressure profile	Time	Force/pressure specification	_runTimeLockedForceProfile()
Position-related force/pressure profile	Position [1) 2)]	Force/pressure specification	_runPositionLockedForceProfile() [1)] _runMotionInPositionLockedForceProfile()[2)]
Position-related force/pressure limiting profile	Position [1) 2)]	Force/pressure limiting	_enablePositionLockedForceLimitingProfile() [1)] _enableMotionInPositionLockedForceLimitingProfile() [2)]
Time-related force/pressure limiting profile	Time	Force/pressure limiting	_runTimeLockedForceLimitingProfile()
Time-related velocity limiting profile	Time	Velocity limitation	_runTimeLockedVelocityLimitingProfile()
Position-related velocity limiting profile	Position [1) 2)]	Velocity limitation	_enablePositionLockedVelocityLimitingProfile() [1)] _enableMotionInPositionLockedVelocityLimitingProfile() [2)]

1) Actual position of axis – the axis position is assigned to the definition range.

2) Interconnected position – the actual position present at the MotionIn interface is assigned to the definition range.

3.23.2 Profile reference

Time-related profiles

The domain of the cam to be used in the command is interpreted as time in the time unit of the axis. The profile can be executed from start to finish or, alternatively, from a starting point that can be predefined.

Position-related profiles with respect to the axis' own position

The domain of the cam to be used in the command is interpreted as the position in the position unit of the axis.

The profile reference is equivalent to the absolute axis position. The profile is started from the current axis position.

The velocity traversing profiles are related to the resulting position setpoint.

The velocity limiting profiles and the force/pressure profiles are related to the actual position value.

Position-related profiles with respect to interconnected position (V3.2 and higher)

The profile is related to the position interconnected via MotionIn connectors.

Table 3-40 Possible commands for reference to a position interconnected via MotionIn

Command	Description
_runMotionInPositionLockedForceProfile()	Force profile related to an interconnected position (MotionIn)
_runMotionInPositionLockedVelocityProfile()	Velocity profile related to an interconnected position (MotionIn)
_enableMotionInPositionLockedForceLimitingProfile()	Force limiting profile related to an interconnected position (MotionIn)
_enableMotionInPositionLockedVelocityLimitingProfile()	Velocity limiting profile related to an interconnected position (MotionIn)

Separate system variables are available for status queries, e.g.:

- **forceMotionInPositionProfileCommand** for force/pressure profiles
- **velocityMotionInPositionProfileCommand** for velocity profiles

3.23.3 Profile types

Velocity profile/velocity limiting profile

The range of the cam to be used in the command is interpreted as the velocity in the velocity unit of the axis. The motion direction, acceleration, and jerk are calculated accordingly.

A transition profile is generated by the axis for discontinuous transitions. The dynamic response parameters of this profile are specified using the command parameters for acceleration and jerk.

Position profile

The range of the cam to be used in the command is interpreted as the position in the position unit of the axis. The motion direction, position, acceleration, and jerk are calculated from this relationship.

The position reference to the axis can be selected as either absolute or relative.

A transition profile is generated by the axis for discontinuous transitions. The dynamic response parameters of this profile are specified using the command profile parameters for acceleration, jerk, and velocity.

Force/pressure profile and force/pressure limiting profile

The range of the cam to be used in the command is interpreted as the force/pressure in the pressure unit of the axis. The derivation of force/pressure for any necessary transition motions (for entering the profile and exiting the profile, for example), can be programmed in the command. The behavior at the end of the profile is set during axis configuration.

Limitation of the acceleration and braking ramps (V3.2 and higher)

The **DecodingConfig.profileDynamicsLimiting** configuration data element is used to determine whether the permitted acceleration and braking ramps should be limited by the programmed values (**derivedCommandValue**) or the maximum values (configuration data).

- Limiting by programmed values:
  The axis is ramped up to the profile or to the values specified in the motion vector (MotionIn) using the minimum value from the dynamic values defined in the command and the maximum values set in MaxJerk, MaxAcceleration, and MaxVelocity.
  While traversing on the profile, the axis is limited to the lowest value of the programmed values and the set maximum values.
- Limiting by maximum values:
  The axis is ramped up to the profile or to the values specified in the motion vector (MotionIn) using the minimum value from the dynamic values defined in the command and the maximum values set in MaxJerk, MaxAcceleration, and MaxVelocity.
  While traversing on the profile, the axis is only limited to the set maximum values.

Status display during profile processing (V3.2 and higher)

During profile processing, extended states are displayed in the system variables for the profiles, e.g. in **positionTimeProfileCommand.processingState**.

The PROFILE_END status is set if the end of profile has already been reached, but the command is still active.

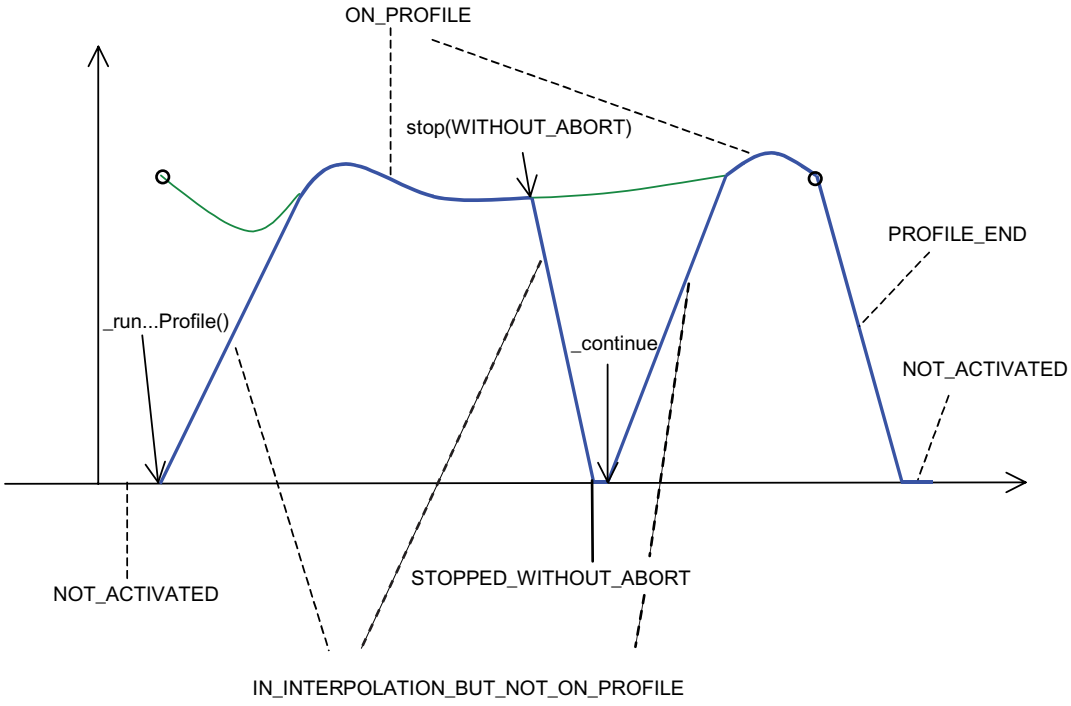


Figure 3-87 Status displays when traveling through a profile (example)

See also

Positioning with user-definable position profile (Page 333)

Enable force/pressure limiting with position-related force/pressure limiting profile (Page 334)

Enabling force/pressure limiting with time-related motion profile (Page 335)

Starting the time-related force/pressure profile (Page 335)

Starting the position-related force/pressure profile (Page 336)

Starting a time-related velocity profile (Page 331)

Starting a position-related velocity profile (Page 332)

Enabling velocity limiting with position-related velocity limiting profile (Page 338)

Enabling velocity limiting with time-related velocity limiting profile (Page 339)

3.23.4 Behavior at the end of the profile (V3.2 and higher)

The **decodingConfig.behaviourAtTheEndOfProfile** configuration data element is used to configure various behaviors.

Table 3-41 Assignable behaviors

Setting	Meaning
MOVE_WITH_CONSTANT_SPEED	Constant continuous motion • Position/velocity: Hold value • Force/pressure: Hold value
STOP_IN_PROFILE_END	Decelerate to target point/velocity 0 • Position/velocity: Velocity 0 at the end of profile (via ramp) • Force/pressure: Value at the profile end point (= Hold value)
STOP_WHEN_PROFILE_END_REACHED	Decelerate after completed traverse • Position/velocity: Velocity 0 when end of profile reached (via ramp) • Force/pressure: Value at the profile end point (= Hold value)

Changes can be made online and take effect immediately.

Note

For position-related profiles with the setting STOP_IN_PROFILE_END, it should be remembered that the velocity calculated by the system can cause vibration.

3.24 Motion commands

3.24.1 Motion execution/interpolator

Axis motion is executed in the interpolator, in the servo, and in Servo_fast.

The servo for all axes is calculated in the position-control cycle clock.

The reference variables are calculated in the interpolator. The interpolator cycle clock of the device is set during the configuration of the execution system. There are two interpolator levels in the system, **IPO** and **IPO2**.

The processing cycle clock (axis-specific interpolator cycle clock) of the Axis technology object can be set to **IPO** or **IPO2**.
For the possible setting **IPO_fast**, see the chapter titled *Second position control cycle clock (Servo_fast)* in the *Motion Control Basic Functions* manual.

This makes it possible to place an interpolator for axes that do not require a high time resolution to calculate reference variables in a cyclical system task with a longer cycle time, thereby requiring less processing power.

Setting for shorter axis reaction times (e.g. faster start of motion) (as of V4.1 SP1)

The **Execution.executionLevel=SERVO** or **processing clock** = Servo setting in the configuration screen form can be used to execute the IPO system component of the axis in the servo following actual value measurement. After the actual value system in the servo, the IPO system functionality is calculated first, followed by the controller and the setpoint system.

This reduces the response time to the switching of external signals or synchronous operation in the control to one servo cycle clock.

Note

Due to the higher performance requirement, this setting should only be used for selected (i.e. a limited number of) axes.

Temporarily high IPO system components in the servo of the axis can result in a level overflow in the servo, thus causing the CPU to go to STOP mode.

For the possible setting **Servo_fast**, see the chapter titled *Second position control cycle clock (Servo_fast)* in the *Motion Control Basic Functions* manual.

System and user tasks

In addition to the technology object system tasks of ServoTask, IPOTask, and IPOTask_2, there are also synchronous user tasks called ServoSynchronousTask, IPOSynchronousTask, and IPOSynchronousTask_2.

How the system and user tasks interact with each other is demonstrated in the following example for the Axis technology object. At the end of cyclic data transfer, the ServoSynchronousTask and ServoTask are started, followed by the IPOSynchronousTask and IPOTask. Axis data which is active in the servo (e.g. superimposed setpoint or manipulated variable value) can be modified in the ServoSynchronousTask. In the IPOSynchronousTask, issued motion commands are processed directly in the subsequent IPOTask (e.g. switchover to superimposed motion or registration mark).

Because the ServoSynchronousTask is executed before the IPO system component in the servo level even if **execution.executionLevel=SERVO** is set, this enables a very rapid response with motion affected at user level as well.

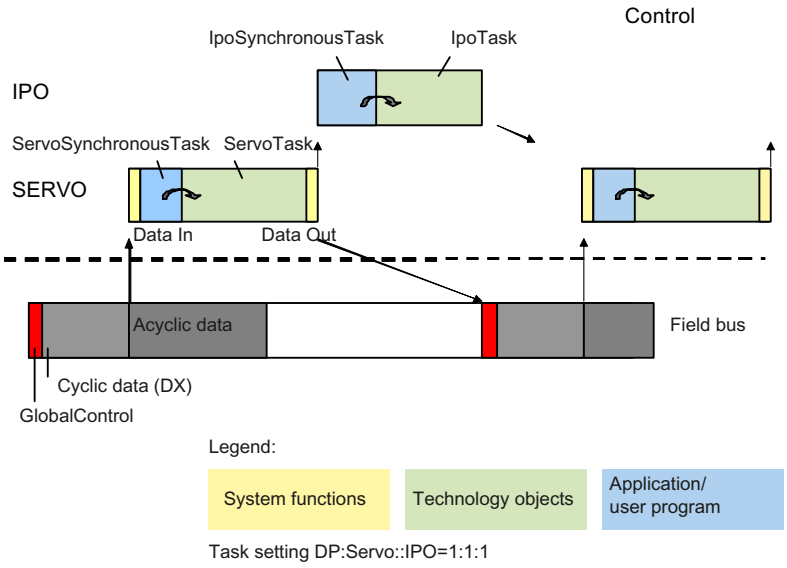


Figure 3-88 Example of cycle clock settings

See also the chapters titled *Execution system/Tasks/System cycle clocks* and *Dynamic response to data processing in the control* in the *Motion Control Basic Functions* Function Manual.

3.24.2 Command groups

Commands on the axis are divided into command groups to enable multiple commands to be active in one **IPO** cycle clock. Commands on the axis are read in and processed in the IPO cycle clock. If the user program issues more than one command for a command group within an interpolator cycle clock and the user program can run, for example, in another task, then a specific behavior is defined for the command group. The commands assigned to a command group are implicitly assigned to the corresponding command buffer.

The overwritten commands trigger technological alarm "030002 Command aborted".

Command buffers and their properties

Every axis has command buffers that can buffer one command each. These buffers are read out in every interpolator cycle clock.

- Buffer[1)] for **_stopEmergency()**, **_stop()**, and **_continue()** commands
  This buffer contains **_stopEmergency()**, **_stop()** without command abort, and **_continue()**. The procedure followed when motions stop depends on the command priorities. Higher priorities are indicated by larger numbers. A command with the same or higher priority displaces a command already in the buffer.
  - **Priority 1: _stop()** without command abort and **_continue()**
  - **Priority 2: _stopEmergency()** with time ramp
  - **Priority 3: _stopEmergency()** with maximum deceleration
  - **Priority 4: _stopEmergency()** with setpoint zero
- Buffer[2)] for **Enable-** and **Disable** commands
  This buffer contains the enable and disable commands. The commands displace each other from the command buffer.
- Buffer[3)] for **superimposed** commands
  Commands that are executed in parallel or superimposed to a main motion are, for example:
  - Motion commands with mergeMode=SUPERIMPOSED_MOTION_MERGE
  - **_redefinePosition()**
  - **_enableAxisAdditiveTorque()**
  - **_homing()** with homingMode=DIRECT_HOMING or PASSIVE_HOMING

  The following applies to SIMOTION < V4.4:
  The buffer has an entry. These commands overwrite each other if they are issued within one IPO cycle clock.
  As of SIMOTION V4.4, the following applies:
  A maximum of 10 command entries are possible in the command buffer for superimposed commands. The number can be set in the **DecodingConfig.lenghtOfBufferForSuperimposedCommands** configuration data element. When a new axis is created, five entries are preset by the system.
- Buffer[4)] for **overriding** and **sequential** commands
  Acceptance of all commands, in particular motion commands that have been programmed as sequential motion using the mergeMode=SEQUENTIAL function parameter or as overriding motion using the mergeMode=IMMEDIATELY function parameter.
  In the case of sequential motions with immediate command advance (nextCommand=IMMEDIATELY and mergeMode=SEQUENTIAL), new commands to be entered can return with errors if the buffer is full.
  In the case of sequential motions with immediate command advance, if the motion command can be accepted by the system (nextCommand=WHEN_BUFFER_READY and mergeMode=SEQUENTIAL), the command does not advance until the motion command has been accepted by the system.
  In the case of mergeMode=NEXT_MOTION or mergeMode=IMMEDIATELY, the command can always be accepted by the system, because the next motion command in the command buffer or the current motion command in the interpolator is overwritten.

The following table shows the assignment of axis commands to command buffers.

Table 3-42 Allocation of commands to the command buffers

Command	Position
_continue() Resume motion	1
_stop() Stop a motion without traversing command abort	1
_stop() Stop a motion with traversing command abort	4
_stopEmergency() Stop the motion with cancellation of the motion commands without further motion commands having any affect	1
_disableAxis() Remove axis enable	2
_disableQFAxis() Remove enable for axis with hydraulic functionality	2
_enableAxis() Set axis enable	2
_enableQFAxis() Set enable for axis with hydraulic functionality	2
_adaptAxisConfigData() Adapts the characteristic data for motors and/or encoders	3
_disableAxisAdditiveTorque() Deactivate additive set torque	3
_disableAxisSimulation() Disable simulation mode	3
_disableForceLimiting() Deactivate force/pressure limiting	3
_disableMovingToEndStop() Disabling command for travel to fixed endstop	3
_disableTorqueLimiting() Deactivate torque limiting	3
_disableAxisInterface() Deactivate actuator interface	3
_disableAxisTorqueLimitNegative() Activate negative torque limiting	3
_disableAxisTorqueLimitPositive() Activate positive torque limiting	3
_disableMonitoringOfEncoder() Deactivate differential encoder monitoring	3
_disableVelocityLimiting() Deactivate velocity limiting	3

Command	Position
_enableAxisAdditiveTorque() Activate additive set torque	3
_enableAxisInterface() Activate actuator interface	3
_enableAxisSimulation() Enable simulation mode	3
_enableAxisTorqueLimitNegative() Activate negative torque limiting	3
_enableAxisTorqueLimitPositive() Activate positive torque limiting	3
_enableForceControlByCondition() Switch over to p(t) profile with switchover criterion	3
_enableForceLimitingByCondition() Switchover to force/pressure limiting when the switchover criterion defined in the command is reached	3
_enableForceLimitingValue() Activate force/pressure limiting with constant force/pressure limit value	3
_enableMonitoringOfEncoder() Activate differential encoder monitoring	3
_enableMotionInPositionLockedForceLimitingProfile() Force limiting profile related to an interconnected position (MotionIn)	3
_enableMotionInPositionLockedVelocityLimitingProfile() Velocity limiting profile related to an interconnected position (MotionIn)	3
_enableMovingToEndStop() Enabling command for travel to fixed endstop	3
_enablePositionLockedForceLimitingProfile() Activate force/pressure limiting with p(s) profile	3
_enablePositionLockedVelocityLimitingProfile() Activate velocity limiting with position-related limiting profile	3
_enableTimeLockedForceLimitingProfile() Activate force/pressure limiting with time-related limiting profile	3
_enableTimeLockedVelocityLimitingProfile() Activate velocity limiting with time-related limiting profile	3
_enableTorqueLimiting() Activate torque limiting	3
_enableVelocityLimitingValue() Activate velocity limiting	3
_redefinePosition() Reset actual and set position	3
_setAndGetEncoderValue() Synchronize measuring systems	3
_setAxisDataSetActive() Switch over data set	3

Command	Position
_homing() Homing	3 4 homingMode = ACTIVE_HOMING
_move() Rotate	4 3 for mergeMode = SUPERIMPOSED
_pos() Positioning	4 3 for mergeMode = SUPERIMPOSED
_runMotionInPositionLockedForceProfile() Force profile related to an interconnected position (MotionIn)	4 3 for mergeMode = SUPERIMPOSED
_runMotionInPositionLockedVelocityProfile() Velocity profile related to an interconnected position (MotionIn)	4 3 for mergeMode = SUPERIMPOSED
_runPositionBasedMotionIn() Activate MotionIn interface with position reference	4 3 for mergeMode = SUPERIMPOSED
_runPositionLockedForceProfile() Run a p(s) profile	4
_runPositionLockedVelocityProfile() Apply a v(s) profile	4 3 for mergeMode = SUPERIMPOSED
_runTimeLockedForceProfile() Run a p(t) profile	4
_runTimeLockedPositionProfile() Apply an s(t) profile	4 3 for mergeMode = SUPERIMPOSED
_runTimeLockedVelocity() Apply a v(t) profile	4 3 for mergeMode = SUPERIMPOSED
_runVelocityBasedMotionIn() MotionIn interface with velocity reference activated	4 3 for mergeMode = SUPERIMPOSED
_setForceCommandValue() Set force/pressure setpoint	4
_bufferAxisCommandId() Store command status permanently	5
_cancelAxisCommand() Abort command with the specified CommandID	5
_getAxisDataSetParameter() Read a data set	5
_getAxisErrorNumberState() Read out error number status	5

Command	Position
_getAxisErrorState() Read out error status, alarm number and alarm parameter of up to eight pending alarms	5
_getAxisInternalPosition() Convert from axis coordinates to encoder position values	5
_getAxisStoppingData() Calculate braking distance depending on a specified actual velocity, actual acceleration, traversing profile and dynamic response parameters	5
_getAxisUserPosition() Convert from encoder positions to axis coordinate system	5
_getForceControlDataSetParameter() Read a data set	5
_getMotionStateOfAxisCommand() Read out motion phase of a command	5
_getProgrammedTargetPosition() Display programmed absolute end position including superimposition	5
_getStateOfAxisCommand() Read out command status	5
_getStateOfMotionBuffer() Shows whether the command queue can be filled	5
_getQFAxisDataSetParameter() Read specific parameters for the axis with hydraulic functionality in the data set	5
_removeBufferedAxisCommandId() End permanent storage of the command status	5
_resetAxis() Reset axis	5
_resetAxisConfigDataBuffer() Clear the configuration data in the buffer without activation	5
_resetAxisError() Reset error on the axis	5
_resetMotionBuffer() Reset command queue	5
_setAxisDataSetParameter() Overwrite data set	5
_setAxisSTW() Specify control bits in STW1 directly	5
_setForceControlDataSetParameter() Overwrite data set	5
_setQFAxisDataSetParameter() Overwrite the specific parameters for the axis with hydraulic functionality in an inactive data set	5

Command	Position
_setQFAxisPCharacteristics() Set characteristics for P value	5
_setQFAxisQCharacteristics() Set characteristics for Q value	5

1) Buffer for emergency stop and stop-continue commands

2) Buffer for enable and disable commands

3) Buffer for superimposed commands

4) Buffer for overriding and sequential commands

5) Not allocated to the command buffers; commands are executed synchronously as called

3.24.3 Changing motion commands into the interpolator

A **DecodingConfig.decodeSequentialMotionCommand** configuration data item can be used to set when the next **SEQUENTIAL** or **NEXT_MOTION** command is switched in or executed in the interpolator - immediately in the same cycle clock or in the next **IPO** cycle clock.

- If the setting is **IMMEDIATELY** (default), the next command is switched in immediately and started in the same cycle clock once interpolation/execution of the current command in the IPO cycle clock is complete.
- If the setting is **NEXT_IPO_CYCLE**, the next command is not changed into the interpolator until the previous command has been completely interpolated.
  This prevents execution of multiple motion commands in the interpolator, thus preventing a greater-than-average interpolator load from being called forth in one cycle clock.

3.24.4 Motion transitions

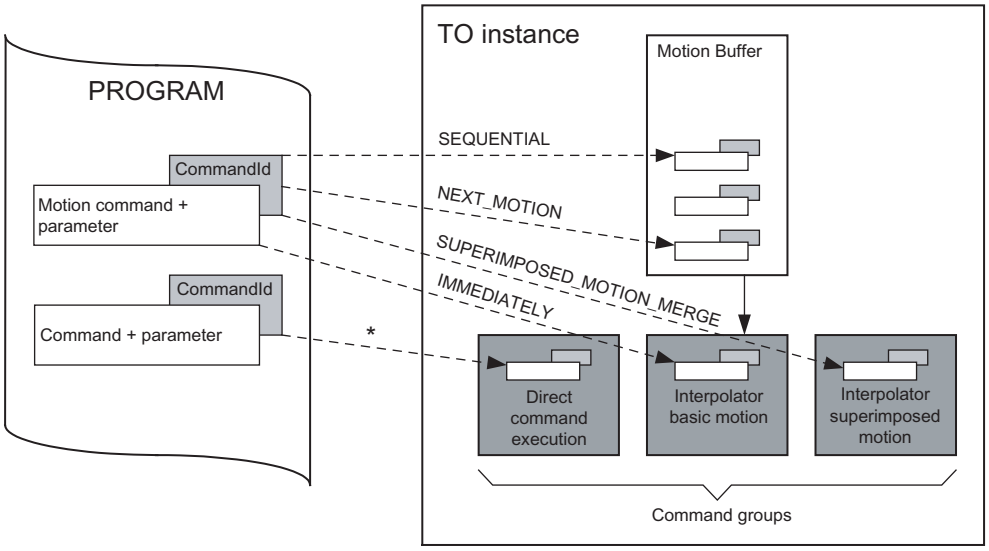
The behavior during a transition between two motions is defined by **mergeMode**.

The programmed motion transitions are decisive for the active motion. The priority of the task in which the motion command is issued does not affect the priority assignment of the command.

Definable motion transitions

- **IMMEDIATELY** (substitute)
  The motion specified with the command is activated immediately. Motions which are already active are overridden and pending commands/motions are aborted.
- **NEXT_MOTION** (attach, delete pending command)
  Execute after the active motion and delete further pending commands/motions.

- **SEQUENTIAL** (attach)
  Append to previous commands/motions.
- **SUPERIMPOSED_MOTION_MERGE** (superimpose)
  Superimposed motion on the axis is possible in addition to the basic motion



* Not allocated to the command buffers; commands are executed synchronously as called (See also allocation 5) in the table titled "Allocation of commands to the command buffers)

Figure 3-89 Command reactions in the Axis technology object

3.24.5 Conditions for command advance

If the condition for the command advance is satisfied, the next command in the user program is executed. Specifying a step enabling condition affects the **execution time** of the next command in the same user task.

- **IMMEDIATELY**
  As soon as the command has been issued, irrespective of whether or not the commanded motion can be executed
- **WHEN_BUFFER_READY**
  After the command has been entered in the motion buffer
- **AT_MOTION_START** (motion start)
  After the command has been changed into the interpolator
- **WHEN_ACCELERATION_DONE** (end of acceleration)
  When the acceleration phase is completed
- **AT_DECELERATION_START** (start of deceleration phase)
  When the deceleration phase starts
- **WHEN_INTERPOLATION_DONE** (end of setpoint interpolation)
  When the setpoint interpolation for this command is completed
- **WHEN_MOTION_DONE** (positioning window reached)
  When the setpoint interpolation is complete and the configured positioning window has been reached

- **WHEN_COMMAND_DONE** (when command is completed or aborted)
  After completion of the command, for example, for commands that require time for their execution, but do not contain a motion element
- **WHEN_TORQUELIMIT_REACHED** (as soon as the torque is limited)
  When torque limiting is triggered
- **WHEN_TORQUELIMIT_GONE** (as soon as the torque limiting is switched off)
  When torque limiting is removed
- **WHEN_LIMITING_COMMAND_ACTIVATED** (when the velocity limiting command is activated)
  When velocity limiting is activated
- **WHEN_LIMIT_REACHED** (as soon as the velocity is limited)
  When velocity limiting is triggered
- **AT_PROFILE_START**
  Beginning of interpolation with profile
- **BY_PROFILE_END**
  End of setpoint interpolation with profile
- **WHEN_AXIS_HOMED** (when axis is homed)
  Axis was homed
- **WHEN_ENDSTOP_REACHED** (when clamping value is reached)
  Clamping value reached (travel to fixed endstop)
- **WHEN_FUNCTION_DISABLED** (when command is aborted or completed)
  Command is completed or aborted (travel to fixed endstop)

3.24.6 State model/axis status

Table 3-43 The axis status is indicated in:

• Axis inactive/can be activated:	control= INACTIVE
• Axis active:	control= ACTIVE
• Motion:	motionStateData.motionCommand= IN_MOTION
• Fault:	error= YES and ErrorReaction <> NONE
• StopEmergency:	stopEmergencyCommand.state= ACTIVE

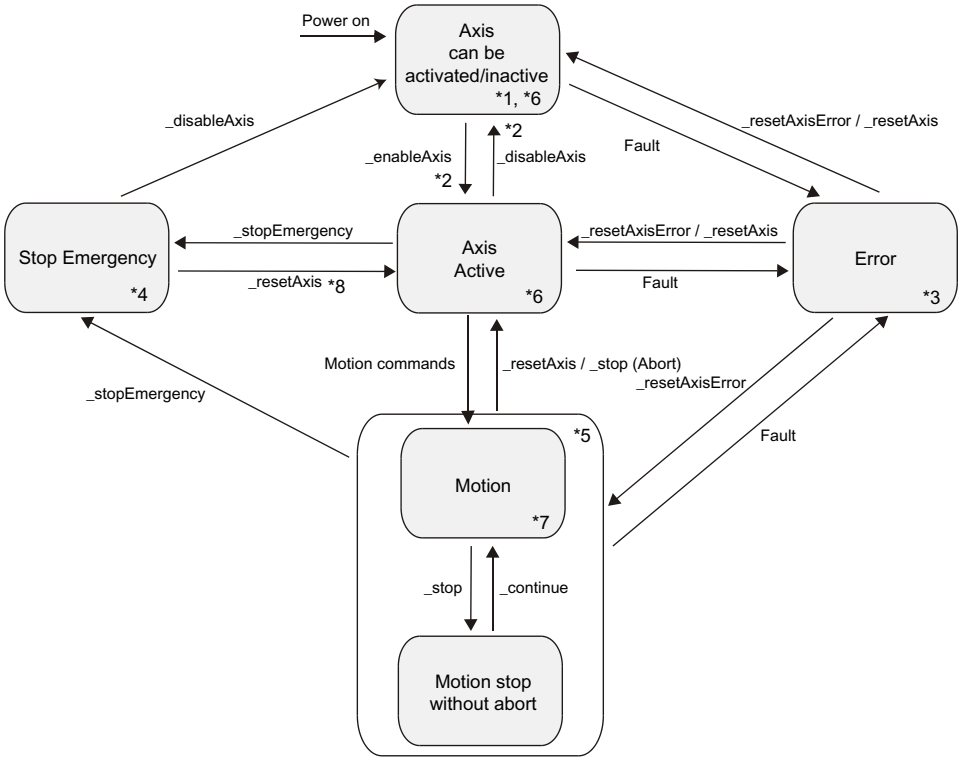


Figure 3-90 State model/axis status

When the axis is configured as an axis with hydraulic functionality (QFAxis) in **TypeOfAxis**, **_enableAxis()** must be replaced with **_enableQFAxis()** and **_disableAxis()** must be replaced with **_disableQFAxis()** in the state model for the axis.

The status of **_stopEmergency()** is displayed in the **stopEmergencyCommand** system variable. Following a reset, this variable must be scanned explicitly by means of **_disableAxis()**.

Table 3-44 The following commands and functions are active in the individual states of the axis:

***1** Axis can be enabled/is inactive with respect to motion control of the axis; also includes the disabled state

***2** For active drive, output and control:
- Switch from follow-up mode or to follow-up mode
- Enable/disable the control

***3** Depending on the stop or error states, **_disableAxis()** / **_disableQFAxis()** and **_stopEmergency()** are permitted

***4** Faults are handled according to existing criteria. Higher-priority local stop reactions can be handled.

***5** Superimposed motions and motions on the following axis are included.

***6** **resetAxis()** permitted

***7** Motion commands permitted

***8** If axis is in position control

Axis in inactive/can be activated state

When the control is switched on, the technology object goes into follow-up mode. In that case:

- All commands in the motion buffer are deleted
- The axis and controller are inactive
- System variables are set either to the configured values or to start values
- The motions commanded by motion commands are not executed; the axis is in follow-up mode
- The **_enableAxis()** or **_enableQFAxis()** command enables the controller and switches the technology object state to active
- Changing from STOP U operating state to RUN, and vice versa, has no effect on the Axis technology object
- During the transition from STOP to STOP U operating state, all alarms that can be reset are acknowledged and the motion buffer is cleared
- Encoders/actual value system is/are active in the STOP operating state
  The axis position is retained, unless an alarm is triggered.

Axis in active state

Motion commands can be issued and executed in this state.

Axis in motion state

- Motion jobs are executed and motion commands can be issued in this state.
- Motion can be stopped by sending the **_stop()** command, stopMode= STOP_WITHOUT_ABORT setting, and resumed by sending the **_continue()** command
- Motion is aborted with the **_stop()** command, stopMode= STOP_AND_ABORT

Axis in fault technology object state

In fault technology object state, the following actions are possible:

- Actions that reset the fault state
- Actions that cause a higher-priority stop reaction
- Actions that do not affect the state
- Actions that are generally permitted according to criteria for technological alarms

Motion jobs and the simulation command **_enableAxisSimulation()** are not executed.

The fault is remedied with the **_resetAxis()** or **_resetAxisError()** command if these commands are permitted for the fault in question.

The status commands **_getStateOfAxisCommand()** and **_getStateOfMotionBuffer()** are permitted, as are the reset commands **_resetMotionBuffer()** and **_disableAxis()** or **_disableQFAxis()**.

Aborted motion commands trigger a **technological alarm** and a notice in the alarm window.

The reactions on the axis can be configured in the technological alarm.

Note

Errors can be acknowledged in SIMOTION SCOUT, the program or the operating panel.

You can find additional information under Error handling in the *Motion Control Basic Functions* Function Manual.

Axis in StopEmergency state

The **_stopEmergency()** command

- Does *not* cause the alarm task to start immediately
- Does *not* disable the axis and drive
- Is *not* active if the axis is in follow-up mode
- Is *not immediately* active if a following error has built up during active torque limitation
- Is *not immediately* active if a pressure axis switches from closed-loop pressure control to pressure limitation

A **_stopEmergency()** command with a higher-priority stop reaction cancels a lower-priority reaction.

The **_stopEmergency()** command generates the **_stopEmergency_Status**. This status can be canceled with the **_disableAxis()** / **_disableQFAxis()** or **_resetAxis()** commands.

Commands active in the interpolator are aborted.

3.25 Data exchange between Axis technology object and DCC

For data exchange between the Axis technology object and DCC charts/DCC blocks, system variables of the technology object can be interconnected directly with block inputs and block outputs.

The update/effectiveness of the system variables is specified in the reference lists.

- Cyclically updated display data in the IPO include:
  - **motionStateData.actualVelocity**
  - **motionStateData.actualAcceleration**
  - **positioningState.actualPosition**
- Cyclically effective system variables in the IPO include:
  - **defaultMotionIn.x** (when activated by means of a command)
  - **defaultMotionIn.y** (when activated by means of a command)
  - **defaultMotionIn.z** (when activated by means of a command)
  - **override.velocity**
  - **override.acceleration**
  - **plusLimitsOfDynamics.velocity**
  - **plusLimitsOfDynamics.postiveAccel**
  - **plusLimitsOfDynamics.negativeAccel**
- Cyclically updated display data in the servo includes:
  - **sensorData[n].position**
  - **sensorData[n].velocity**
  - **sensorData[n].acceleration**
  - **sensorData[n].incrementalPosition**
- Cyclically effective system variables in the servo include:
  - **servoSettings.additionalCommandValue**
  - **servoSettings.additionalSetpointValue**

You can find additional information about DCC in the *Motion Control Basic Functions* Manual.

3.26 Drive communication based on DPV1 services

Drive communication based on PROFIdrive provides the basis for the digital SIMOTION/ SINAMICS drive system.
The focus is primarily on cyclic services. Acyclic DPV1 services also provide the option of on-demand data transfer, e.g. to set various parameters or options for slave devices during operation.

The LDPV1 library provides the basis for the quick and easy use of DPV1 services in applications.

Functions supported:

- Buffer management
- Time-of-day synchronization SIMOTION-SINAMICS
- Ramp-up coordination
- Assignment of SIMOTION technology objects to SINAMICS objects

- Reading of SINAMICS errors and warnings
- Activation and deactivation of technology objects and drive objects
- Reading and writing of SINAMICS parameters
- Functions for SINAMICS Safety Integrated

You can find the LDPV1 library in the *SIMOTION Utilities & Applications* included in the scope of delivery of SIMOTION SCOUT (under **Applications > Cross-sector applications > Drive communication**).

Each individual function supported is accompanied by documentation describing the actual application.

Note

As of SIMOTION firmware V4.2, time-of-day synchronization between SIMOTION and SINAMICS is automatic.

3.27 Links/interconnections to other technology objects

If you interconnect technology objects with one another, they must be configured in the same execution level - with the exception of synchronous axes.

If you need to connect a technology object in a different execution level with an axis, you can work with a virtual synchronous axis that is in the desired execution level and is connected to the actual axis. You can generally use this to evaluate cyclic data of an axis in other levels also (e.g. from the user program).

Note

Further information is available under Interconnection of technology objects in the Motion Control Basic Functions Function Manual.

Configuring an axis

4

4.1 Overview of axis configuration

Axes are configured using an axis wizard,
which is started automatically when a new axis is created. In order to change the axis settings, the configuration can be launched using *Configure displayed data set...* in the *Configuration* axis parameter assignment dialog box. Important axis parameters and drive connections are configured in the axis wizard.

Data sets are also managed in the *Configuration* axis parameter assignment dialog for the purpose of data set changeover and encoder configuration.

For more information, see Data sets (Page 200).

You can specify other selected parameters via the axis parameter assignment dialog boxes, which can be accessed under Axis Object in the project navigator. You can also use the expert list to access all configuration data and system variables relating to the Axis technology object in list format.

System default settings of the configuration data

The system default information shows the settings of the configuration data, which are backward compatible (can also be used for older versions). These are documented in the reference lists for the configuration data.
When a technology object is created or when drives are assigned, the system may make other technological settings than the system defaults in order to support system optimizations or functions of the new versions for the respective context.

4.2 Linking digital drives

4.2.1 Overview of linking digital drives

The Assign dialog for the drive link is included in the axis wizard when setting up an axis. Once an axis has been set up, this dialog can also be called via the **...** button in the **Configuration** axis dialog or from the address list (**All addresses** view). Drives can also be assigned symbolically, see Coupling of digital drives (Page 56).

Digital drives are linked to the SIMOTION Axis technology object via a PROFIdrive message frame.

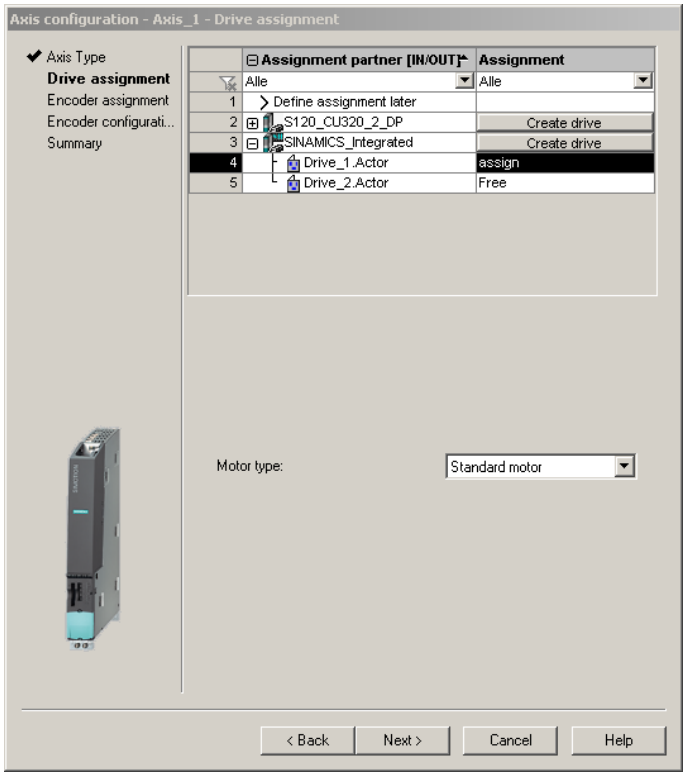


Figure 4-1 SINAMICS drive assignment

The dialog for drive assignment supports both immediate and retrospective drive assignment. Furthermore, a new drive can be created and assigned to the axis. See also Coupling of digital drives (Page 56).

4.2.2 Configuring PROFIBUS DP in HW Config to optimize run-time

The configuration of PROFIBUS DP in HW Config to optimize run-time is described in the **Motion Control Basic Functions** manual under *Configuring PROFIBUS DP in HW Config to optimize run-time*.

4.3 Linking analog drives to SIMOTION

Connection of any drive with an analog setpoint interface to a SIMOTION device (e.g. C2xx).

Axis - analog drive relationship

For an axis with an analog drive, the speed setpoint is applied to the analog output as a voltage signal, and the actual value is read in via an encoder interface.

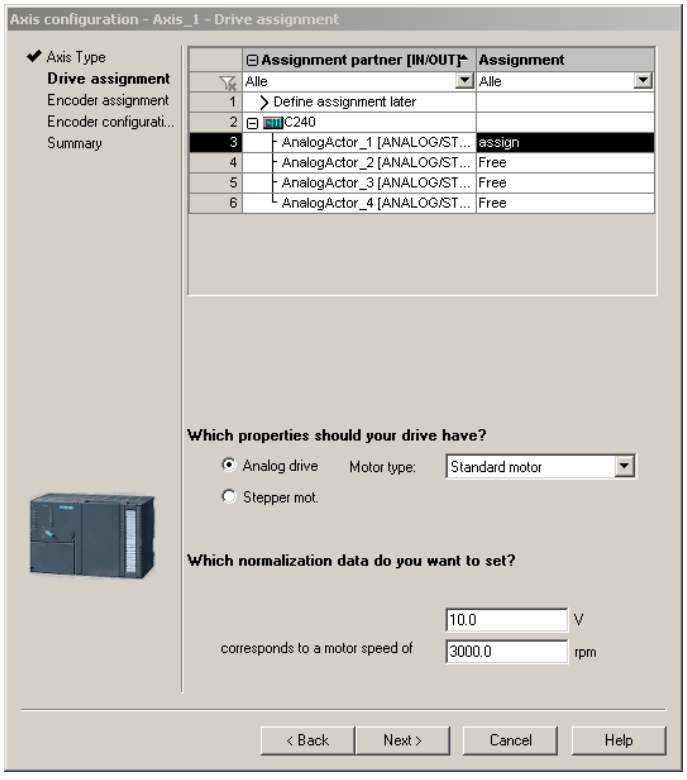


Figure 4-2 Drive assignment of analog drives

Drive

A drive is controlled with +/- 10 V via an **analog interface**.

Axis-drive connection

An axis is connected to a drive by entering the **logical address of the selected analog output** of the SIMOTION device when the axis is configured. Use the axis configuration wizard to do this.

If you change the hardware configuration in **HW Config** at a later time, the specified logical addresses may become incorrect. In this case, repeat the configuration in the hardware configuration wizard.

Make sure that the Volt >> Speed setting in the drive matches the setting in the axis.

ADI4/IM174

In addition to the option of operating analog axes on the onboard inputs of the C2xx, the ADI4 and IM 174 modules are available for use in all platforms as interfaces for analog drive connections. From the SIMOTION perspective, these modules behave like digital drive links. See also Coupling of digital drives (Page 56) for more information.

You can find further information in the *ADI4 - Analog drive interface for 4 axes* and *Distributed I/O, PROFIBUS module IM174* manuals.

See also

Coupling of digital drives (Page 56)

Setting as a real axis with analog drive link (Page 38)

4.4 Axis with stepper motor connection

Connection of a stepper drive with a pulse/direction interface to a SIMOTION device (e.g. C2xx).

For an axis with stepper motor connection, there is one clock pulse, directional signal, and enable signal output per axis.

For a link via a bus system (e.g., to IM174 via PROFIBUS DP), the general information for configuring PROFIdrive drives is applicable, with special consideration of the behavior of a stepper motor (see Stepper drives (Page 143)).

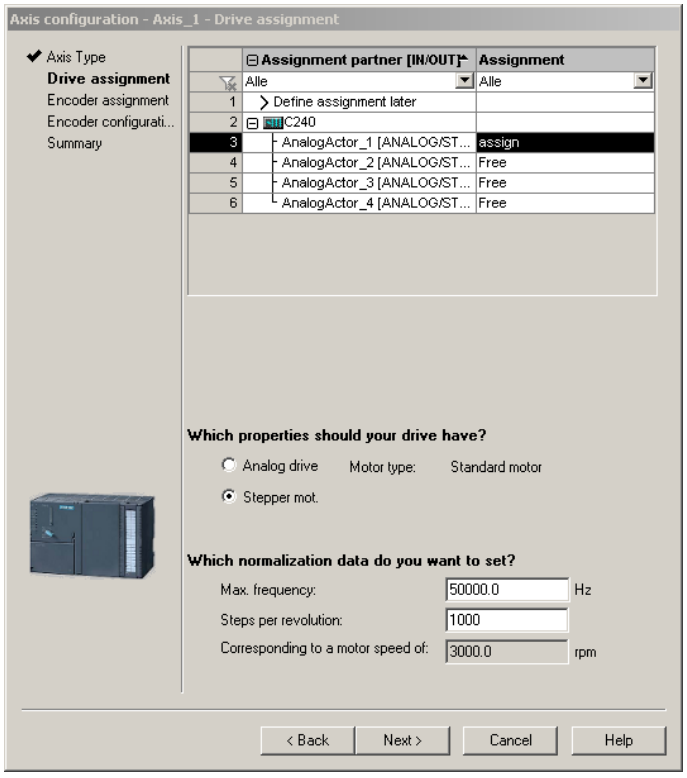


Figure 4-3 Drive assignment of the stepper motor

An axis with stepper motor connection can be moved either with or without an encoder. With an encoder, the actual value signals are read in via the encoder interface; without an encoder, the actual value information is generated from the output motor pulses.

Note

The product of the maximum frequency of the stepper motor and the reciprocal of the number of steps per motor revolution gives the maximum speed available to the control loop (corresponds to maxSpeed for conventional drives).

Note

When an encoderless stepper motor connection is configured on a position axis, an encoder input is still automatically reserved and cannot be used for an external encoder, for example.

With a drive axis, the encoder interface is available for and can be assigned to an Axis or External Encoder technology object.

For operation without an encoder, additional data can be specified in the configuration data for rotation monitoring.

- **NumberOfEncoders.Encoder_1.stepMotorMonitoring.enable** for enabling/disabling the monitoring function
- **NumberOfEncoders.Encoder_1.stepMotorMonitoring.beroCycleDistance** for setting the permitted deviation in terms of motor steps per revolution
- **NumberOfEncoders.Encoder_1.stepMotorMonitoring.beroCycleTolerance** for setting a tolerance range around the beroCycleDistance

Rotation monitoring can be activated for operation without an encoder:

An external zero mark is used and is connected to the associated external zero mark input of the axis channel.

For a linear axis, the external zero mark must monitor the rotation of the motor shaft.

If a signal is not received within the specified motor steps + tolerance, a technological alarm is issued.

IM174/PROFIBUS drives

In addition to the option of operating stepper drives on the onboard inputs of the C2xx, the IM174 module is available for use in all platforms as an interface for stepper drives. From the SIMOTION perspective, stepper drives linked via IM174 behave like digital drive links.

Alternatively, stepper drives can be linked with a PROFIBUS interface if they support the PROFIdrive profile. See Coupling of digital drives (Page 56).

You can find more information in the *Distributed I/O IM 174 PROFIBUS Module Manual*.

See also

Position control (Page 111)

Setting as a real axis with stepper drive C2xx (V3.2 and higher) (Page 58)

Stepper drives (Page 143)

4.5 Using the expert list for an axis

For the standard SIMOTION application, the required parameters (configuration data and system variables) are queried in the axis configuration wizard or are specified automatically. For the Axis technology object (TO), you can specify additional selected parameters via the axis parameter assignment dialog boxes (under Axis Object in the project navigator).

The expert list provides read/write access to all configuration data and system variables of the Axis technology object. The list includes data which cannot be set in the axis wizard or in axis parameter assignment dialog boxes.

With V4.1 SP1 and higher, the **Selected Parameters** tab in the expert list provides a default view of the most important configuration data and system variables.

This tab also shows the most important configuration data and system variables for programming and diagnostics of the virtual, drive, and position axis types, the following object of a following axis, and the external encoder.

Note

In individual SIMOTION applications, it may be necessary to change automatically specified parameters. These configuration data and system variables can only be displayed and changed in the expert list.

To open the expert list for the Axis technology object:

1. In the project navigator, double-click TO Axis. The configuration window appears in the working area and the **Axis** menu is displayed.
2. Double-click the **Expert list** entry under the Axis TO in the project navigator. The expert list is displayed in the working area.

Additional information on working with the expert list can be found in the **Motion Control Basic Functions** Function Manual.

4.6 Automatic controller setting

4.6.1 Overview of automatic controller setting (as of V4.1 SP1)

In the **Automatic Controller Setting** screen form, you can configure an automatic setting of the speed controller and the DSC position controller for SINAMICS drive units. The necessary steps for this calculation can be controlled from this screen form. The calculated parameter values for the speed controller or position controller are displayed and can then be transferred online to the drive or axis on the control.

Open the Automatic Controller Setting screen form

The following options are available for opening this screen form:

- Via the *Target System – Automatic Controller Setting* menu command from the main menu
- From the project navigator (under *Drive – Commissioning*)
- Via the main toolbar (in the Trace/Measuring Functions group)
- From the *Static Controller Data* dialog box for the axis via a button next to the Kv factor

Requirements

The conditions for the automatic speed controller setting and position controller setting are listed below. You can find additional requirements for the automatic position controller setting in the section about the automatic position controller setting (Page 235).

- Drive is a SINAMICS drive
- Drive is operated in the "SERVO" operating type
- The control is performed with the motor encoder
- An online connection to the associated drive device exists

Note

The software limit switches for the axes do not function during automatic controller setting.

Note

In certain cases, automatic controller setting may not be able to determine the optimum controller settings for servo. This applies to the positioning of band-stops, for example.

Supported devices

An automatic controller setting is possible with the following devices: SINAMICS Integrated, CX32, CX32-2, S120 - CU320, S120 - CU320-2, S120 - CU310 and S120 - CU310-2.

Procedure

The sequence below must be followed for the automatic controller setting:

1. Set the speed controller
   (see also Automatic speed controller setting (as of V4.1 SP1) (Page 231))
2. Set the DSC position controller
   (see also Automatic position controller setting (as of V4.1 SP1) (Page 235))

Note

You can cancel automatic controller setting by pressing the SPACEBAR.

- The step currently being executed is canceled.
- The drive enable is inhibited.

See also

Overview of commissioning the position controller of positioning axes (Page 147)

Assigning automatic controller optimization (Page 350)

4.6.2 Automatic speed controller setting (as of V4.1 SP1)

In the **Automatic Controller Setting** screen form, you can select the SINAMICS drive unit and the drive for which you want to carry out an automatic speed controller setting.

The automatic setting is divided into individual steps in which the measuring functions on the drive can be initiated, among other things. Drive parameters are changed online to perform these steps. After an individual step has been executed or canceled, the parameter set is restored with its status before the step was performed.

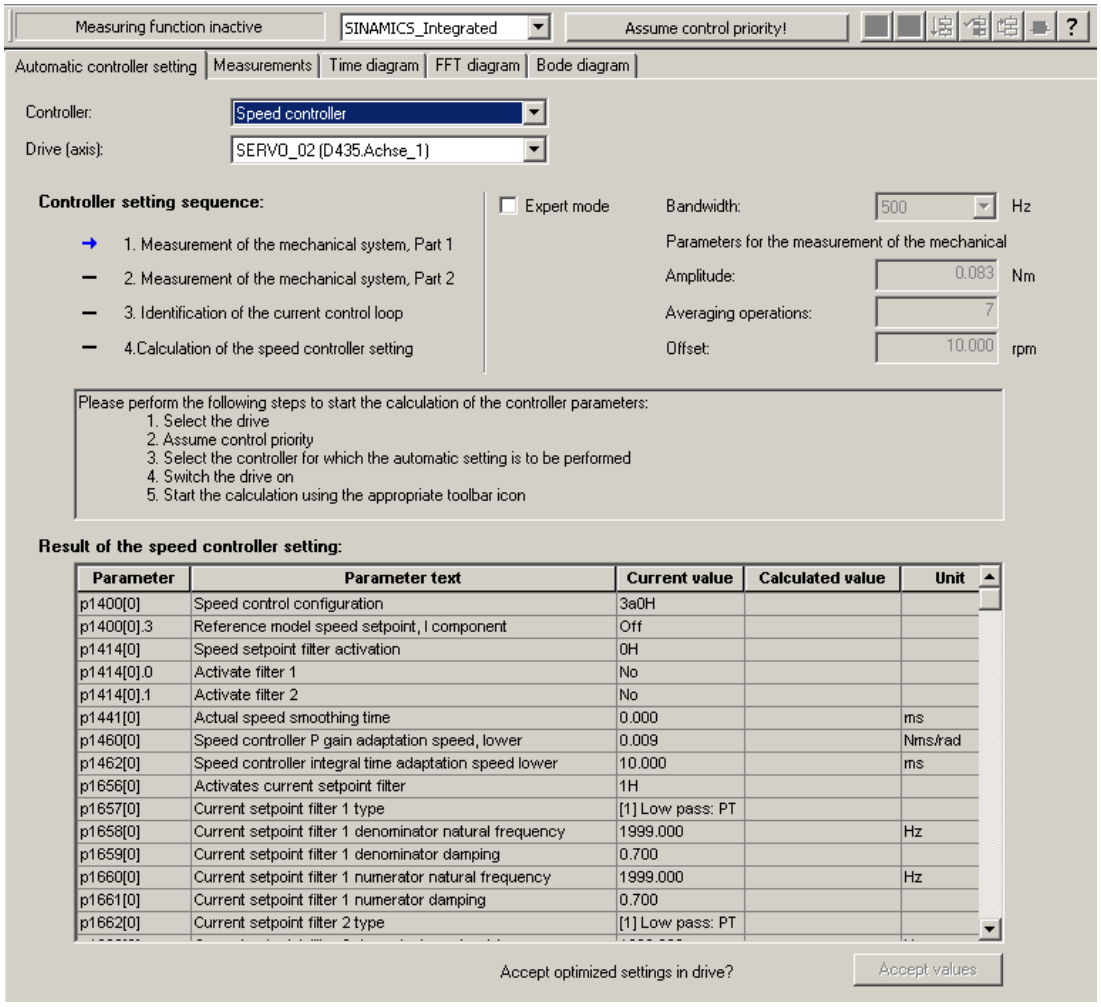


Figure 4-4 Automatic controller setting - Speed controller example

Features

The automatic speed controller setting has the following characteristics:

- Damping of resonances in the speed control section
- Setting of filters in the current setpoint branch
- Automatic setting of the Kp gain factor and the Tn reset time of the speed controller
- The speed setpoint filter and the reference model are not changed

The requirements for the speed controller setting are described in Overview of the automatic controller settings (Page 230).

Save the current settings

Online drive parameters are changed during a step. It is recommended that the current settings are saved before setting the controller. If the online connection is canceled during the execution of a step, you can restore these saved settings.

Proceed as follows to make a backup:

1. In the project navigator, select the SINAMICS unit with the drive for which you want to perform the automatic setting
2. Load it to the programming device (*Target System - Load - Load to PG...*)

To restore the data, perform a download.

Procedure

Perform the following steps for the automatic setting of the speed controller:

1. Call the **Automatic Controller Setting** screen form (the automatic speed controller setting is already preset in the Controller field)
2. Select the drive unit and drive
3. Assume control priority
4. Enable the drive
5. Perform these four steps in automatic mode or in individual steps
6. Consider the calculation results for the relevant parameters
7. Transfer the calculated speed controller parameter values to the drive
8. Disable the drive
9. Return control priority
10. Save the online parameters

Note

Emergency cancelation of automatic setting with <space key>

The following actions are performed:

- The step currently being executed is canceled
- The drive is disabled

Transferring the calculated parameter values to the online parameters

You initiate the transfer of the calculated parameter values to the corresponding online parameters of the drive with the "Accept" button.

Backing up the automatically set parameters

Proceed as follows to save the parameters:

1. In the project navigator, select the SINAMICS unit with the drive for which you want to perform the automatic setting
2. Copy RAM to ROM (*Target System - Copy RAM to ROM*)
3. Load it to the programming device (*Target System - Load - Load to PG...*)

Note

If required, the automatic controller settings can be checked using the measuring functions.

See also

SIMOTION measuring functions (Page 238)

Assigning automatic controller optimization (Page 350)

4.6.3 Automatic position controller setting (as of V4.1 SP1)

In the **Automatic Controller Setting** screen form, you can select the SINAMICS drive unit and the drive for which you want to carry out an automatic DSC position controller setting. The necessary steps for this calculation can be performed from this screen form. The calculated Kv value is displayed and can then be accepted online in the configuration data of the axis that is assigned to the drive.

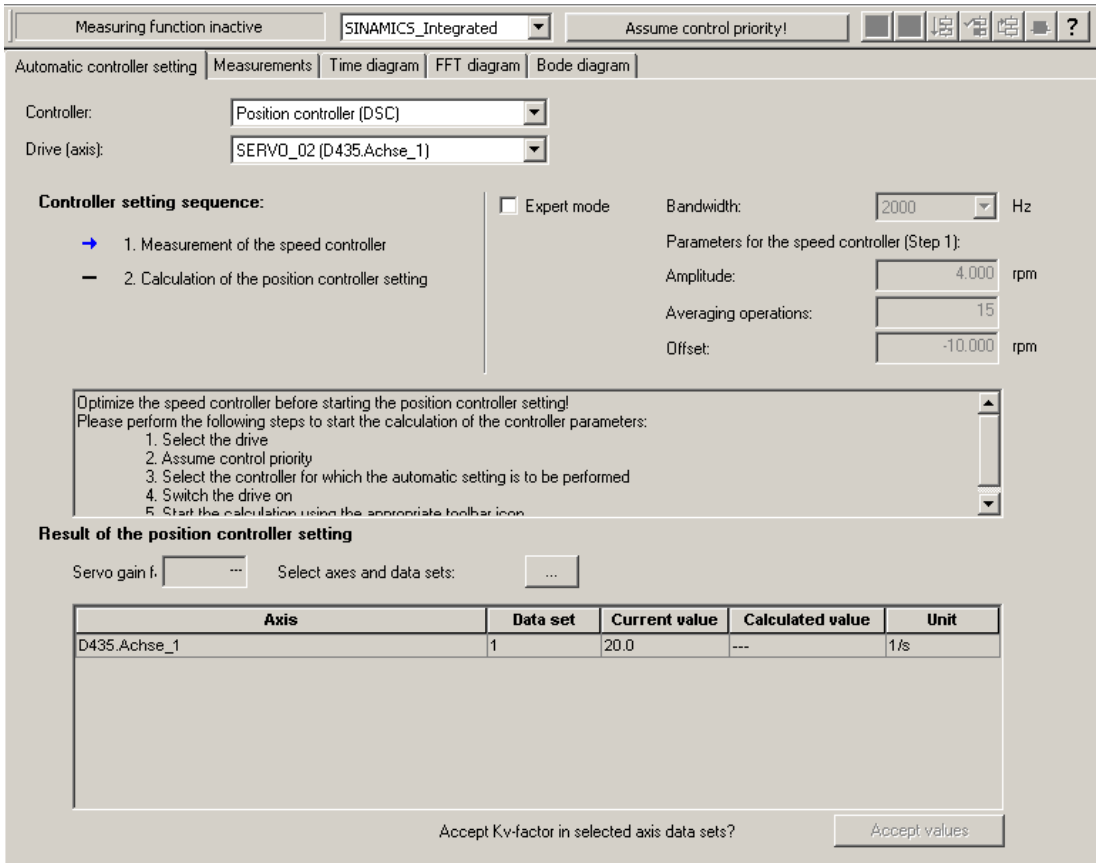


Figure 4-5 Automatic controller setting - Position controller example

Requirements and boundary conditions

In addition to the General requirements (Page 230) for the automatic controller setting, the following requirements/boundary conditions apply:

- DSC or DSC with spline is required for the position controller setting.
  - You use a telegram that supports DSC (telegram 5, 6, 105, 106, 125 or 126)
  - If you have selected a telegram which does not support DSC, you must select one of the telegrams mentioned above instead.
    As various presettings are required for DSC that can only be set when running through the axis wizard the first time, you have to make these settings manually:
    - Static controller data: Precontrol with weighting factor 100%
      (**NumberOfDataSets.DataSet_1.ControllerStruct.PV_Controller.preCon** = YES and **NumberOfDataSets.DataSet_1.ControllerStruct.PV_Controller.kpc** = 100%)
    - Static controller data: Symmetry filter (extended symmetry filter active)
      (**NumberOfDataSets.DataSet_1.ControllerStruct.PV_Controller.balanceFilterMode** = MODE_2)
    - Static controller data: Fine interpolator (acceleration continuous interpolation)
      (**FineInterpolator._type** = CUBIC_MODE)
    - Dynamic controller data: Speed control loop 0.0 equivalent time and position control loop 0.0 equivalent time
      (**NumberOfDataSets.DataSet_1.DynamicData.positionTimeConstant** = 0.0 and **NumberOfDataSets.DataSet_1.DynamicData.velocityTimeConstant** = 0.0)
- The speed controller has been set previously (e.g. with the automatic speed controller setting).
- At least one axis is connected to the SINAMICS drive (servo).
- The actual values and the manipulated variables between the controller and the drive have been correctly scaled by the user. No validation is performed.
  Match the configurations of the SIMOTION controller with the drive, for example, using the "Transfer data from the drive" button in the axis wizard.
  The following parameters in the drive are used for SINAMICS:
  P2000: Normalization speed
  P1082: Maximum speed
  See also Coupling of digital drives (Page 56).
- An online connection to the SIMOTION device must exist for the result of the automatic position controller to be transferred.
- The balancing filter is not changed.
- For operation without precontrol, the equivalent time constant of the position controller must be adjusted manually by the user (PostionTimeConstant = 1/Kv).
- Vibration on the load side is not taken into account for the position controller setting.

Procedure

Perform the following steps for the automatic setting of the position controller:

1. Open the **Automatic Controller Setting** screen form
2. Select the drive unit and drive (axis)
3. Specify the "Position controller (DSC)" controller preselection
4. Assume control priority

5. Enable the drive
6. Perform these two steps in automatic mode or in individual steps
7. Select the axis data sets to which the Kv factor are to be transferred
8. Transfer the calculated Kv factor to the axis data sets you have selected
9. Disable the drive
10. Return control priority
11. Back up the online parameters

Note

Emergency cancelation of the automatic setting with <spacebar>

The following actions are performed:

- The step currently being executed is canceled
- The drive is disabled

Transferring the calculated parameter values to the online parameters

You initiate the transfer of the calculated Kv factor to the axis data set of the target device with the "Accept" button.

Backing up the automatically set parameters

Proceed as follows to back up the parameters:

1. In the project navigator, select the SIMOTION unit with the axis for which you want to perform the automatic setting
2. Copy current data to RAM function (*Target System - Copy current data to RAM*)
3. Copy RAM to ROM (*Target System - Copy RAM to ROM*)
4. Load the configuration data to the programming device (*Target System - Load - Load Configuration Data to PG*)

Note

If required, the automatic controller settings can be checked using the measuring functions.

See also

SIMOTION measuring functions (Page 238)

Assigning automatic controller optimization (Page 350)

4.7 SIMOTION measuring functions

The SIMOTION measuring functions are used to commission the axis controller without requiring a user program. In SIMOTION SCOUT, the user can choose a predefined measuring function from a selection of functions. Based on this selection, parameters are then assigned in the target device for the SIMOTION Trace, the function generator, and the required axis enables and motion functions. The user can then launch these measuring functions via SIMOTION SCOUT and evaluate the resulting measurements in the SIMOTION Trace diagrams.

DSC with spline is deactivated when the measuring functions are executed.

The user can configure a user-defined measuring function in expert mode.

The following measuring functions are available:

- **Final controlling element jump (as of V4.0)**
- **Final controlling element frequency response (as of V4.0)**
- **Position control ramp (as of V4.0)**
- **Position control reference frequency response (as of V4.0)**
- **Expert mode (as of V4.0)**

In addition, the **circularity test** is also available for synchronous operation.

Table 4-1 Available measuring functions based on the axis type

	0	1	2	3	4	5	6	7
Positioning/following axis								
Final controlling element jump	X	-	X	X	X	X	-	-
Final controlling element frequency response	X	-	X	X	X	X	-	-
Position control ramp	X	-	X	X	X	X	-	-
Position control reference frequency response	X	-	X	X	X	X	-	-
Expert mode	X	-	X	X	X	X	-	X
Circularity test	X	-	X	X	X	X	-	X
Speed-controlled axis								
Final controlling element jump	X	-	X	X	X	X	-	-
Final controlling element frequency response	X	-	X	X	X	X	-	-
Expert mode	X	-	X	X	X	X	-	-

0 Real electric axis (REAL_AXIS)

1 Virtual axis (VIRTUAL_AXIS)

2 Real electric axis with force/pressure control (REAL_AXIS_WITH_FORCE_CONTROL)

3 Real hydraulic axis (Q valve) (REAL_QFAXIS)

4 Real hydraulic axis with force/pressure specification (Q-valve and P-valve) (REAL_QFAX-IS_WITH_OPEN_LOOP_FORCE_CONTROL)

5 Hydraulic axis with pressure/force control (Q valve) (REAL_QFAX-IS_WITH_CLOSED_LOOP_FORCE_CONTROL)

6 Real hydraulic axis with force/pressure specification (P-valve) (REAL_QFAX-IS_WITH_OPEN_LOOP_FORCE_CONTROL_ONLY)

7 Real electric axis with master value output via encoder signal simulation (REAL_AX-IS_WITH_SIGNAL_OUTPUT)

Note

Requirements

- An online connection to the SIMOTION device must be established.
- The target device must contain the current axis configuration. If necessary, download the project or upload the configuration data for alignment purposes (Target system > Load > Load configuration data to PG).
- It must be permissible to change the operating mode to STOP_U in the SIMOTION device. An automatic switchover to the STOP_U operating mode will be performed.
- No alarms may be pending on the axis. If necessary, acknowledge the pending alarms in the alarm window and restart the measuring function.

Final controlling element jump measuring function (as of V4.0)

This measuring function can be used for optimizing the lower-level control loop or process (e.g. speed control) in the time range.

A jump function is superimposed on the manipulated variable.

Signal form	Jump with velocity offset
Superimposition point	internalServoSettings.additionalSetpointValue[0]
Measured variables	• Superimposed manipulated variable internalServoSettings.additionalSetpointValue[0] • Actual velocity sensorData[x].velocity **1)**

1) The index is based upon the selected measuring system. The measuring system is selected by the user.

Final controlling element frequency response measuring function (as of V4.0)

This measuring function can be used for optimizing the lower-level control loop or process (e.g. speed control) in the frequency range.
This measuring function can also be used to determine the frequency response of an actuator (e.g. hydraulic process).

A PRBS signal (Pseudo-Random-Binary-Signal) generated by the signal generator is superimposed on the manipulated variable. The control loop is opened.

Signal form	PRBS signal
Superimposition point	internalServoSettings.additionalSetpointValue[0]
Measured variables	• Superimposed manipulated variable internalServoSettings.additionalSetpointValue[0] • Actual velocity sensorData[x].velocity **1)**

1) The index is based upon the selected measuring system. The measuring system is selected by the user.

Position control ramp measuring function (as of V4.0)

This measuring function can be used to optimize the position controller in the time range.

A ramp function is superimposed on the position setpoint. The position control loop is closed.

There are two variations of this measuring function:

- Position control ramp:
  The signal is injected before the dynamic response adaptation.
- Position control ramp without precontrol and setpoint filter:
  The setpoint is injected directly before the summation point of the position controller.

Signal form	Rectangle + rectangular integrator
Superimposition point	• A) Position control ramp internalServoSettings.additionalCommandValue[0] • B) Position control ramp without precontrol and setpoint filter internalServoSettings.additionalControllerCommandValue[0]
Measured variables	• Superimposed setpoint Superimposition point A) or B) • Manipulated variable actorData.totalSetpoint • Actual velocity sensorData[x].velocity **1)**

1) The index is based upon the selected measuring system. The measuring system is selected by the user.

Position control reference frequency response measuring function (as of V4.0)

This measuring function can be used to optimize the position controller in the frequency range.

A PRBS signal (pseudo random binary signal) is superimposed on the position setpoint. The position control loop is closed.

There are two variations of this measuring function:

- Position control reference frequency response:
  The signal is injected before the dynamic response adaptation.
- Position control reference frequency response without precontrol and filter:
  The setpoint is injected directly before the summation point of the position controller.

Function generator 1 for reference frequency response (ramp)		
	Signal form	Ramp (continuous in 1st derivation)
	Superimposition point	• A) Position control reference frequency response internalServoSettings.additionalCommandValue[0] • B) Position control reference frequency response and precontrol and setpoint filter internalServoSettings.additionalControllerCommandValue[0]
Function generator 2 for reference frequency response		
	Signal form	PRBS signal
	Superimposition point	• C) Position control reference frequency response internalServoSettings.additionalCommandValue[1] • D) Position control reference frequency response and precontrol and setpoint filter internalServoSettings.additionalControllerCommandValue[1]
Measured variables		• Superimposed position setpoint 1, ramp Application of function generator 1 A) or B) • Superimposed set position 2, PRBS Application of function generator 2 C) or D) • Actual position sensorData[x].position 1) • Modulo overflow counter, actual value sensorData[x].moduloCycles 1)

[1)] The index is based upon the selected measuring system. The measuring system is selected by the user.

Expert mode measuring function (as of V4.0)

Various settings can be changed individually in expert mode. Thus, the signal type and superimposition point can be selected explicitly.

Possible signal types:

- PRBS (pseudo random binary signal)
- Jump
- Triangular
- Sinusoidal

The **internalServoSettings.~** structure contains internal system variables for setpoint and manipulated variable override of the measuring functions.

Table 4-2 Possible superimposition points (internalServoSettings.~ structure)

Identifier	Variable	Comment/function
Setpoint	additionalCommandValue	Setpoint superimposition before dynamic response adaptation Same function as servoSettings.additionalCommandValue
Manipulated variable	additionalSetpointValue	Manipulated variable superimposition Same function as servoSettings.additionalSetpointValue
Setpoint on the controller	additionalControllerCommandValue **1)**	Setpoint superimposition on the controller Setpoint superimposition before the summation point of the position/speed controller

1) For measurements that are performed without precontrol or setpoint filter (dynamic response adaptation, balancing filter), the setpoint excitation must occur directly before the summation point of the position or speed controller.

Note

The system variables for superimposition are intended for exclusive use by the SIMOTION measuring functions. These variables cannot be modified by the user.

Circularity test - evaluation of the axis dynamics for synchronous operation with angular synchronism

Minor deviations of amplitude and phase can be seen clearly in a circularity diagram. Another advantage of the circularity test is that it enables exclusive observation of actual values. In contrast, a disadvantage of a following error observation is that the following error (followingError system variable) is corrected, for example, with DSC, and is thus subject to error in a sense.

Function generator 1 of the circularity test, Axis 1		
	Signal form	Sinusoidal
	Superimposition point	• internalServoSettings.additionalCommandValue[0]
Function generator 2 of the circularity test, Axis 2		

	Signal form	Sinusoidal
	Superimposition point	• internalServoSettings.additionalCommandValue[0]
Measured variables		• Position setpoint, Axis 1 internalServoSettings.additionalCommandValue[0] • Position setpoint, Axis 2 internalServoSettings.additionalCommandValue[0] • Actual position, Axis 1 sensorData[x].position **1)** • Actual position, Axis 2 sensorData[x].position **1)** • Actual value modulo overflows, Axis 1 sensordata[x].moduloCycles **1)** • Actual value modulo overflows, Axis 2 sensordata[x].moduloCycles **1)**

1) The index is based upon the selected measuring system. The measuring system is selected by the user.

See also

Setpoint superimposition (Page 128)

Manipulated variable superimposition (Page 137)

4.8 Axis control panel

The axis control panel is used to control and monitor individual axes without requiring a user program. You can use it to traverse axes along with the drive.

Application examples

- Function test on individual axes before the axes are traversed in a coordinated manner under the control of a program
- Traversing axes before the user program is available or if only parts of the user program are available
- Traversing the axes for optimization purposes (controller setting)
- Axis homing or absolute encoder adjustment
- Enabling and disabling an axis
- Acknowledging axis alarms
- Retracting **an axis**: Traverse the axis to another position quickly and independently of the program

Requirements

- An online connection to the SIMOTION device must be established.
- The target device must contain the current axis configuration. If necessary, download the project or upload the configuration data for alignment purposes (Target System - Load - Configuration Data to PG).

⚠ **WARNING**
Danger to life through unexpected machine movement
Failure to observe the safety instructions can result in personal injury and material damage.
Note the safety instructions in the **Assume Control Priority** SCOUT dialog box.

The system variable **controlPanel** indicates whether the control panel on the Axis technology object is active.

The control panel can be used in operating state STOPU and (from V4.4) in operating state RUN. Please refer to the additional information in the SCOUT Online Help (Axis control panel index).

Support for SINAMICS Safety Integrated Functions 5

5.1 Overview - Support of SINAMICS Safety Integrated Functions on the Axis technology object

SIMOTION provides support for SINAMICS drives that can perform safety-related functions. The safety status information of the drive is provided in the system variables.
As of V4.4, SIMOTION commands for triggering safety-related functions are possible on the drive (e.g. SBT).

The purpose of this support from SIMOTION is to prevent stop responses on the drive side by ensuring for the safety-related monitoring functions that the drive does not exit the monitored operating state.

As of SIMOTION V4.4, the SIMOTION-side support for *SINAMICS Safety Integrated Functions* has been simplified and adapted to a generic concept with Drive Safety Data Block (DSDB).

If symbolic assignment is activated (standard setting for new projects), communication is automatically adjusted to the safety-related functions enabled on the drive.

Always use the standard setting (DSDB) on the Axis technology object in conjunction with symbolic assignment for new projects as of V4.4.

For projects < V4.4, the compatibility mode with Safety Information Data Block (SIDB) is available.

The following figure provides decision guidance for managing projects < V4.4 with SCOUT as of V4.4.

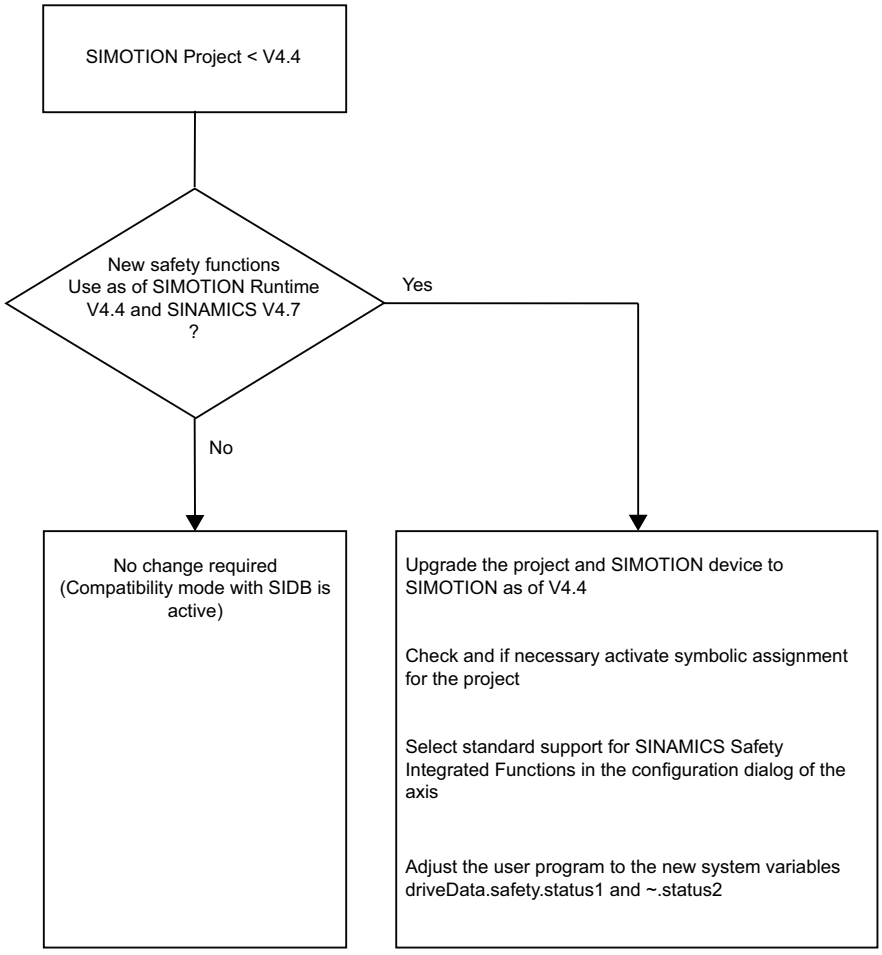


Figure 5-1 Using older SIMOTION projects with SIMOTION SCOUT as of V4.4

If the project was created with SCOUT < V4.2, please refer to Section Restrictions after switching from compatibility mode (SIDB) to standard (DSDB). (Page 278).

The status information < V4.4 (Page 259) is still provided for compatibility reasons, even with the standard setting (DSDB), with the exception of SSM.

Note

Further documentation

The safety-related functions of SINAMICS: are described comprehensively in the *SINAMICS Safety Integrated* Function Manual.

As of SIMOTION V5.3

- SLA - Safely-Limited Acceleration with fine resolution and acceleration filtering
- 2 ms monitoring cycle on a double-axis Motor Module

- SS2ESR (Safe Stop 2 Extended Stop and Retract) (The PROFIsafe telegram has been extended with the External Stop E)
- Reluctance motor can be used with Safety encoderless

As of SIMOTION V5.2

- Acceptance of the F destination address from HW Config (STEP 7) or HWCN (TIA Portal)
- Support of new safety-related functions SLA and SCA
- Support of PROFIsafe telegram 903
- SLP, SP, Safe Referencing, SCA. As of SINAMICS V5.1, Advanced Safety is required SP synchronous cannot be used with SIMOTION
- New safety licenses as of SINAMICS V5.1, see Safety licenses (Page 253)

As of SIMOTION V4.5

- Selection of response to PROFIsafe failure (STOP B)

As of SIMOTION V4.4

Simplification and implementation of the SIMOTION-side support with DSDB

Extension of the safety-relevant functionality:

- SIMOTION-side support of the DSDB with automatic selection of telegram extension
- *SINAMICS Safety Integrated Basic Functions* support on the TO axis
- Support for new safety-related functions: SP, SLP, SBT
- Support for an additional PROFIsafe telegram (902 on the TIA Portal)

As of SIMOTION V4.3

Extension of the safety-relevant functionality:

- SDI function is available in SIDB
- SKS stage 1 can be evaluated with a factor transferred via PROFIsafe, whereby almost any SLS limit values can be specified online
- Support for new SINAMICS V4.5 functions
- Support for additional PROFIsafe telegrams (31 and 901)

As of SIMOTION V4.2

Extension of the safety-relevant functionality:

- SIMOTION-side support for SIDB with automatic telegram extension
- Additional generation of the technological alarm 50023 in the event of drive-autonomous changes in status
  Drive-autonomous changes in status occur e.g. when SS1 or SS2 is selected
- Read-out of the safety message buffer via system function

See also Coupling of digital drives (Page 56).

As of SIMOTION V4.1 SP1

SIMOTION supports the *SINAMICS Safety Integrated Extended Functions* for the axis TO in the following way:

- Display of the status of safety-related functions in the drive in system variables of the technology object
- Message of a new safety event in the safety message buffer of the drive (alarm TO)
- Message of the status for SLS, SOS and SS2 (alarm TO)

See also

Technology data (Page 179)

Safety status information (part of the DSDB) (Page 254)

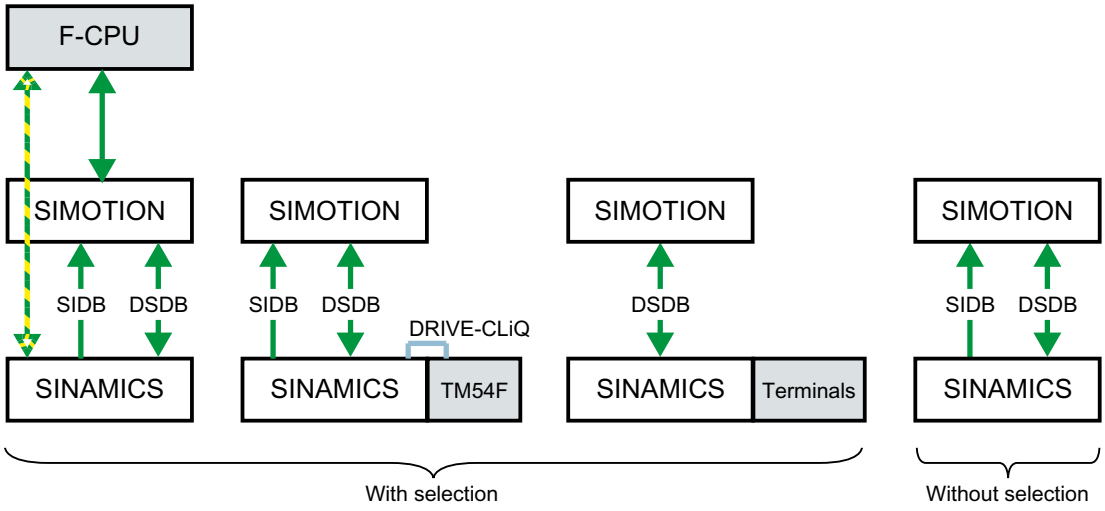
Safety channel and safety channel types (when symbolic assignment is deactivated) (Page 251)

Standard (DSDB) as of SIMOTION V4.4 (Page 253)

5.2 Communication

As of SIMOTION V4.4, the safety information is transferred between the drive and SIMOTION by default as a telegram extension via the DSDB. Both *SINAMICS Safety Integrated Basic Functions* and *SINAMICS Safety Integrated Extended Functions* are supported.

Prior to SIMOTION < V4.4, the safety information is transferred from the drive via the SIDB. *SINAMICS Safety Integrated Extended Functions* are supported.



Optional SIDB (compatibility mode) or DSDB (standard)
SIDB: Supports the SINAMICS Safety Integrated Extended Functions
DSDB (as of SIMOTION V4.4): Supports the SINAMICS Safety Integrated Basic & Extended Functions

Figure 5-2 Communication with DSDB or SIDB in different safety configurations

5.3 Main procedure for safety configuration with SIMOTION

To use the support for the *SINAMICS Safety Integrated Extended Functions* to the axis TO, proceed as follows:

- Start up the axis and the drive without safety functions
- Configure the safety-related functions in the drive (see *SINAMICS Safety Integrated* Function Manual)
- If the safety-related function is to be activated via PROFIsafe, configure PROFIsafe communication to a safety PLC (e.g. SIMATIC S7 CPU317F).
  See Section *PROFIsafe* in the *SIMOTION SCOUT Communication* System Manual.
- Activate the SIMOTION-side support for the safety-related functions:
  - Connect the drive symbolically in the axis configuration to the axis. To do so, click the ... button under drive assignment.
    or
  - Activate the safety-related functions in the safety dialog.
- Only if symbolic assignment is deactivated:
  Configure the DSDB or SIDB as extension in the telegram.
  As of SIMOTION V4.2, this is set up automatically by the system and interconnected in the drive if symbolic assignment is activated.

The safety status information transferred by DSDB (Page 254) or SIDB (Page 259) can be evaluated via system variables on the TO. The status is required for many safety functions to bring the axis into the permissible operating range via the user program.

The *SINAMICS Safety Integrated Functions* are displayed in the configuration dialog of the axis.

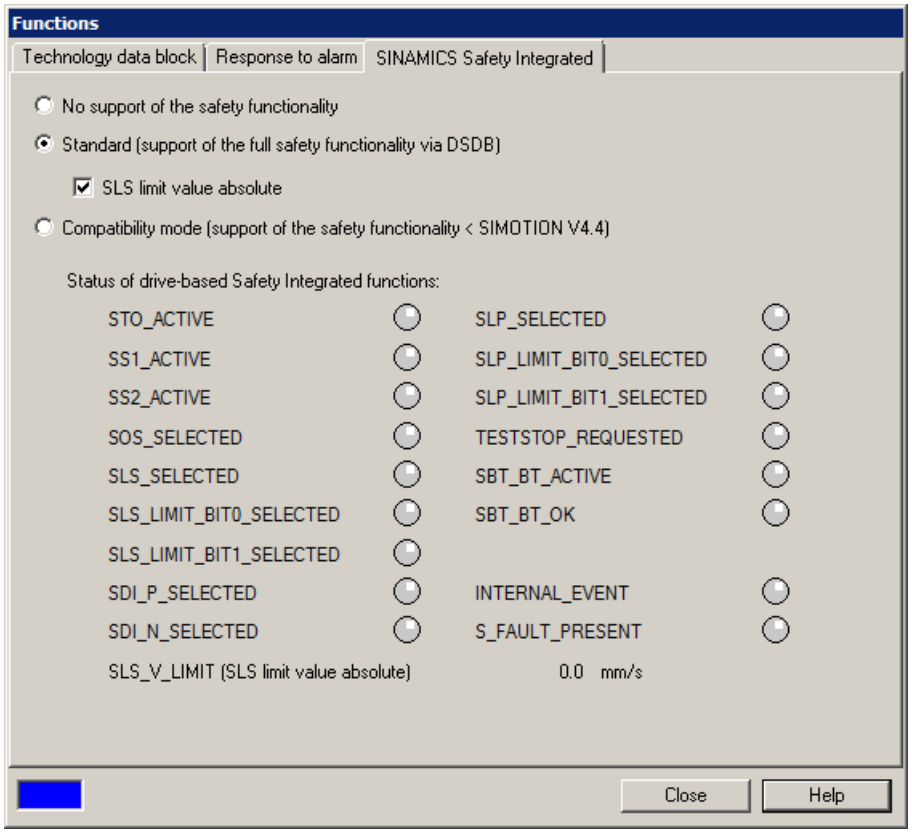


Figure 5-3 Safety dialog

Table 5-1 Support safety functions and their status information

	Safety-related function in the drive	**DSDB - standard (driveData.safety.status1 and driveData.safety.status2)**	**SIDB - compatibility mode (driveSafetyExtendedFunctionsInfoData.state)**
STO	Safe Torque Off	Status1 - bit 0 (STO_ACTIVE)	Bit 0 (STO_ACTIVE)
SS1	Safe Stop 1 (SS1)	Status1 - bit 1 (SS1_ACTIVE)	Bit 1 (SS1_ACTIVE)
SS2	Safe Stop 2 (SS2)	Status1 - bit 2 (SS2_ACTIVE)	Bit 2 (SS2_ACTIVE)
SOS	Safe operating stop	Status1 - bit 3 (SOS_SELECTED) Status2 - bit 4 (SOS_ACTIVE)	Bit 3 (SOS_SELECTED)
SLS	Safely Limited Speed	Status1 - bit 5 (SLS_SELECTED) Status1 - bit 6 (SLS_LIMIT_BIT0_SELECTED) Status1 - bit 7 (SLS_LIMIT_BIT1_SELECTED) Status2 - bit 5 (SLS_ACTIVE)	Bit 4 (SLS_SELECTED)
SSM	Safe Speed Monitor (SSM)	-	Bit 7 (SSM_ACTIVE)
SDI	Safe Direction (safe direction of motion)	Status1 - bit 8 (SDI_P_SELECTED) Status1 - bit 9 (SDI_N_SELECTED)	Bit 12 (SDI_P_SELECTED) Bit 13 (SDI_N_SELECTED)
SP	Safe Position SP synchronous cannot be used with SIMOTION	Status1 - bit 23 (REF_POS_LATCHED_OR_VALID) Status1 - bit 24 (REF_REQUESTED)	-

	Safety-related function in the drive	DSDB - standard (driveData.safety.status1 and driveData.safety.status2)	SIDB - compatibility mode (driveSafetyExtendedFunctionsInfoData.state)
SLP	Safely Limited Position (SLP)	Status1 - bit 11 (SLP_SELECTED) Status1 - bit 12 (SLP_LIMIT_BIT0_SELECTED) Status1 - bit 23 (REF_POS_LATCHED_OR_VALID) Status1 - bit 24 (REF_REQUESTED)	-
SBT	Safe Brake Test (SBT)	Status2 - bit 15 (SBT_SELECTED) Status2 - bit 16 (SBT_CONTROL_PRIO_DRIVE) Status2 - bit 17 (SBT_SIGN_LOAD) Status2 - bit 18 (SBT_BT_BRAKE_2) Status2 - bit 19 (SBT_BT_ACTIVE) Status2 - bit 20 (SBT_BT_OK) Status2 - bit 21 (SBT_BT_FINISHED) Status2 - bit 22 (SBT_BT_BRAKE_CLOSE_REQ)	-
SLA	Safely-Limited Acceleration	Status1 - bit 16 (SLA_SELECTED)	-
SCA	Safety-related function in the drive: Safe Cam	Standard: Status1 - bit 23 (REF_POS_LATCHED_OR_VALID) Status1 - bit 24 (REF_REQUESTED)	-

The stop responses in the drive (e.g. STOP A) for the applicable safety-related functions are described in the *SINAMICS Safety Integrated* Function Manual.

5.4 Safety channel and safety channel types (when symbolic assignment is deactivated)

This section is only applicable if the project standard setting **Use symbolic assignment** has been deliberately deactivated.

For communication, a distinction is made between various SC types (safety channel types). The various SC types can be used on the SC channels (SIDB, DSDB_1, DSDB_2).

In compatibility mode, the SIDB is the only SC type, so there is no need to distinguish between types.

Use of SIDB and DSDB is optional. Parallel use is not possible.

Table 5-2 List of SC types in SIDB

SC types of SIDB	Supported safety-relevant functions	Input parameter	Output parameter
SIDB	STO SS1 SS2 SOS SLS SDI	r9722.0 r9722.1 r9722.2 r9722.11 r9720.4, r9733[2] r9734.12, r9734.13	-

Note

Support for DSDB when symbolic assignment is deactivated

Note that the standard setting (DSDB) must not be used for SINAMICS drives < V4.7.

Table 5-3 List of SC types in DSDB

SC types of DSDB_x	Supported safety-relevant functions	Input parameter	Output parameter
SC1	STO SS1 SS2 SOS SLS SDI SLA	r9734	-
SC2	as SC1 SLP SP[1)] SCA	r9734 r9743	p10250
SC3	as SC2 SBT	r9734 r9743 r10234	p10250 p10235
SC100		r9733[2]	-

1) SC2 only when using SP with a safe absolute position

5.5 Safety licenses

SINAMICS makes a distinction between the following function groups:

- Safety Integrated Basic Functions
- Safety Integrated Extended Functions
- Safety Integrated Advanced Functions

As of SIMOTION V5.2 and SINAMICS V5.1, you require the Safety Integrated Advanced Functions license for SLP, SP and SCA.

5.6 Standard (DSDB) as of SIMOTION V4.4

DSDB is the standard setting for safety as of SIMOTION V4.4. The setting provide the complete safety functionality set up on the drive at the TO. Configuration is automatic and uses the DSDB channel (Drive Safety Data Blocks).

Note

The Drive Safety Data Block is an extension of the PROFIdrive axis telegrams. Transmission is according to PROFIdrive and is therefore not a secure safety telegram.

5.6.1 Display of SINAMICS Safety Integrated Functions in SIMOTION

If support for *SINAMICS Safety Integrated Functions is activated* (**technologicalData.numberOfDriveSafetyDataBlock** > 0), the following information is displayed via the system variable:

- **driveData.safety.status1**
  Contains safety status information to which the user program generally must respond immediately.
- **driveData.safety.status2**
  Contains safety status information to which no time-critical response is generally needed.
- **driveData.safety.slsSpeedLimitSelected**
  Effective absolute set velocity limitation in the drive from the DSDB.
  slsSpeedLimitSelected contains the maximum set velocity at which the Axis technology object may run.

  Note

  The value in **slsSpeedLimitSelected** is already weighted with drive parameter p9533. If p9533 < 100%, it does not represent the maximum possible actual velocity monitored by the drive (p9531[x]).

If **technologicalData.numberOfDriveSafetyDataBlock** (standard with DSDB) = 0 and **technologicalData.driveSafetyExtendedFunctionsEnabled** (compatibility mode with SIDB) = NO, there is no support for *SINAMICS Safety Integrated Functions* on the Axis technology object.

In the system variables, zero is displayed or the contents are irrelevant.

5.6.2 Axis configuration

If *SINAMICS Safety Integrated Functions* are configured in the drive, support is activated automatically on the Axis technology object if a drive is assigned.

The following configuration data settings are made:

- **technologicalData.numberOfDriveSafetyDataBlock** > 0
- **technologicalData.SafetyDataBlock_n** = type, logAddressIn, logAddressOut

Deactivated symbolic assignments

The rest of the chapter is only relevant if the project standard setting **Use symbolic assignments** is deliberately deactivated.

To configure the *SINAMICS Safety Integrated Functions*, proceed as follows:

- Select the SC type in the safety dialog for a maximum of two data blocks (DSDB_1 and DSDB_2) and enter the logical base addresses.
- Evaluate the status of the pulse enable of the drive on the Axis technology object (Page 273).

If a safety commissioning of the *SINAMICS Safety Integrated Functions* in the drive has already been performed and the data has been loaded into the configuration (load into the PG), click the "Data transfer from the drive" button (drive assignment in the axis wizard) to make these settings.

If this is not required, *SINAMICS Safety Integrated Functions* support can be deactivated by setting the **technologicalData.numberOfDriveSafetyDataBlock**=0 configuration data element.

5.6.3 Safety status information (part of the DSDB)

The following tables show the structure of the safety status information in system variables on the TO axis.

Table 5-4 **Allocation of the system variable driveData.safety.status1**

Bit	State	Identifier	Interconnection in the drive
0	STO is active	STO_ACTIVE	r9734.0
1	SS1 is active	SS1_ACTIVE	r9734.1
2	SS2 is active	SS2_ACTIVE	r9734.2
3	SOS is selected	SOS_SELECTED	r9734.5
4	-	-	-
5	SLS is selected	SLS_SELECTED	r9734.6

Bit	State	Identifier	Interconnection in the drive
6	Bit 0 for selected SLS limit value	SLS_LIMIT_BIT0_SELECTED	r9734.9
7	Bit 1 for selected SLS limit value	SLS_LIMIT_BIT1_SELECTED	r9734.10
8	SDI positive selected	SDI_P_SELECTED	r9734.12
9	SDI negative selected	SDI_N_SELECTED	r9734.13
10	-	-	-
11	SLP selected	SLP_SELECTED	r9743.7
12	Bit 0 for selected SLP limit value	SLP_LIMIT_BIT0_SELECTED	r9743.4
13	-	-	-
14	SLP acceptance test on the drive is active	ACCEPTANCE_TEST_SLP_ACTIVE	r10234.14
15	-	-	-
16	SLA is selected (as of V5.2)	SLA_SELECTED	r9734.8
17	-	-	-
18	-	-	-
19	-	-	-
20	-	-	-
21	-	-	-
22	-	-	-
23	Pos latch command has been recognized by the drive	REF_POS_LATCHED_OR_VALID	r9743.15
24	Axis is not homed - SLP and SP function not available	REF_REQUESTED	r9743.14
25	-	-	-
26	-	-	-
27	Acceptance test on the drive is selected	ACCEPTANCE_TEST_SELECTED	r10234.15
28	Test stop sequence running on the drive	TESTSTOP_ACTIVE	r9743.12
29	Test stop timer has ended	TESTSTOP_REQUESTED	r9743.13
30	Entry present in the SINAMICS safety fault buffer	S_FAULT_PRESENT	r9734.15
31	Safety error has occurred	INTERNAL_EVENT	r9734.7

Table 5-5 **Allocation of the system variable driveData.safety.status2**

Bit	State	Identifier	Interconnection in the drive
0	-	-	-
1	-	-	-
2	-	-	-
3	-	-	-

Bit	State	Identifier	Interconnection in the drive
4	Safe standstill monitoring is active	SOS_ACTIVE	r9734.3
5	Safe speed monitoring is active	SLS_ACTIVE	r9734.4
6	-	-	-
7	-	-	-
8	Request retract function	ESR_REQUESTED	r9734.14
9	-	-	-
10	-	-	-
11	-	-	-
12	-	-	-
13	-	-	-
14	-	-	-
15	Safe brake test is selected	SBT_SELECTED	r10234.0
16	Setpoint specified during safe break test in the drive	SBT_CON-TROL_PRIO_DRIVE	r10234.1
17	Sign of the load torque is positive	SBT_SIGN_LOAD	r10234.7
18	Test being performed with brake 2	SBT_BT_BRAKE_2	r10234.2
19	Brake test sequence is active	SBT_BT_ACTIVE	r10234.3
20	Test of brake was OK	SBT_BT_OK	r10234.4
21	Brake test was completed	SBT_BT_FINISHED	r10234.5
22	Request close external brake	SBT_BT_BRAKE_CLOSE_REQ	r10234.6
23	-	-	-
24	-	-	-
25	-	-	-
26	-	-	-
27	-	-	-
28	-	-	-
29	-	-	-
30	-	-	-
31	-	-	-

5.7 Compatibility mode (SIDB) as of SIMOTION V4.1

The compatibility mode is available for the following projects:

- Projects < SIMOTION V4.4 with Safety Information Data Block (SIDB)
- Projects that have been converted upwards from < V4.4

This mode can only be activated if the *SINAMICS Safety Integrated Extended Functions* settings have been set on the drive.

5.7.1 Safety Information Data Block (SIDB)

The Safety Information Data Block (SIDB) shows the state of the activated *SINAMICS Safety Integrated Extended Functions* in SIMOTION.

Note

The Safety Information Data Block (SIDB) is an extension of the PROFIdrive axis telegrams. Transmission is according to PROFIdrive and is therefore not a secure safety telegram.

Table 5-6 Structure of the Safety Information Data Block (SIDB) in the actual value telegram

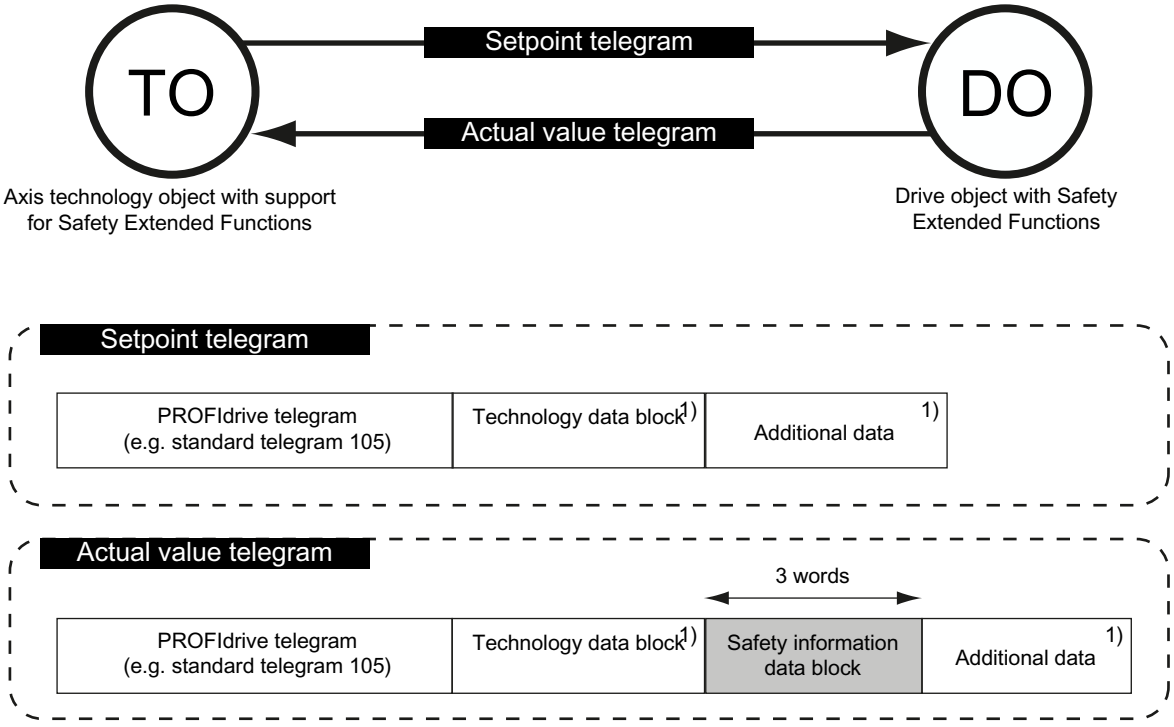
Word	Meaning
1	Safety status word (Page 259)
2	Effective absolute set velocity limitation in the drive
3	**Note:** This double word must be interconnected with r9733[0] (up to SINAMICS V4.4 only) or with r9733[2] (as of SINAMICS V4.4). For the *Interconnection of the Safety Information Data Block (SIDB) to the actual value telegram in the drive*, see figure below.

As of SIMOTION V4.2, the Safety Information Data Block (SIDB) is set up by the system and automatically interconnected in the drive.

Telegram extension (V4.1 only or with symbolic assignment deactivated)

The rest of this chapter is only relevant if the project default setting **Use symbolic assignment** is deliberately deactivated as of V4.2, or if you are working with a project which uses SIMOTION V4.1.

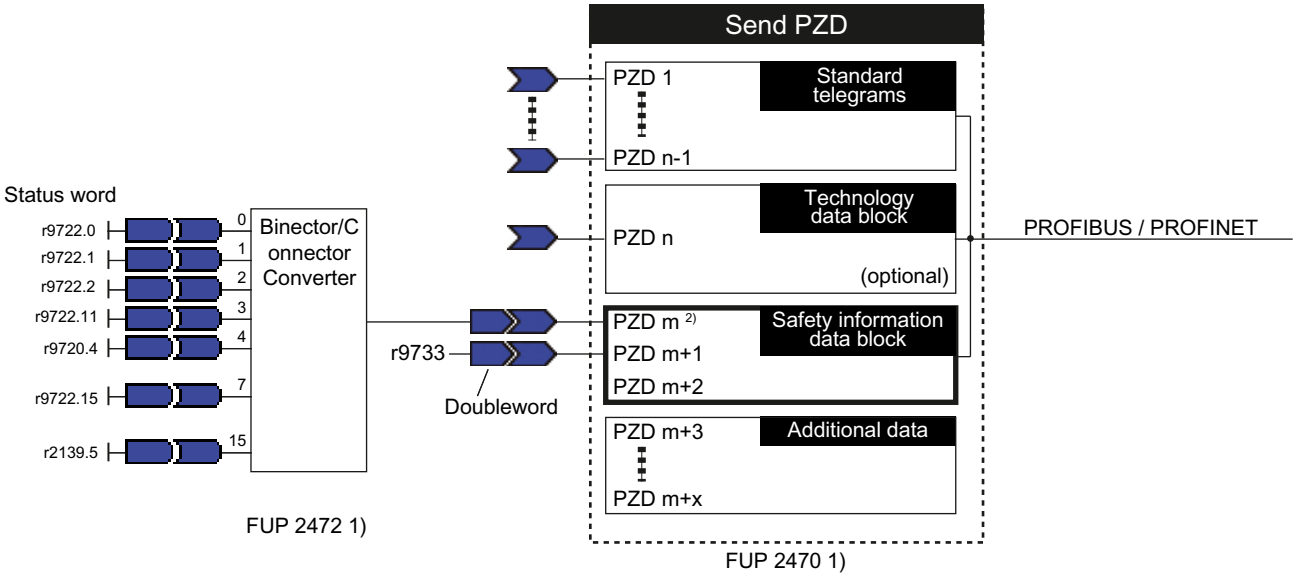
The SIMOTION support for **SINAMICS Safety Integrated Extended Functions** requires the actual value telegram to be extended by the Safety Information Data Block (SIDB) (three words).



1) Optional

Figure 5-4 Layout of a drive telegram with Safety Information Data Block

In order for the *SINAMICS Safety Integrated Extended Functions* to be displayed in SIMOTION, the corresponding SINAMICS parameters must be interconnected with the Safety Information Data Block (SIDB) using BICO.



1) Function charts can be accessed in online help via the “PDF documents” button
2) If technology data block is available: m = n + 1
If no technology data block is available: m = n

Figure 5-5 Interconnection of the Safety Information Data Block (SIDB) to the actual value telegram in the drive

The configuration of the telegram extension is configured using the SINAMICS device > Communication > Telegram configuration dialog with an extension of the input data of the axis telegram by three words.
Via *Configure telegram*, the interconnection of the user-defined PZDs can be carried out using a binector-connector converter.

The logical base address for the Safety Information Data Block (SIDB) **technologicalData.driveSafetyExtendedFunctionsInfoDataIn.logAddress** results as follows:
standard telegram base address + (m-1) * 2
(m is the number of the PZD where the Safety Information Data Block (SIDB) begins). (See also figure above).

Also refer to the information for creating the technology data block in Chapter Coupling of digital drives (Page 56).

See also

Display of SINAMICS Safety Integrated Functions in SIMOTION (Page 261)

5.7.2 Safety status information (part of the SIDB)

The following table shows the structure of the safety status word:

If standard SINAMICS Safety Integrated functionality with DSDB is activated, the displays in the safety status word of the SIDB are also updated for compatibility reasons. The content of the

driveSafetyExtendedFunctionsInfoData.state system variable then also displays the information for the *Safety Integrated Basic Function* in this case.

Table 5-7 Structure of the safety status word

Bit	State	Identifier		Interconnection in the drive
0	STO is active	STO_ACTIVE	1-active	r9722.0
1	SS1 is active	SS1_ACTIVE	1-active	r9722.1
2	SS2 is active	SS2_ACTIVE	1-active	r9722.2
3	SOS is selected	SOS_SELECTED	1-active	r9722.11
4	SLS is selected	SLS_SELECTED	0-active	r9720.4
5	Reserved	-	-	-
6	Reserved	-	-	-
7	SSM (n below limit) (See also Restrictions after switching from compatibility mode (SIDB) to standard (DSDB). (Page 278))	SSM_ACTIVE	1-active	r9722.15
8	Reserved	-	-	-
9	Reserved	-	-	-
10	Reserved	-	-	-
11	Reserved	-	-	-
12	SDI positive selected (as of SINAMICS V4.4)	SDI_P_SELECTED	1-active	r9734.12
13	SDI negative selected (as of SINAMICS V4.4)	SDI_N_SELECTED	1-active	r9734.13
14	Reserved	-	-	-
15	Entry present in the SINAMICS safety fault buffer (See SINAMICS Safety Fault Buffer (Page 276))	S_FAULT_PRESENT	1-active	r2139.5

5.7.3 Display of SINAMICS Safety Integrated Functions in SIMOTION

If support for *SINAMICS Safety Integrated Extended Functions* is activated (**technologicalData.driveSafetyExtendedFunctionsEnabled**=YES), the following information is displayed via the system variable:

- **driveData.driveSafetyExtendedFunctionsInfoData.state**
  Displays the safety status word (Page 259).
  The status word can be used to evaluate the status of the safety functions in the drive in the controller. The status is required for many safety functions, in order to bring the axis from the user program to the permissible operating range. See also Overview (Page 262).
- **driveData.driveSafetyExtendedFunctionsInfoData.safeSpeedLimit**
  Effective absolute set velocity limitation in the drive from the Safety Information Data Block (SIDB), see also the table for the Structure of the Safety Information Data Block (SIDB) (Page 257).
  safeSpeedLimit contains the maximum set velocity at which the Axis technology object may run.

  Note

  The value in **safeSpeedLimit** is already weighted with drive parameter p9533. If p9533 < 100%, it does not represent the maximum possible actual velocity monitored by the drive (p9531[x]).

If **technologicalData.numberOfDriveSafetyDataBlock** = 0 (standard with DSDB) and **technologicalData.driveSafetyExtendedFunctionsEnabled** = NO (compatibility mode with SIDB), there is no support for *SINAMICS Safety Integrated Functions* on the Axis technology object.

In the system variables, zero is displayed or the contents are irrelevant.

5.7.4 Axis configuration

As of SIMOTION V4.2, if *SINAMICS Safety Integrated Extended Functions* are configured, support is activated automatically in SIMOTION.

The configuration data element **technologicalData.driveSafetyExtendedFunctionsEnabled** = YES is also set**.**

Symbolic assignments deactivated or SIMOTION < V4.2

The rest of this chapter is only relevant if the project default setting **Use symbolic assignment** is deliberately deactivated as of V4.2, or if you are working with a project which uses SIMOTION V4.1.

To configure the *SINAMICS Safety Integrated Extended Functions*, proceed as follows:

- Set the **technologicalData.driveSafetyExtendedFunctionsEnabled** configuration data element to YES. This activates the Extended Functions support.
- Enter the logical base address for the Safety Information Data Block (SIDB) in **technologicalData.driveSafetyExtendedFunctionsInfoDataIn.logAddress**.
- Evaluate the status of the pulse enable of the drive on the axis technology object (Page 273).

If a safety commissioning of the *SINAMICS Safety Integrated Extended Functions* in the drive has already been performed and the data has been loaded into the configuration (load to PG), click the **Data transfer from the drive** button (drive assignment in the axis wizard) to make these settings.

If this is not required, *SINAMICS Safety Integrated Extended Functions* support can be deactivated by setting the **technologicalData.driveSafetyExtendedFunctionsEnabled** configuration data item to NO.

5.8 Behavior and user responses

5.8.1 Overview

The safety functions in the drive are selected and deselected either from a Safety PLC (e.g. SIMATIC S7 CPU317F) using secure PROFIsafe communication, from a safety-related SINAMICS terminal module TM54F or using the fail-safe terminals on the CU (D410-2, CU310-2 or CU305). The safety functions can be selected and deselected independently of the Motion Control user program.

Many safety functions (e.g. SOS, SLS) contain monitoring functions and must be supported by the SIMOTION user program so that the drive can be set to or kept in the monitored operating state. If the monitored limits are exceeded, this causes a safety-related stop response in the drive.

The user can evaluate the safety status word via system variables. See also the status information for DSDB (Page 254) and SIDB (Page 259).

General recommendation for the user program when using SINAMICS Safety Integrated Functions as of SIMOTION V4.2

The system responds automatically to the selection of SS1 or SS2 by switching the axis to follow-up mode. The following requirements must be satisfied to do this:

- Default settings for symbolic assignment
- Automatic telegram selection
- Evaluation of the pulses enabled bits must not be changed

When STO is selected, the drive shuts down automatically.
Therefore, safety functions STO, SS1, and SS2 do not require any provision to be made in the user program in terms of a response.

For other functions such as SOS or SLS, it must be ensured via the user program that the monitored operating state is not violated.

See also

Overview - Support of SINAMICS Safety Integrated Functions on the Axis technology object (Page 245)

Display of SINAMICS Safety Integrated Functions in SIMOTION (Page 261)

5.8.2 Overview of safety functions

How the drive reacts and what user responses we recommend in SIMOTION are described below.

Note

Design your user program in accordance with the priority of the safety drive functions. Take into account possible consecutive responses of the drive, e.g. STOP_B in the event of SLS limit value violation.

Evaluate the status bits cyclically in your user program.

Also take into account user responses for connected axes.

Table 5-8 Responses of the drive and responses in the user program when a safety function is selected

Function	Drive response	Recommended response in the user program
STO	Drive switches off immediately	No response possible
SS1E	Drive switches off after expiration of a delay time	Bring axis to a standstill before expiration of the delay time and switch off the drive
SS1	Drive decelerates on the OFF3 ramp and then switches off	Set axis to follow-up mode (automatic as of SIMOTION V4.2)
SS2	Drive decelerates on the OFF3 ramp and then monitors the operating stop	Set axis to follow-up mode (automatic as of SIMOTION V4.2)
SOS	Drive monitors the operating stop after expiration of the delay time	Bring axis to a standstill and remain at a standstill
SLS	Drive monitors the maximum permissible velocity after expiration of a delay time	Limit the axis velocity accordingly
SDI	Drive monitors the direction of motion after expiration of a delay time	Maintain or stop the direction of motion of the axis accordingly
SP	Drive transfers actual position to the F-CPU	-
SLP	Drive monitors compliance with a defined position range after expiration of a delay time	Maintain position range of the axis accordingly
SLA	The drive monitors the maximum permissible acceleration after expiration of a delay time	Limit the axis accelerations accordingly
SCA	The drive transfers safe cam information to the F-CPU	-

5.8.3 Safe Torque Off (STO)As of SIMOTION V4.1

When the **STO** is selected, pulse suppression will be activated directly in the drive: A moving motor coasts to standstill. If there is a motor holding brake and activation via sequential control system, the brake closes.

If the drive enables have been canceled by **STO**, they can be reset by the SIMOTION user program with the **_enableAxis()** command. An enable, however, is possible only when **STO** has been deselected again (safe power-on disable).

In compatibility mode with SIDB, **STO** is only supported in conjunction with the *SINAMICS Safety Integrated Extended Functions* on the axis.

See also

Safety status information (part of the DSDB) (Page 254)

Safety status information (part of the SIDB) (Page 259)

5.8.4 Safe Stop 1 (SS1)As of SIMOTION V4.1

SS1 with OFF3

When **SS1** is selected, the drive independently decelerates on the OFF3 ramp in a speed-controlled manner (n=0) and enters pulse suppression after a configured delay time.

The drive is shut down when pulse suppression is achieved. This is indicated on the Axis technology object by alarm 20005 reason 4.

NOTICE
Abort of the OFF3 ramp For certain settings, the OFF3 ramp cannot be aborted on the control side! See SIMOTION system behavior with a drive-autonomous OFF3 ramp (Page 54).

User response as of SIMOTION V4.2:

The system responds automatically to SS1 selection if the configuration data item **TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation** is set to YES (set accordingly when creating an axis as of SIMOTION V4.2).
At the start of the OFF3 ramp, TO alarm 50023 (drive makes transition to autonomous state, default local response = OPEN_POSITION_CONTROL) is signaled, switching the axis to follow-up mode automatically.
Consequently, both the position controller enable and following error monitoring are inactive during the OFF3 ramp, and no more motion commands are effective. The drive enables can be canceled with the **_disableaxis()** user command, for example.
Automatic follow-up mode is only activated for the axis concerned in each case. For a possible synchronous operation line-up, the corresponding responses must be converted in the application.

If the drive enables have been canceled by **SS1**, they can be reset by the SIMOTION user program with the **_enableAxis()** command. An enable, however, is possible only when **SS1** has been deselected again (safe power-on disable).

User response with SIMOTION V4.1 or TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation=COMPATIBILITY_MODE:

When SS1 is selected (status information SS1_ACTIVE=TRUE), the current axis motion must be canceled and the position setpoint switched to follow-up mode (with the **_stop()** command at velocity = 0 and movingMode=SPEED_CONTROLLED, for example). This prevents the enables for the drive from having to be canceled (as a result of alarm 50102, following error monitoring, for example).

SS1E external stop (as of SINAMICS V4.5 and SIMOTION V4.3)

If SS1E is selected, the drive switches off after a delay time (p9652 Safety Integrated Basic Functions / p9556 Safety Integrated Extended Functions) without being autonomously decelerated along the OFF3 ramp. Within the delay time, synchronous axes can be stopped synchronously via the user program.

User response to use of the DSDB (SIMOTION V4.4 and higher) or the SIDB (for Safety Integrated Extended Functions):

If SS1E is selected (status information SS1_ACTIVE = TRUE), stop the axis before the delay time elapses and switch off the drive.

User response to Safety Integrated Basic Functions in the user program in SIMOTION V4.3:

The following section is only relevant to SIMOTION V4.3 or if the DSDB is not used.

Transfer parameters p9772.2 and p9872.2 and check them cyclically, for example, in the IPO-synchronous task.

As soon as at least one of the two parameters is TRUE, the axis must be stopped before the delay time elapses and the drive switched off.

Control via terminals

If controlling via terminals, please see the *SINAMICS Safety Integrated* Functional Manual, Chapter *Simultaneity and tolerance time of the two monitoring channels*.

See also

Safety status information (part of the DSDB) (Page 254)

Safety status information (part of the SIDB) (Page 259)

5.8.5 Safe Stop 2 (SS2)As of SIMOTION V4.1

When Safe Stop 2 is selected, the drive decelerates on the speed-controlled OFF3 ramp (n=0) and then switches to the Safe Operating Stop (SOS). The drive then decouples itself from the SIMOTION setpoint interface.

NOTICE
Abort of the OFF3 ramp For certain settings, the OFF3 ramp cannot be aborted on the control side! See SIMOTION system behavior with a drive-autonomous OFF3 ramp (Page 54).

Note

If the drive is not yet ready to run and SS2 has already been selected, all drive enables can only be set once SS2 has been deselected.
This may be the case following a further ramp up of the selected SS2, for example.

The drive enables can be set using the **_enableaxis()** user command, for example.

User response as of SIMOTION V4.2:

The system responds automatically to SS2 selection if the configuration data item **TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation** is set to YES (set accordingly when creating an axis as of SIMOTION V4.2):
As a result, the user does not have to carry out cyclic evaluation of whether SS2 is selected. At the start of the OFF3 ramp, TO alarm 50023 (drive makes transition to autonomous state, default local response = OPEN_POSITION_CONTROL) is signaled, switching the axis to follow-up mode automatically.
As long as SS2 is selected, the position controller enable and the following error monitoring therefore remain inactive and no more motion commands are effective. The drive enables can be canceled with the **_disableaxis()** user command, for example.
Automatic follow-up mode is only activated for the axis concerned in each case. For a possible synchronous operation line-up, the corresponding responses must be converted in the application.

After SS2 has been deselected, the position controller enable has to be reset with the **_enableaxis()** user command, for example, so that the axis can be traversed again.

User response with SIMOTION V4.1 or TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation=COMPATIBILITY_MODE:

When SS2 is selected (status information SS2_ACTIVE=TRUE), cancel the current motion on the axis and switch the position setpoint to follow-up mode, e.g. using the **_stop()** command at velocity = 0 and movingMode=SPEED_CONTROLLED. This prevents the enables for the drive from having to be canceled (as a result of alarm 50102, following error monitoring, for example).

Deselecting SS2 (status information SS2_ACTIVE=FALSE) terminates standstill monitoring in the drive. If no other safety function is active, the axis can be traversed again without the drive enables having to be reset.

See also

Safety status information (part of the DSDB) (Page 254)

Safety status information (part of the SIDB) (Page 259)

5.8.6 Safe Operating Stop (SOS)As of SIMOTION V4.1

When SOS is selected, the drive continues to follow the setpoint interface and activates the standstill monitoring after the delay time configured in the drive.

Response in the user program:

When SOS is selected (status information SOS_SELECTED=TRUE)), bring the drive to a standstill within the time parameterized in p9551/p9531 and keep it in this state.

When SOS is deselected, the standstill monitoring in the drive will be deactivated; the drive can traverse again without restriction provided no other safety function is active.

See also

Safety status information (part of the DSDB) (Page 254)

Safety status information (part of the SIDB) (Page 259)

5.8.7 Safely-Limited Speed (SLS)As of SIMOTION V4.1

When SLS is selected, the drive continues to follow the setpoint interface and activates the maximum velocity monitoring after the delay time configured in the drive.

When using SLS, ensure that the checkbox **SLS limit value absolute** is selected in the safety dialog.

Response in the user program:

When SLS is selected (status information SLS_SELECTED), the velocity must be reduced to the defined maximum velocity (system variable **driveData.safety.slsSpeedLimitSelected**). For this purpose, ensure that the application observes the distance to the actual limit value monitored

in the drive. Reduction either on the drive side using the p9533 parameter or using a programming reduction factor.

Note

It should be noted that updating the system variables with the status information and the defined maximum velocity is not necessarily simultaneous. Two cases must therefore be allowed for in the user program:

1. SLS is selected in the drive
   The system variable with the status information reports this.
   The system variable **~.slsSpeedLimitSelected** still includes the maximum velocity of the axis and not the limited velocity.
2. SLS is deselected in the drive.
   The system variable with the status information still reports SLS selected.
   The system variable **~.slsSpeedLimitSelected** already contains the maximum velocity of the axis.

Please also see the note regarding the reduction factor for **~.slsSpeedLimitSelected** in Chapter Display of SINAMICS Safety Integrated Functions in SIMOTION (Page 253)

The velocity limit value monitored in the drive can be changed by switching between the SLS levels. Therefore, the system variable **~.safeSpeedLimit** should be continuously monitored for changes while SLS is selected.

In the user program, the activation of velocity reduction or the change to the reduced speed must be cyclically checked and the current velocity setpoint must be compared with the safe maximum velocity.

A detailed procedure can be found in the table below.

Table 5-9 User responses for SLS

	SLS_{new} selected	**SLS active and new SLS speed activated**	
		SLS_{new} is greater than SLS_{old}	**SLS_{new} is less than SLS_{old}**
Production speed is less than the limited speed SLSnew	No response necessary	If production speed is greater than SLS_{old}, increase to production speed is possible	No response necessary
Production speed is greater than the limited speed SLSnew	Deceleration to limited speed SLS_{new} required	Increase to SLS_{new} is permissible	Deceleration to limited speed SLS_{new} required

Production speed = set velocity before selecting a safety function

Note

For active velocity monitoring on a following axis (synchronous coupling and with a fixed gear or coupling via MotionIn interface), the speed of the master axis is reduced or, if this is not possible, synchronous operation during the reduction is canceled.

Compatibility mode with SIDB

The defined maximum speed is saved in the system variable **driveData.driveSafetyExtendedFunctionsInfoData.safeSpeedLimit**.

Please also see the note regarding the reduction factor for **~.safeSpeedLimit** in Chapter Display of SINAMICS Safety Integrated Functions in SIMOTION (Page 261)

See also

Safety status information (part of the DSDB) (Page 254)

Safety status information (part of the SIDB) (Page 259)

5.8.8 Safe Direction (SDI)As of SIMOTION V4.3

After selecting SDI (status information SDI_P_SELECTED or SDI_N_SELECTED), the motion may only be performed in the enabled direction.

Response in the user program:

If the direction of motion is incorrect, the motion must be stopped via the SIMOTION user program and it must be ensured that the motion is only performed in the enabled direction.

See also

Safety status information (part of the DSDB) (Page 254)

Safety status information (part of the SIDB) (Page 259)

5.8.9 Safe Position (SP)

Function SP can be selected optionally with reference to an absolute position. Use of SLP with transfer of the safe position SP corresponds to using SP with reference to an absolute position.

To update the relevant status bits in DSDB, drive parameter p9501.27 must be set.

Using SP with a safe absolute position

As for SLP (Page 270), safe homing of the drive is required.

The position transferred via PROFIsafe can be used in the F-CPU to evaluate the position.

Using SP without a safe absolute position

The position transferred via PROFIsafe can only be used in the F-CPU to ascertain the speed of the drive.

See also *SINAMICS S120 Safety Integrated* Function Manual.

Synchronous transfer of safe position values (SP)

An isochronous transfer of safe positions (isochronous PROFIsafe telegram) is not possible with SIMOTION.

5.8.10 Safely-Limited Position (SLP)As of SIMOTION V4.4

Monitor the absolute traversing range of an axis using SLP. You may define a maximum of two traversing ranges in the drive. The active traversing range is displayed in SLP_LIMIT_BIT0_SELECTED of the system variable **driveData.safety.status1**.

The motion limitation when SLP function is activated must be performed by the user program. SLP activation is displayed in SLP_SELECTED of system variable **driveData.safety.status1**.

To update the status bits relevant for SLP in the DSDB, drive parameter p9501.27 must be set.

Requirement

The safe limited position can only be monitored if the following conditions are met:

- The drive must be homed at standstill while the machine is being commissioned.
- User agreement must be set once during commissioning after homing.
  Exception: It is reset by the user or by a positional tolerance violation in the drive.
- The SLP acceptance test must be performed.

Status **REF_REQUESTED** of the system variable **driveData.safety.status1** indicates whether the drive has been homed and/or whether homing has been requested by the drive. Homing must be performed again, for example, after the CU has been started up.

Note

Status of user agreement

The DSDB does not indicate whether user agreement is lost after successful homing of the drive. Only the S_FAULT bit is set. Evaluation via acyclic read requests may take too long to recognize this state and trigger a response in the user program if SLP is selected or already active.

Make sure a suitable response is implemented in the F-CPU user program in this case. For example, evaluate the "safely homed" bit in the PROFIsafe telegram. For example, trigger SOS if the drive is not safely homed.

Homing and user agreement

Homing must be performed once and user agreement set during commissioning. This can be done in the expert list of the drive or via an HMI.

Therefore home the TO axis and set the user agreement via an HMI:

1. Home the TO axis to a position that you can check in situ and make the axis remain stationary at this position.
2. Transfer the current position of the axis to the drive and apply the home position at standstill (two acyclic **_writeDriveParameter()** requests).
3. Wait for at least five safety cycles, then read the actual position of the drive (r9708[0]) and transfer it to the HMI.
4. Check whether the position of the TO axis is the same as the drive position and that this position corresponds to the machine position.

5. If the positions in step 4. are equal, set the user agreement with a **_writeDriveMultiParameter()** command.

   Note

   If the command is executed and user agreement is active, user agreement will be reset.

6. Now perform the SLP acceptance test.

In the Internet, you will find a program example for homing and setting user agreement as an FAQ (https://support.industry.siemens.com/cs/ww/en/view/100778695).

Violation of the traversing range when the SLP Retraction function is activated

If an SLP limit is exceeded, a safety error is displayed on the drive. A safe stop response is triggered on the drive as error response.

The **INTERNAL_EVENT** in the system variable **driveData.safety.status1** displays the safety error on the Axis technology object.

Retract the axis after an SLP violation as follows:

1. Activate the SDI function in the opposite direction to the violated direction of travel. Depending on the application, implement further safety requirements. For example, additionally activate the lowest SLS level and/or ensure that the drive can only be retracted by the F-CPU when a button is permanently pressed.
2. Deactivate SLP as soon as SDI is active.
3. Acknowledge the safety errors.
4. Travel the axis manually back into the valid SLP traversing range.
5. Activate the SLP function.
6. Deactivate the SDI function as soon as SLP is active.

5.8.11 Safe Cam (SCA)

With SCA, you can define safe cams that are available for evaluation in the F-CPU.

To update the status bits relevant for SCA in the DSDB, drive parameter p9501.27 must be set. With safe cams, for example, other safety functions can be selected in the F-CPU for defined positions.

Requirement

- Use of PROFIsafe telegram 903
- The drive must be homed safely at standstill while the machine is being commissioned. This must be performed as for SLP (Page 270).

5.8.12 SLA - Safely-Limited Acceleration (as of SIMOTION V5.2)

When SLA is selected, the drive continues to follow the setpoint interface and activates the maximum acceleration monitoring in the drive.

The activation of **SLA** is displayed in **SLA_SELECTED** of the **driveData.safety.status1** system variable.

Response in the user program

For selection, and so active SLA (**SLA_SELECTED** status information), the acceleration for traversal commands must be reduced to the defined maximum acceleration of the drive. For this purpose, ensure that the application observes the distance to the actual limit value monitored in the drive.

New functions as of SINAMICS firmware V5.2

As of SINAMICS FW V5.2, SLA has been extended with fine resolution and acceleration filtering. You can parameterize the acceleration limit value and the stop reaction on SLA, and the filter time as of V5.2.

If the determination of the acceleration results in very noisy signals, the drive cannot reasonably monitor the acceleration. Remedy: In this case, increase the **"SLA filter time" (p9576)**. Note that **SLA** reacts with a delay when you increase the filter time.

Transmission via SIC

Once **SLA** has been parameterized and selected, the monitoring results are also transmitted in **SIC** in status word **S_ZSW1B.8** (Section "Safety Info Channel and Safety Control Channel"). You will find this status word in telegrams 700 and 701.

5.9 Safety brake test as of SIMOTION V4.4

As of V4.4, SIMOTION support the SINAMICS safe brake test. It is triggered by a SIMOTION command from the Axis technology object. During the SBT, the drive no longer follows the controller setpoint.

During the test, the load torque applied at the motor is measured once. The test torque is then increased n times depending on the direction and the number of brakes, maintained for a period of time and then reduced again.

The brake test can be performed for a maximum of two brakes in one test cycle. One internal and one external or two external brakes are possible if there are two brakes. Each brake can be tested against two different holding torques.

Requirements for the safe brake test

- The axis must be enabled with the **_enableAxis()** command in mode **enableMode**=POWER with **servoControlMode**=INACTIVE.
- All brakes present on the motor must be open.

Perform the safe brake test for an internal brake as follows:

1. Start the test by calling the **_executeAxisSafetySbt()** function in the user program.
2. Monitor the status if necessary via the **driveData.safety.sbtState** system variable.
3. Wait until the function has finished and then check the return value of the function.

Perform the safe brake test as follows if there is at least one external brake:

1. Start the test by calling the **_executeAxisSafetySbt()** function in the user program.
2. If there are two external brakes:
   Check the status **SBT_BT_BRAKE2** of the system variable **driveData.safety.status2** to see whether brake 1 or 2 is tested.
3. Check the status **SBT_BT_BRAKE_CLOSE_REQ** of system variable **driveData.safety.status2** cyclically in the user program. Open or close the external brake, depending on the status.
4. Report the current status of the external brake to the system using the **_setAxisSafetySbtExtBrakeState()** function.
   After a change of status of **SBT_BT_BRAKE_CLOSE_REQ**, the feedback must be provided within 11 seconds.
5. Monitor the status if necessary via the **driveData.safety.sbtState** system variable.
6. Repeat steps 2 to 5 until the **_executeAxisSafetySbt()** function has ended and then check the return value of the function.

Note

External brakes

Take into account the brake opening and closing times for status feedback.

5.10 Pulse enable evaluation (when standard settings are deactivated)

This chapter is only applicable if standard settings such as **Use symbolic assignment** have been deactivated.

For *SINAMICS Safety Integrated Functions* support, the status of the pulse enable of the drive on the Axis technology object must be evaluated. This status information is transferred to various positions in the telegram depending on the telegram type.
This means the **driveControlConfig.pulsesEnabled.pzdNumber** and

driveControlConfig.pulsesEnabled.bitNumber configuration data as appropriate for the telegram configuration must be set as follows:

- Standard telegram 2-6
  pzd = 4; bit = 10 (ZSW2; "Enable pulses")
  (possible as of SINAMICS V2.6)
- Ten SIEMENS telegrams
  pzd = 5; bit = 13 (message word; "Enable pulses")
- Non-assigned telegrams
  The user has sole responsibility for ensuring that both of the configuration data settings referred to are set according to the position of the pulse enable status bit within the telegram.

Note

- Standard telegram 1 cannot be used to support safety functions SS1 and SS2, since it does not transmit the status of the pulse enable.
- Depending on the telegram used, set p2038 as follows:
  - For standard telegrams in the drive p2038=0
  - For SIEMENS telegrams in the drive p2038=1

5.11 Messages and alarms

The messages are displayed as technological alarms of the *SINAMICS Safety Integrated Extended Functions* on the SIMOTION Axis technology object.

Technological alarms:

- **Alarm 50023: Drive makes transition to autonomous state (as of SIMOTION V4.2)**
  If the drive behaves autonomously, e.g. braking on the OFF3 ramp in response to an SS2 selection, alarm 50023 is signaled.
- **Alarm 50201: Safety alarm in the drive (see SINAMICS Safety Message Buffer)**
  A new event is present in the safety message buffer of the drive.

Note

Acknowledgment of alarm 50201

Earlier than SIMOTION V4.4:
Then alarm can only be acknowledged permanently if there are no longer any entries in the safety message buffer.

As of SIMOTION V4.4 (with DSDB):
Then alarm can only be acknowledged permanently if neither S_FAULT_PRESENT nor INTERNAL_EVENT is set.

- **Alarm 50202: SINAMICS Safety Integrated Extended Function is selected**
  Alarm 50202 is signaled if the safety functions SS2/SOS/SLS are selected and, as of V4.4, SDI/SLP as well.
  Exceptions:
  - On transition to SOS with SS2 active (automatic transition or explicit selection), no new 50202 alarm is activated.
  - If the *SINAMICS Safety Integrated Extended Functions* are activated using PROFIsafe, alarm 50202 is generated after a SINAMICS reset without a safety function being selected.
  - If the *SINAMICS Safety Integrated Extended Functions* are activated using PROFIsafe, alarm 50202 is generated if the SIMOTION CPU or F-CPU switches to STOP.

  The alarm is used only for display in the SCOUT or HMI.
  User responses should reference system variable **driveData.safety.status1/ driveData.safety.status2** or **driveData.driveSafetyExtendedFunctionsInfoData.state**.
- **Alarm 50203: SINAMICS Safety Integrated Extended Function is deselected**
  Alarm 50203 is signaled if the last active safety function SS2/SOS/SLS is deselected and, as of V4.4, SDI/SLP as well.
  Exception:
  When controlling the *SINAMICS Safety Integrated Extended Functions* via PROFIsafe, the PROFIsafe driver is acknowledged (user action on the F-CPU side as an additional prerequisite for full drive enable (drive parameter r46.8)), alarm 50203 is generated without a safety function being deselected.
  The alarm is used only for display in the SCOUT or HMI.
  User responses should reference system variable **driveData.safety.status1/ driveData.safety.status2** or **driveData.driveSafetyExtendedFunctionsInfoData.state**.
- **Alarm 50209: Error on brake test**
  Alarm 50209 is signaled if an error occurs when the SBT brake test is performed.

Note

Alarm 50023 only becomes active on the Axis technology object if configuration data **TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation** is set to YES.

Alarms 50201, 50202 and 50203 are only displayed on the TO axis when DSDB and SIDB are activated.

No safety alarm is triggered when STO and SS1 are selected. Alarm 20005 reason 4 is signaled if the pulses are deleted by the drive. There is no alarm to signal that the functions have been deselected.

Illustration of safety alarm behavior (50202, 50203) based on a pulse diagram

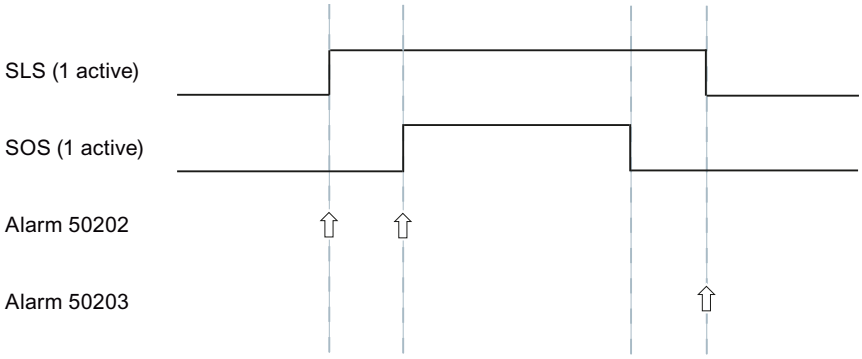


Figure 5-6 Pulse diagram illustrating safety alarm behavior

5.12 SINAMICS Safety Message Buffer

Errors in the safety function in the drive or stop responses that are triggered by the monitoring functions initiate messages.

Messages for the *SINAMICS Safety Integrated Basic Functions* are displayed in the drive's message buffer (r947).

Messages for the *SINAMICS Safety Integrated Extended Functions* are displayed in the safety message buffer (r9747).

Table 5-10 Basic structure of the message buffer

Parameter	Meaning
r944	Message buffer change count
r947 [0...63]	Message code
r949 [0...63]	Additional information
r952	Messages counter

Table 5-11 Basic structure of the safety message buffer

Parameter	Meaning
r9744	Message buffer change count
r9747 [0...63]	Message code
r9749 [0...63]	Additional information
r9752	Messages counter

Acknowledge

Errors in the safety functions or stop responses may only be acknowledged on a fail-safe basis. A distinction must be made here between the *Basic Functions* and *Extended Functions*.

Basic Functions: Acknowledgment via two-channel selection/deselection by STO.

Extended Functions: Acknowledgment of safety PLC via the PROFIsafe signal Internal_Event_Acknowledge - Bit 7 of the PROFIsafe control word or via a terminal on TM54F or F-DI OnBoard for D410-2.

With extended alarm acknowledgment, the messages can be acknowledged via the acknowledgment mechanism of the *Basic Functions*.

Note

Limitation for SINAMICS V2.5

Fail-safe acknowledgment via PROFIsafe bit or a fail-safe digital input following violation of SLS limit value is only possible if STOP C (p9563=2) has been configured as a fault response. For other stop responses, an acknowledgment is only possible by means of Power Off/Power On.

For further information, see the *SINAMICS Safety Integrated* Function Manual.

Display and evaluate an error message

An entry in the safety message buffer is displayed by S_FAULT_PRESENT in the system variable **driveData.safety.status1** (Bit 30).

The occurrence of a safety STOP response is displayed by INTERNAL_EVENT in the system variable **driveData.safety.status1** (Bit 31).

If this bit is set, the **50201 Safety Alarm in the drive** alarm will be triggered in SIMOTION.

The **_readDriveFaults()** system function can be used to read out the message code.

As of SIMOTION V4.2 it is possible to fetch data from the safety message buffer directly by setting faultType=SAFETY (drive parameter p9747). The additional information for the message code can be read out by the user program using the system function **_readDriveParameter()** from drive parameter r9749.

Compatibility mode with SIDB

An entry in the safety message buffer is displayed by S_FAULT_PRESENT in the system variable **driveData.driveSafetyExtendedFunctionsInfoData.state** (Bit 15). If this bit is set, the **50201 Safety Alarm in the drive** alarm will be triggered in SIMOTION.

5.13 Isochronous PROFIBUS operation

PROFIsafe data must be transferred isochronously.
When PROFIsafe data is transferred isochronously, additional configuration and programming steps are required or are useful.

Essential settings for isochronous PROFIsafe operation

1. Synchronization of the SIMOTION CPU with the F-CPU
   In the I-slave F-proxy and I-device F-proxy topologies, synchronize the connection from the SIMOTION user program to the **_enableDpInterfaceSynchronizationMode()** system function.
2. Consistent process image partition for the safety program
   Make sure that the update of the process image partition for isochronous operation is not performed during the processing of the safety program.
   An example can be found in Chapter *F-CPU: Isochronous operation and safety operation* of the FAQ Actuating internal drive safety functions via SIMOTION and PROFINET with PROFIsafe (https://support.industry.siemens.com/cs/ww/en/view/50207350).

Additional recommended settings for isochronous PROFIsafe operation

1. Check the "synchronous" status on the SIMOTION controller
   Check for error-free synchronization through an I/O status query of the PROFIsafe PZD ("synchronous").
2. Call the safety program on the F-CPU in a multiple of the SI monitoring cycle
   Select the cycle of the cyclic interrupt in which the safety program is called (typically OB35) as a multiple of the SI cycle (drive parameter p9500).
   In this way, you avoid problems during startup (SINAMICS fault 1611/1711 with fault value 6165).

5.14 Restrictions after switching from compatibility mode (SIDB) to standard (DSDB).

The following points should be noted when switching from compatibility mode (SIDB) to standard (DSDB):

- Safe Speed Monitor, **SSM**
  Safe speed monitoring via the SSM bit is no longer available after switching to standard (DSDB).
- System variable **safeSpeedLimit**
  The system variable is only provided after switching to standard (DSDB) if the following conditions are met:
  - The *SINAMICS Safety Integrated Extended Functions* are configured.
  - The **SLS limit value absolute** option is selected.
- Alarm 50201
  The alarm is only triggered by S_FAULT_PRESENT if the SIMOTION Runtime is less than V4.4.
  As of SIMOTION Runtime V4.4, the alarm can also be triggered by INTERNAL_EVENT.
  See also SINAMICS Safety Message Buffer (Page 276).
- Always evaluate the pulses enabled bit
  The configuration data item **TypeOfAxis.DriveControlConfig.pulsesEnabledEvaluation** must be set to YES if the DSDB is used and correct transfer of the pulses-enabled bits must be ensured.

See also

Axis configuration (Page 254)

Safety channel and safety channel types (when symbolic assignment is deactivated) (Page 251)

Overview - Support of SINAMICS Safety Integrated Functions on the Axis technology object (Page 245)

Safety status information (part of the DSDB) (Page 254)

Part II Hydraulic Functionality

6

6.1 Hydraulic functionality overview

The hydraulic functionality is contained in the **Axis technology object (TO)**.

Like the electric axis, the hydraulic axis can be configured with the following technologies:

- Speed-controlled axis
- Positioning axis
- Synchronous axis

The user view of the electric and hydraulic axes is maintained together, to the extent possible.

Example:

- Motion commands
- Axis settings

Important differences compared to the electric axis include:

- Allowance for a valve characteristic curve
- Special compensation
- A valve can be switched between more than one axis
- No lower-level velocity limiting or torque control in the actuator / drive
- Separate actuators/valves for flow (Q valve) and force/pressure (P valve), if necessary

The hydraulic functionality is explained in the following sections, building on the description of the axis.

See also

Overview of axis technologies (Page 21)

General information about axes (Page 17)

Fundamentals of hydraulic functionality 7

7.1 Axis settings / drive assignment

7.1.1 Overview of axis settings / drive assignment

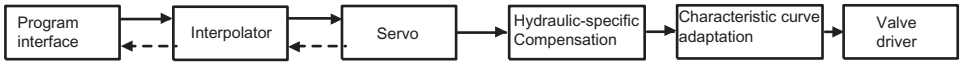


Figure 7-1 Overview of motion control for axes with hydraulic functionality

Valve types:

- **Q valve** (valve for closed-loop control of volumetric flow, path valve)
  Hydraulic valve for control of direction and volume of a material flow, suitable for closed-loop velocity control
- **PQ valve**
  Special Q valve suitable for force, position, velocity, and pressure control
- **P valve** (pressure limiting valve)
  Valve used for limiting the system pressure, suitable for protecting a hydraulic system against excessive pressure (overpressure protection)

7.1.2 Setting as a real axis with hydraulic functionality

The manipulated variable values for the final controlling elements (valves) are defined as analog or direct values.

Options are:

- Analog outputs on the C2xx
- Analog outputs in the I/O area
- Analog outputs on a PROFIBUS module, e.g. SINUMERIK ADI4, SIMATIC IM174
  With SIMODRIVE ADI4 and SIMATIC IM174, PROFIdrive profile standard message frame 3 must be used.

A specific enable signal is available for each final controlling element (see the drive enable signal for electric axes).

The enable signal for the Q valve is set/reset via the qOutputEnable parameter in the **_enableQFAxis()/_disableQFAxis()** command. The status of the enable signal is indicated in the **actorMonitoring.driveState** and **actorMonitoring.power** system variables. When the hydraulic output is connected to the IM174, these system variables are formed using the bits returned in the status word. The status of the Q output is displayed in **actorMonitoring.qOutputState**.

The enable signal for the P valve is set/reset via the fOutputEnable parameter in the **_enableQFAxis()/_disableQFAxis()** command. The status of the enable signal for the P valve is

displayed with the **actorMonitoring.fOutputEnable** system variable. The status of the P output is displayed in **actorMonitoring.fOutputState**.

When defining an axis as an axis with hydraulic functionality, it is possible to assign several axes to a final controlling element during configuration. During runtime, they are assigned using the **_enableQFAxis()/_disable QFAxis()** enable/disable commands.

A real axis with hydraulic functionality has manipulated variables for the flow rate (Q, Q valve) and, if required, a separate manipulated variable for direct force/pressure limiting or force/pressure control (F, P valve).

On axes with a P valve, the technological commands and system variables for force/pressure limiting can be used to specify the force and pressure.

The addresses in the I/O area for outputting the Q value and, if applicable, for outputting the F/P value can be set on the axis.

Note

You can find application examples for hydraulic axes under *FAQs > Technology* in the *SIMOTION Utilities & Applications* provided with SIMOTION SCOUT.

See also

Access to the same final controlling element from multiple axes (Page 303)

7.1.3 Setting as a real axis with Q valve only

With hydraulic axes with Q valve only, the functions for traversing the axis, for velocity limitation, for force/pressure control, and for force/pressure limiting are available as with the electric axis.

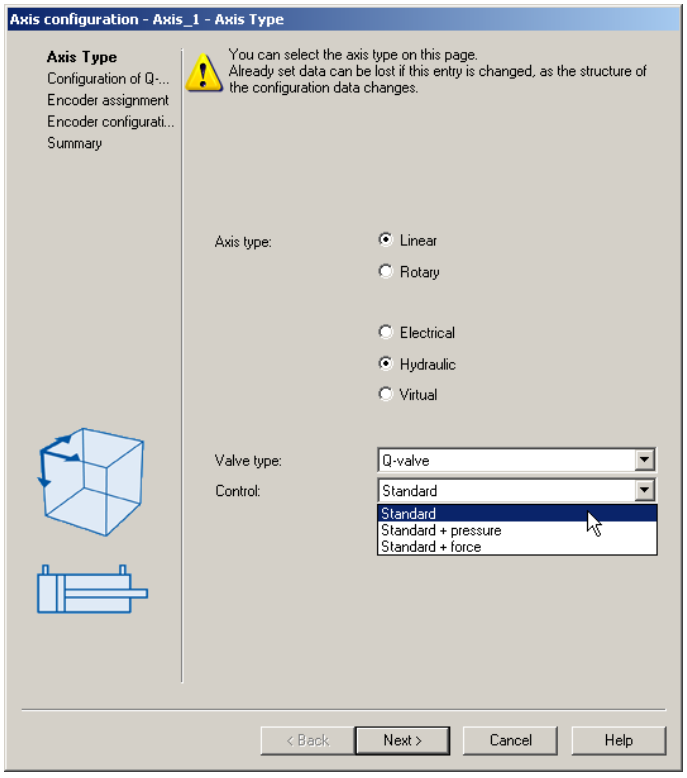


Figure 7-2 Setting the axis type

Table 7-1 Setting the control

Standard	Motion via Q valve
Standard+pressure	Motion and force/pressure control via Q valve Unit: Pascal, bar
Standard+force	Motion and force/pressure control via Q valve Unit: Newton

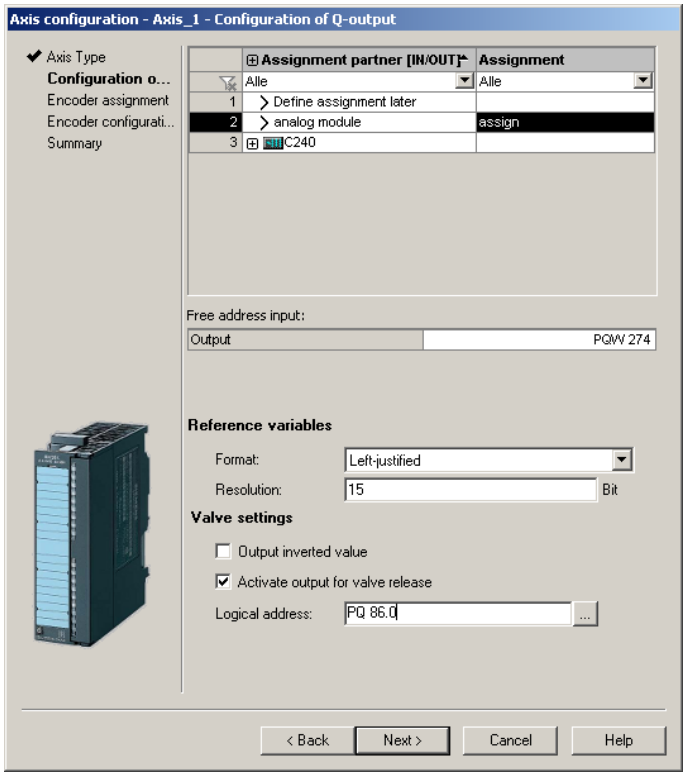


Figure 7-3 Example configuration of the Q output using analog output module

The bit resolution of the analog output module is set under Resolution.

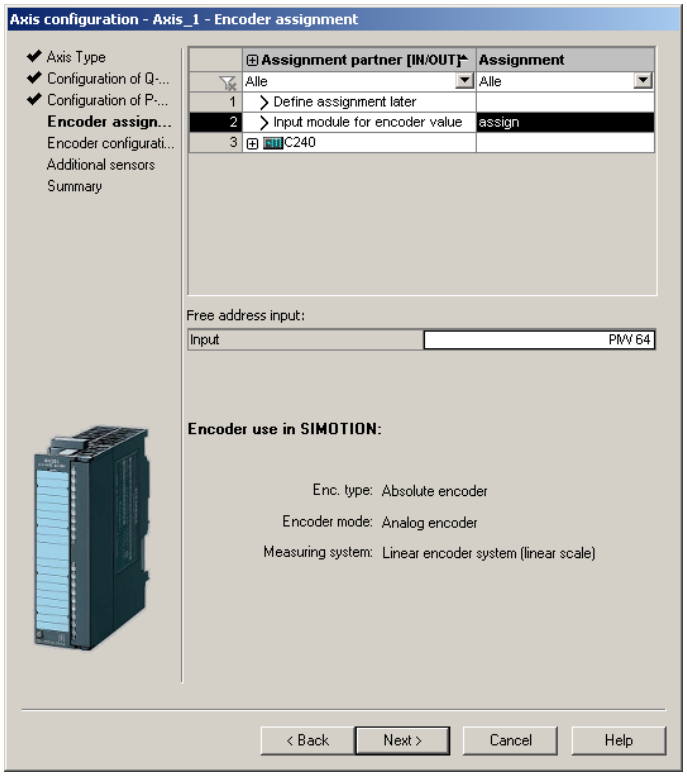


Figure 7-4 Example of configuration for the encoder assignment

The logical hardware address of the input module can be found in the HW Config in SCOUT.

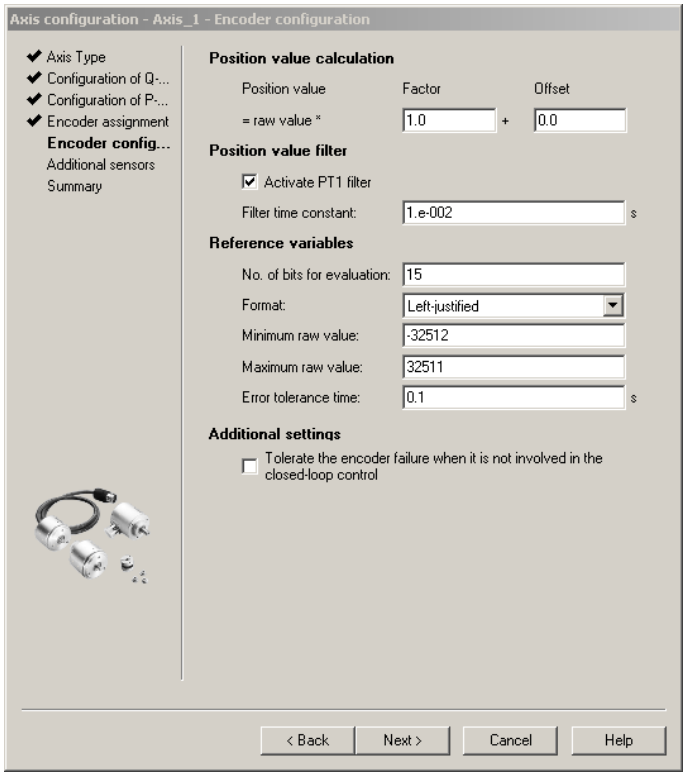


Figure 7-5 Example of configuration for the position value

The **Factor** (scaling factor) and **Offset** are used to convert the internal value into a physical variable that can be represented.

The value can be output as right-justified or left-justified within the word width (16 bits).

The minimum and maximum raw values are the limitations. If the measured actual value is outside these limits, technological alarm **50013: The permissible range limit has been exceeded** is issued and the internal actual value is set to the limit value.

The error tolerance time states how long an error can be pending before a technological alarm is set.

See also

Setting for the hydraulic axis type (Page 29)

TypeOfAxis configuration data (Page 31)

7.1.4 Setting as a real axis with Q valve + P valve/F output

With hydraulic axes with Q valve + P valve/F output, the functions for traversing the axis (Q valve) are available. In addition, a manipulated variable can be controlled and output on the P valve/F output.

The following variants are available:

- Two valves (P valve and Q valve)
- One valve with two connections (analog)
- One variable-capacity pump

On the Axis technology object, one analog controller output is configured and controlled for the Q valve (flow, velocity) and one for the P valve (force/pressure limiting).

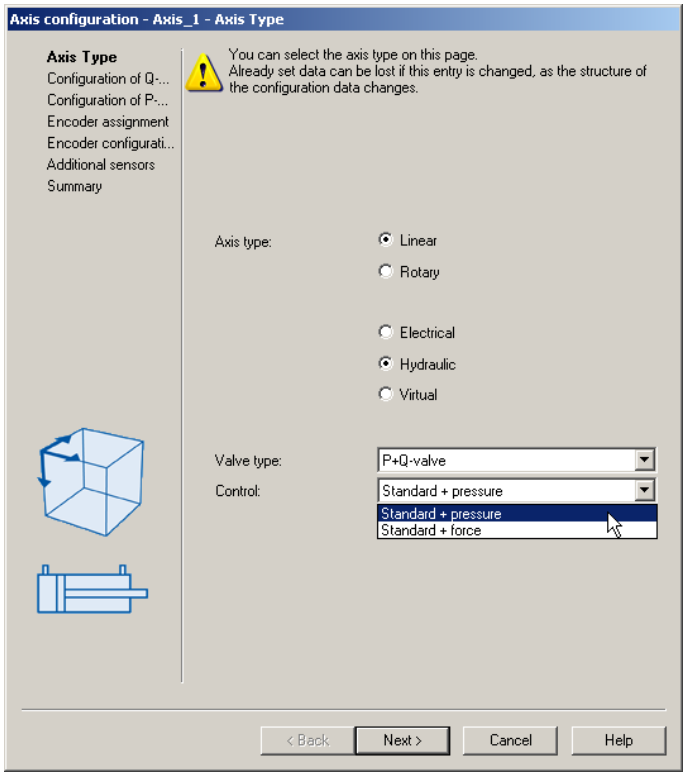


Figure 7-6 Setting the axis type

Table 7-2 Setting the control

Standard+pressure	Motion at the Q output and pressure limiting at the P valve/F output Unit: Pascal, bar
Standard+force	Motion at the Q output and force limiting at the P valve/F output Unit: Newton

Hydraulic axes with Q valve and P valve/F output do not have any pressure control (pressure control commands are rejected); in this case, the pressure limiting commands act on the P valve/F output. The pressure limiting value in the command is issued as a manipulated variable at the P valve/F output.

See also

Setting for the hydraulic axis type (Page 29)

TypeOfAxis configuration data (Page 31)

7.1.5 Setting an axis as a real axis with P valve only (V3.2 and higher)

At the P valve (F output of the axis), a time-related or actual-position-related profile or a manipulated variable can be output. This means that no position control, velocity control or force/pressure control takes place. There are no force/pressure sensors required, but they can be configured.

Position encoders cannot be configured.

You must configure a speed-controlled axis.

The following commands are possible on this axis:

- _resetAxis()
- _resetAxisError()
- _getStateOfAxisCommand()
- _bufferAxisCommandId()
- _removeBufferedAxisCommandId()
- _enableQFAxis()
- _disableQFAxis()

Commands to output a force/pressure limiting value or a force/pressure limiting profile can also be used:

- _enableForceLimitingValue()
- _enableTimeLockedForceLimitingProfile()
- _enableMotionInPositionLockedForceLimitingProfile()
- _disableForceLimiting()

Position encoders cannot be configured or activated.

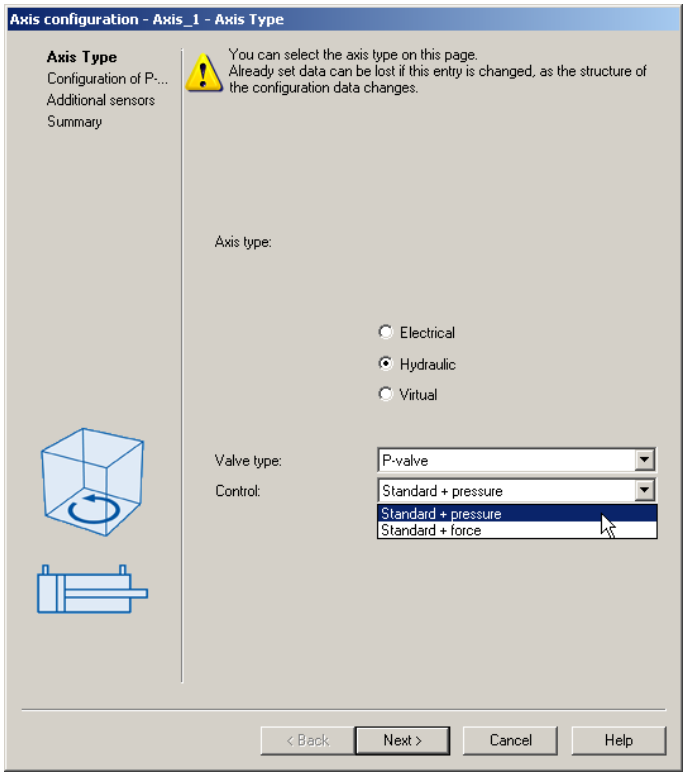


Figure 7-7 Creating an axis with P valve only

Table 7-3 Setting the control

Standard+pressure	Force/pressure control at the P valve/F output Unit: Pascal, bar
Standard+force	Force/pressure control at the P valve/F output Unit: Newton

See also

Overview of axis technologies (Page 21)

Setting for the hydraulic axis type (Page 29)

TypeOfAxis configuration data (Page 31)

7.1.6 Setting an axis as a real hydraulic axis without a valve (axis simulation)

Hydraulic axes with manipulated variable output directly in the I/O area cannot be defined for axis simulation.

Axis simulation with hydraulic axes is only possible with a Q valve and with a digital drive interface or onboard C2xx.

For more information, see Setting as a real axis without drive (axis simulation) (Page 60).

7.2 Input limits, technological limiting functions

Information about the system input limits and technological limitations is contained in Input limits, technological limiting functions (Page 79).

7.3 Settings for axis and encoder mechanics

Note

With the hydraulic axis, not all of the setting options for the electric axis are required or displayed.

See also

Overview of setting options for axis and encoder mechanics (Page 79)

7.4 Defaults

Information about default settings for system variables can be found in Defaults (Page 84).

7.5 Homing

A short summary for homing, absolute encoder and incremental encoder is contained in Overview of homing (Page 84).

7.6 Differential pressure measurement (V3.2 and higher)

The actual pressure value can be set as a pressure difference.

The pressure difference is implemented as a separate sensor type that determines the resulting differential pressure from two sensor-measured values using the following formula:

$F_{act} = (p_A \times A_A - p_B \times A_B) \times F_{factor}$
(F_{factor}: force factor)

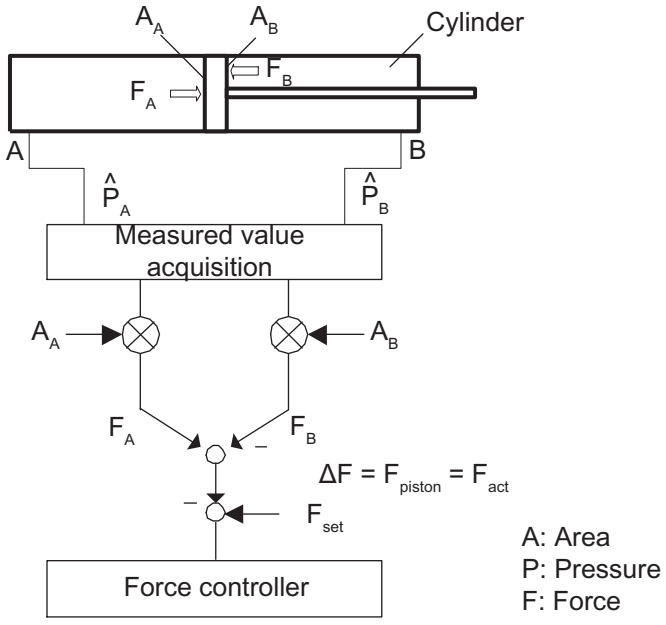


Figure 7-8 Example of a pressure difference measurement

Like the measuring sensors, the pressure difference is defined as force/pressure sensors.

Proceed as follows to specify the setting via the expert list:

- Configure at least two pressure sensors on the axis.
- In the expert list, increase the value of **TypeOfAxis.NumberOfAdditionalSensors.number** by 1.
- Set the other sensor as a "differential pressure sensor" with PRESSURE_DIFFERENCE_MEASUREMENT via the **TypeOfAxis.NumberOfAdditionalSensors.AdditionalSensor_n.additionalSensorType** configuration data element.
- The sensors whose values are being used and the individual factors are set in the elements of the **TypeOfAxis.NumberOfAdditionalSensors.AdditionalSensor_n.pressureDifferenceMeasurement** structure.

The pressure difference may be the pressure difference at a cylinder or another pressure difference. All sensors used for the pressure difference measurement must be configured on the same technology object.

The pressure difference measurement may also be applied to the electric axis.

If pressure difference measurement is used, force control is available in accordance with the function definition. In this case, the pressure sensors are also displayed in the force unit, dependent upon the system.

See also

Using the expert list for an axis (Page 229)

7.7 Differential position measurement (V3.2 and higher)

Information on differential position measurement is contained in Differential position measurement (V3.2 and higher) (Page 99)

7.8 Monitoring/limiting functions

The monitoring/limiting functions of the electrical axis are applied.

For the hydraulic axis, the pressure control and limiting functions can also be applied to the drive axis.

See also

Overview of monitoring/limiting functions (block diagram) (Page 100)

7.9 Motion profiles

The motion profiles that can be applied are the same as for the electrical axis.

See also

Overview of traversing with user-defined motion and force/pressure profiles (Page 203)

7.10 Hydraulic axis with position control/velocity control

7.10.1 Position control for setting a positioning axis with hydraulic functionality

Block diagram of the hydraulic axis with closed-loop control:

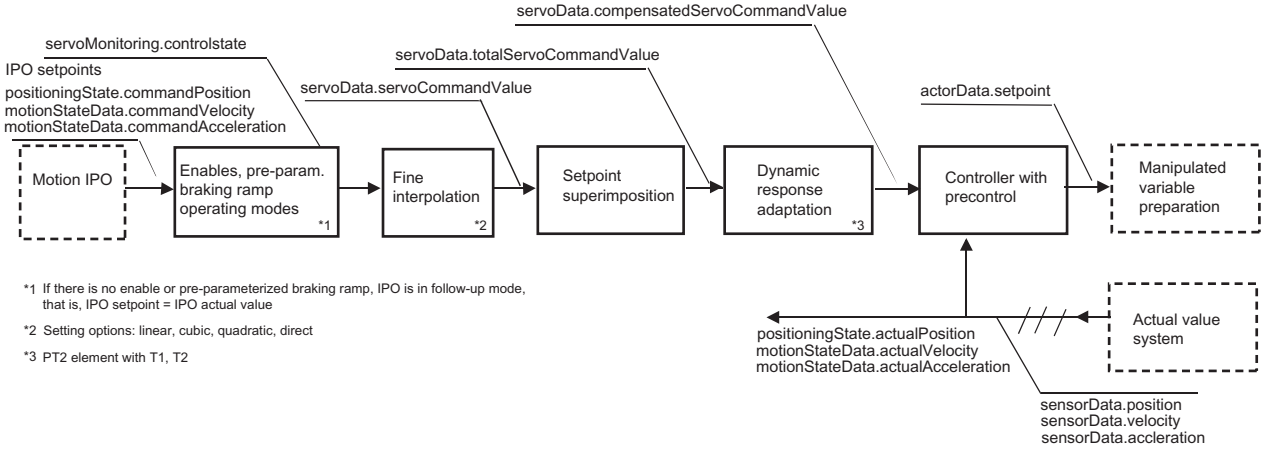


Figure 7-9 Overview of hydraulic axis with closed-loop control

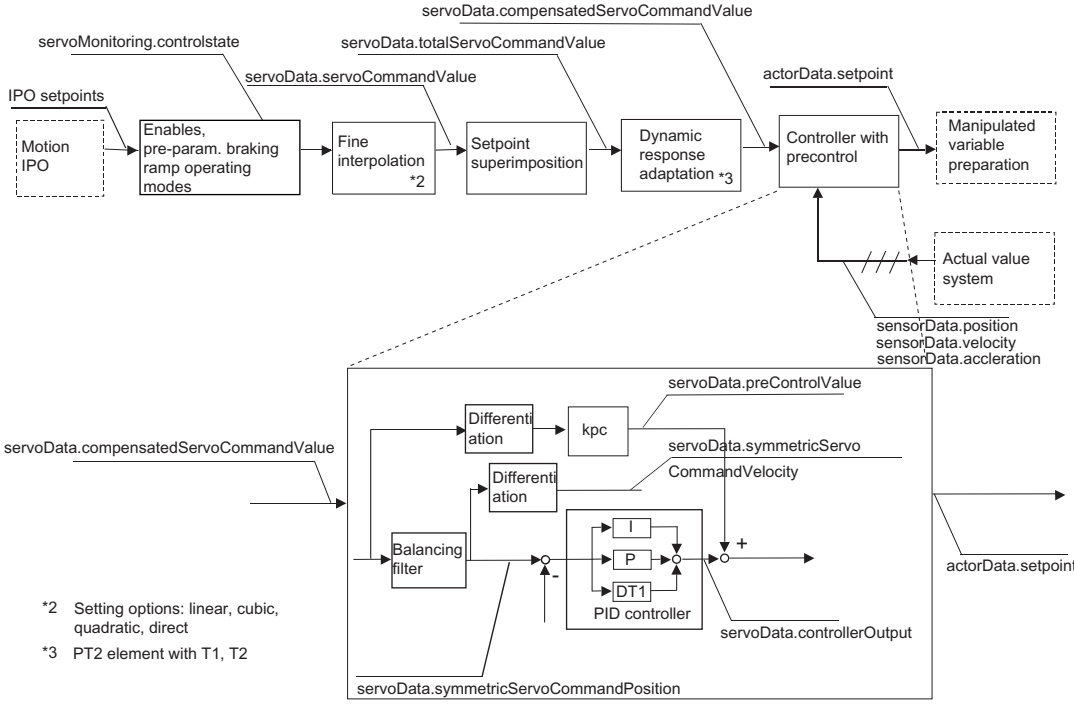


Figure 7-10 PID controller with precontrol

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

Remark

You can specify during configuration whether the D component of the controller refers to the control deviation or the actual value (in the **ControllerStruct.conType** configuration data element).

The dynamic behavior of the process is taken into account in the balancing filter.

With the hydraulic positioning axis, the dynamic response of the position control loop is specified in the **dynamicData.positionTimeConstant** configuration data element. The dynamic response of the process is specified in **dynamicQFData.qOutputTimeConstant**.

The hydraulic position axis can now be activated in non-position-controlled mode with the **_enableQFAxis()** command via the parameter **movingMode**= SPEED-CONTROLLED (as of V4.1 SP1).

See also

Overview of positioning axis with position control (Page 110)

Position control (Page 111)

7.10.2 Velocity controller when setting speed-controlled axis with hydraulic functionality

When an axis is set as a drive axis with hydraulic functionality, the velocity controller is calculated in SIMOTION.

With the electric drive axis, the velocity controller is included in the drive and the control specifies the speed setpoint.

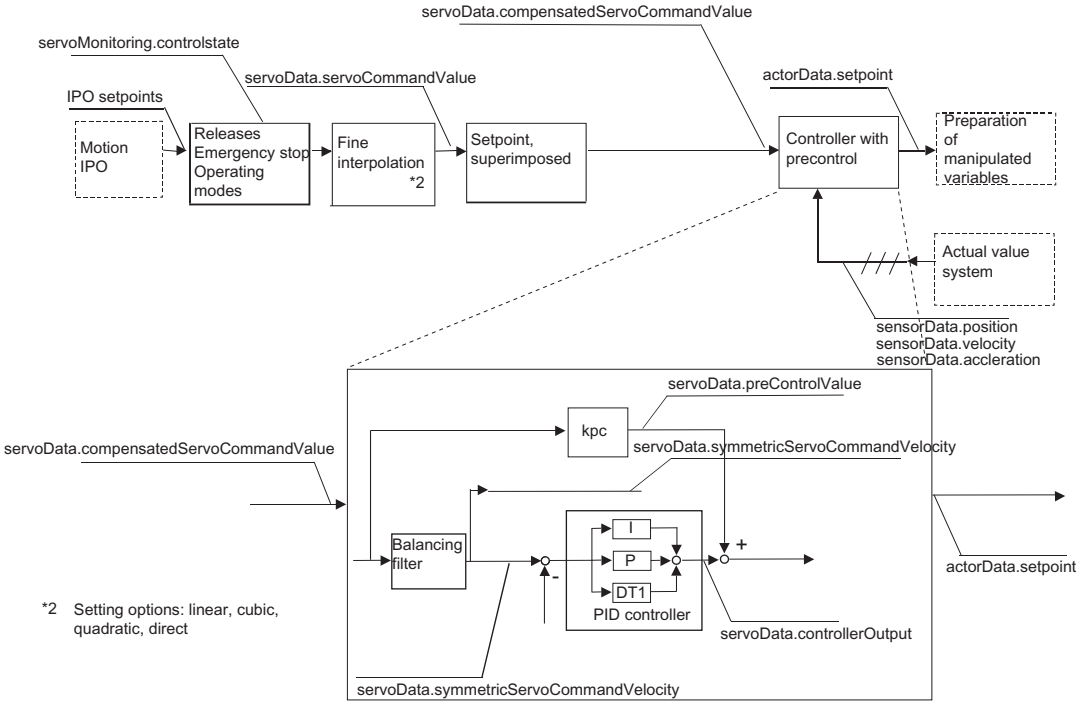


Figure 7-11 Block diagram of the velocity controller

Setting an axis as a drive axis with hydraulic functionality

Functionality:

- PID controller available as velocity controller
- Precontrol of the manipulated variable available
- Velocity monitoring available

If a controller is applied, the axis is traversed with closed-loop velocity control. If no controller is applied, the axis is traversed with open-loop velocity control.

Note

The closed-loop velocity-controlled hydraulic axis cannot be moved with open-loop velocity control via movingMode=SPEED_CONTROLLED.

The status of the velocity controller is displayed in the **servoControl.controlState** system variable.

Controller error monitoring is displayed in **servoMonitoring.controllerDifferenceError**.

The components **dynamicFollowingError**, **dynamicFollowingWarning**, **positioning**, and **standstill** in **servoMonitoring** are irrelevant.

With setting **ControllerStruct.conType** = DIRECT, the controller can be disconnected.

Monitoring

Data and statuses for the velocity controller on the axis are displayed in the **servoData** components.

See the list manuals for details.

For the velocity controller, the setpoints, actual values, and superimpositions refer to the velocity. Position-related information such as **actualPosition** or **symmetricServoCommandPosition** are irrelevant.

preControlValue displays the precontrol value.

In **servosettings**, the **additionalCommandValue** setting refers to the velocity.

Standstill monitoring is not available on the drive axis.

The standstill signal is available.

Remarks

Dynamic response adaptation is not active on the axis with hydraulic functionality.

Superimpositions are in effect.

With the hydraulic drive axis, the dynamic response of the velocity control loop is specified in the **dynamicQFData.velocityTimeConstant** configuration data element; the dynamic response of the process is specified in the **dynamicQFData.qOutputTimeConstant** configuration data element.

See also

Overview of positioning axis with position control (Page 110)

7.10.3 Preparation of manipulated variables for axis with hydraulic functionality

Preparation of manipulated variables for axis with hydraulic functionality, Q output

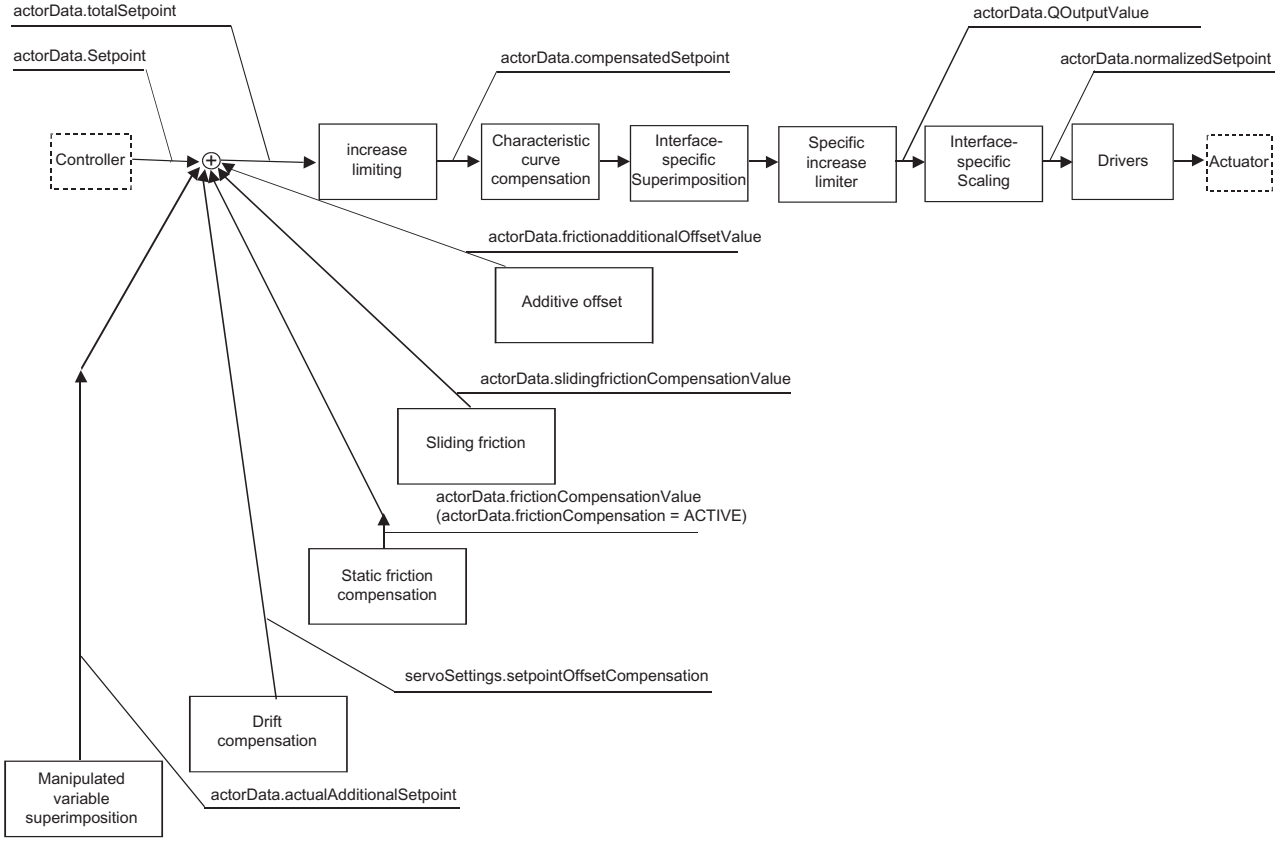


Figure 7-12 Preparation of manipulated variables for axis with hydraulic functionality, Q output

Preparation of manipulated variables for axis with hydraulic functionality, F output

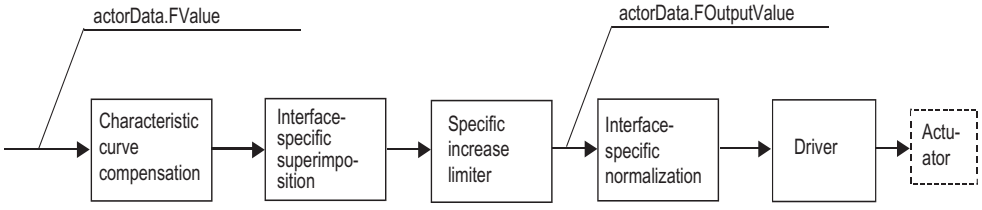


Figure 7-13 Preparation of manipulated variables for axis with hydraulic functionality, F output

Note

For hydraulic axes, a valve characteristic can be assigned to each output (P or Q). If this is not the case, a linear characteristic is used. Here, 100% corresponds to the limit value of the axis (**TypeOfAxis.MaxVelocity**, **TypeOfAxis.MaxForceCommandData**).

The interface-specific superimposition is specified as a percentage (%), e.g. for specifying manipulated variables when recording of the characteristic curve.

An increase limiting is specified in the command for output activation and for activation of the characteristic curve because the manipulated variable must be continuos on the valve, e.g. the increase limiting (ramp function) is used to prevent a step change. If the increase limiting is active, the I component is retained in the position controller or force/pressure controller.

The specific increase limiting after allowing for the valve characteristic is only in effect for the transition at **_setQFAxisFCharacteristics()**, **_setQFAxisQCharacteristics()**, and **_disableQFAxis()**, **_enableQFAxis()**. The increase limiting is specified in the command (default in **userDefaultQFAxis.maxDerivative**).

Note

If the RELEASE_DISABLE or OPEN_POSITION_CONTROL alarm responses occur, the setpoint or manipulated variable = 0 will be output. For hydraulic axes, the value 0 will be converted into an appropriate output value using the valve characteristic.

A velocity encoder must be present for speed-controlled operation. The control supports the following velocity encoders:

- Pulse encoder via SM335 or E510
- Analog velocity encoder via analog input modules

See also

Overview of positioning axis with position control (Page 110)

7.10.4 Manipulated variable filtering (as of V4.1 SP1)

A man. var. filter can be set as a PT1 filter in the **setpointFilter** configuration data element. This filter acts after the controller and after the precontrol value and the additive manipulated variable value (additionalSetpoint) have been added.

A change in the filter data takes effect immediately.

7.10.5 Compensations that are active only on the axis with hydraulic functionality

A static compensation component (additive sliding friction) and a compensation component proportional to velocity (sliding friction) can be set on the axis with hydraulic functionality.

These settings are made in the Closed-Loop Control view in the project navigator under the axis. The Further Compensations tab appears when Expert mode is selected.

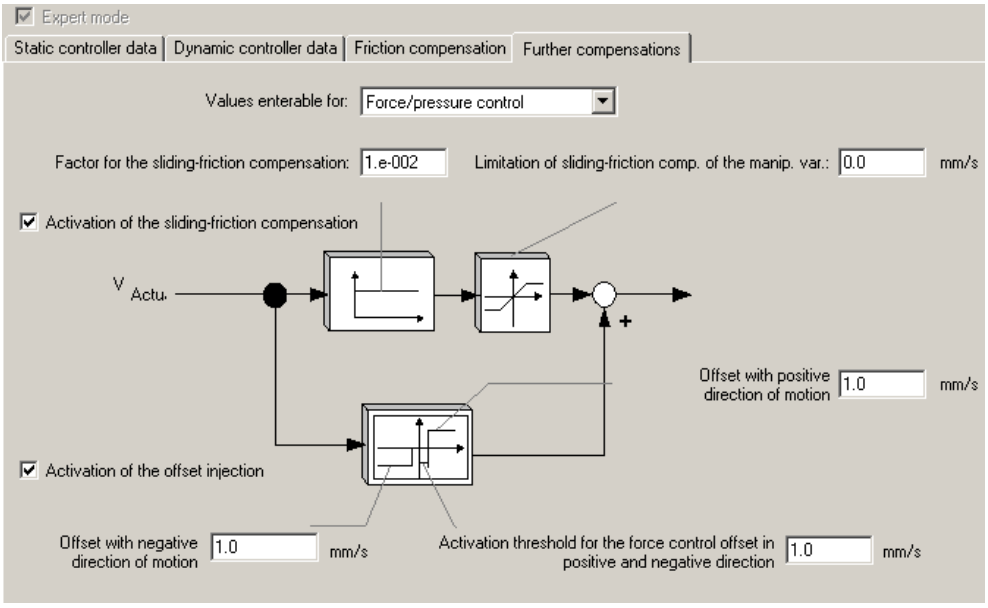


Figure 7-14 Sliding-friction compensation/additive sliding-friction compensation on an axis

Sliding-friction compensation

Sliding friction is applied relative to the velocity setpoint in axis motion via motion specification, and relative to the actual velocity value in axis motion via force/pressure specification.

The factor for sliding-friction compensation can be set specifically for axis motion via motion specification (**factorMotionControl**) and for axis motion via force/pressure specification (**factorForceControl**) in the structure elements for **SlidingFriction**.

The current value of the sliding-friction compensation is displayed in the **actorDataSlidingFrictionCompensationValue** system variable.

Additive sliding-friction compensation (offset application)

Additive sliding friction is applied relative to the velocity setpoint in axis motion via motion specification, and relative to the actual velocity value in axis motion via force/pressure specification.

The factor for **additive sliding friction** can be set specifically for axis motion via motion specification, and separately for each velocity direction, in factorMotionControlPositive and factorMotionControlNegative; it can be set for axis motion via force/pressure specification in factorForceControlPositive and factorForceControlNegative.

These factors are set in the structure elements for AdditionalOffset. The current value is set in **frictionAdditionalOffsetValue** (direction-dependent sliding-friction compensation value).

The current value for additive sliding friction is indicated in the **actorDataFrictionAdditionalOffsetvalue** system variable.

See also

Overview of positioning axis with position control (Page 110)

7.10.6 Consideration of valve characteristic when specifying a hydraulic axis

The existing non-linearity between the technological manipulated variable, for example, an oil flow rate (Q) or a velocity or force/pressure value (F), and the manipulated variable value of the valve is mapped by means of a characteristic curve in the open-loop control system and is taken into account in calculating the manipulated variable value.

The valve characteristic is set using a cam. See the Technology Objects Synchronous Operation, Cam description of functions.

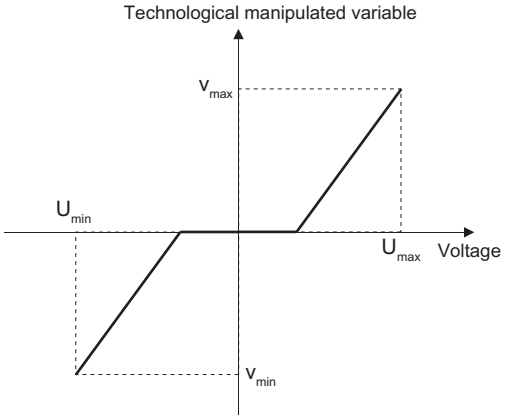


Figure 7-15 Characteristic curve setting for hydraulic axis

In the cams for the characteristic curves, the technological manipulated variable (velocity, pressure/force) is specified by means of the manipulated variable value of the final controlling element.

The valve characteristic curves are selected/switched using **_setQFAxisQCharacteristics()** or **_setQFAxisFCharacteristics()**. They can also be switched during the motion.

If several characteristic curves are interconnected with an axis, one characteristic curve must be explicitly selected with the command.

If only one valve characteristic curve is available for an axis, then a selection command is not required.

If there is no valve characteristic curve selected for the hydraulic axis under the Profiles screen form, the value in the **TypeOfAxis.MaxVelocity** configuration data element is determinative for the manipulated variable normalization. In the valve characteristic curve for the P output, the **TypeOfAxis.MaxForceCommandData** configuration data element is determinative.

Assignment of range limits

- The valve position is specified as a percentage (%).
  - Cam - master: manipulated variable -100% to +100%
  - Cam - slave: velocity or pressure -min% to +max%

  See the figure below for an example of how to assign characteristic curve parameters in SCOUT.
- The value specified in **maxSetpointVoltage** or **maxOutputVoltage** is assigned to 100%.
- A valve position of zero is mapped onto an output voltage of zero; thus, the mapping is defined explicitly.
- The interface-specific superimposition is specified as a percentage (%).
- The technology variable is specified in the application unit.

Procedure for recording the characteristic curve for a Q valve

- Enable the axis via **_enableQFAxis()** without controller
- Specify an output value between -100% and +100% via the **servosettings.additionalQOutputValue** (**servosettings.additionalFOutputValue** for P valve) system variable

  Note

  Only traverse the machine within valid range!

- Measure the actual velocity (QOutputValue and velocity can be recorded using the trace function)
- Read out the resulting technological variables
- Enter the values and the technological variables in the characteristic curve
  Each technological variable received for a specified output voltage is entered in the characteristic curve
- Assign the cam to the axis and activate it under Profiles/Valves

  Note

  With F output, the procedure applies for P valves accordingly.

Example of how to set the characteristic curve in SCOUT

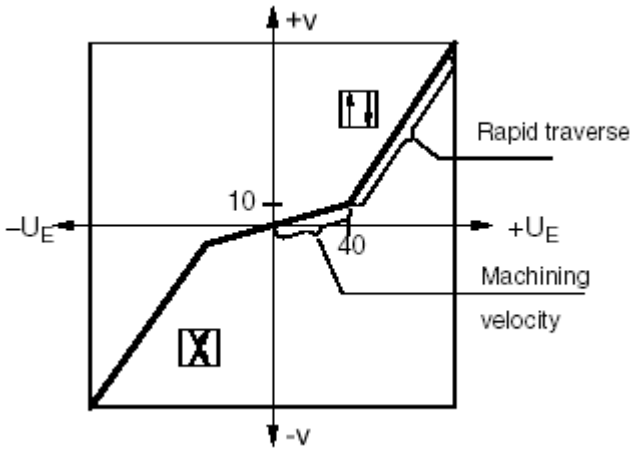


Figure 7-16 Valve characteristic curve from the manufacturer's catalog

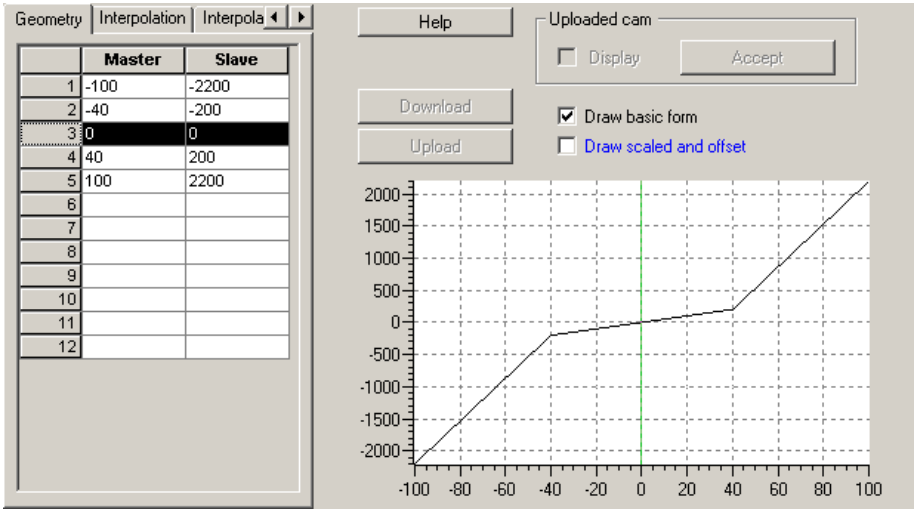


Figure 7-17 Assigning characteristic curve parameters in SCOUT

At 100% manipulated variable, a velocity of 2,200 mm/s is reached in the example (position axis).

See also

Overview of positioning axis with position control (Page 110)

7.10.7 Access to the same final controlling element from multiple axes

The axis with hydraulic functionality has its own activation/deactivation command **_enableQFAxis()/_disableQFAxis()**.

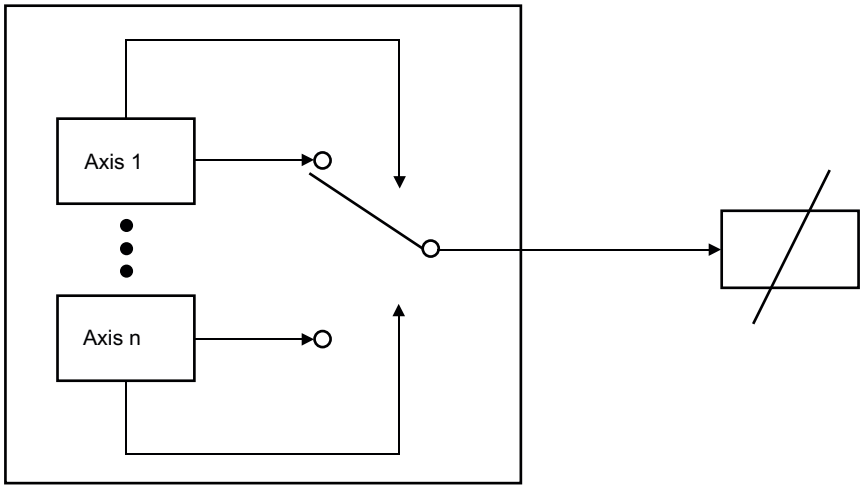


Figure 7-18 Axis with hydraulic functionality: one valve for several axes

When an axis is specified as an axis with hydraulic functionality, you can assign a final controlling element to more than one axis in the configuration. Only one axis can specify the setpoint for a final controlling element at a given time.

Application example: one variable-capacity pump for several axes. The axes are traversed sequentially. After the motion is ended, the axis is disabled (**_disableQFAxis()** with valve enable). Digital valves switch to the next axis. The next axis can be enabled.

This assignment is made during runtime using the **qOutput** or **fOutput** parameter in the **_enableQFAxis()**/**_disableQFAxis()** activation and deactivation commands and is displayed in the **actorMonitoring.qOutputState** or **actorMonitoring.fOutputState** system variable. A final controlling element that has been activated, and thus occupied, by an axis must be re-enabled by this axis before another axis can activate and occupy this final controlling element for itself.

Replacement values (stop values) can be predefined for enabling a final controlling element. This is achieved by enabling the final controlling element (command **_disableQFAxis()**, **qOutput/fOutput** = DISABLE) but without canceling the enabling signal (command **_disableQFAxis()**, **qOutputEnable/fOutputEnable** =. DO_NOT_CHANGE). The default replacement value 0% is output at the final controlling element. With **qOutputValueSetMode**/**fOutputValueSetMode** = set and **qOutputValue/fOutputValue** = 5 in the **_disableQFAxis()** command, a manipulated variable of 5% is output. The replacement value is output until a technology object occupies the final controlling element again (**_enableQFAxis()**).

The manipulated variable change is limited when the final controlling element with replacement value is applied or when changing to a different characteristic curve on the axis. These limits are specified in the **userDefaultQFAxis.maxDerivative** system variable.

The enabling signal for a final controlling element can be configured and set on multiple axes. The axis does not specifically occupy the enabling signal; rather, the enable is performed directly.

See also

Overview of positioning axis with position control (Page 110)

7.11 Travel to fixed endstop

The travel to fixed endstop functionality is not available for hydraulic axes.

See also

Travel to fixed endstop (Page 176)

7.12 Force/pressure control with hydraulic axes with Q valve only

On the hydraulic axis with Q valve only, the force/pressure control acts on the velocity setpoint (Q valve manipulated variable value) analogous to how the force/pressure control acts on the speed setpoint on the electric axis.

The handling requirements and functionality are therefore the same as for the electric axis.

The **_enableForceControlByCondition()** switchover command is active.

The **_enableForceLimitingByCondition()** switchover command is not active.

For the local RELEASE_DISABLE or OPEN_POSITION_CONTROL alarm response, the manipulated value 0 will be output using the valve characteristic value.

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

Note

Special Q valves (so-called PQ valves) also support pressure control.

See also

Overview of force/pressure control (Page 187)

7.13 Force/pressure limiting with hydraulic axes with Q valve only

With hydraulic axes with Q valve only, the force/pressure limiting control is available the same as with the electric axis.

The handling requirements and functionality are therefore the same as for the electric axis.

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

See also

Overview of force/pressure limiting (Page 196)

7.14 Force/pressure limiting with hydraulic axes with P valve

The specifications for force/pressure limiting are switched to the P valve/F output. The pressure limitation value in the commands is output on the P-valve/F output as a manipulated variable. The **_enableForceLimitingByCondition()** switchover command with condition is active. The pressure limiting value is not subject to fine interpolation if **_enableForceLimitingValue()** or **_enable...LimitingProfile()** with derivativeLimitingMode=WITHOUT_LIMITING is set.

When the event occurs, the motion is not aborted; the motion specifications continue to be output to the Q-valve. An application can be used to switch back to the motion.

No force/pressure limiting controller is in effect.

No force/pressure control is in effect.

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

7.15 Force/pressure control with hydraulic speed-controlled axes with Q valve only

Force/pressure control is available for hydraulic speed-controlled axes (V3.2 and higher). (**typeOfAxis**=REAL_QFAXIS_WITH_CLOSED_LOOP_FORCE_CONTROL)

On-the-fly switching between pressure control and speed control is possible (V4.0 and higher).

Note

Note the following points for the transitions from speed control to pressure control and vice versa:

- Apply the force/pressure control during motion (as of V4.0)
- Switch on pressure control at standstill (as of V3.2)
- Switch off pressure control at standstill (as of V3.2)
- Transition to speed control from force/pressure control (as of V4.0)
- If pressure control is active, only the RELEASE_DISABLE stop reaction is in effect. (V3.2)
  - No applying / overriding of the pressure control through motion
  - No preassigned braking ramp
- Switchover with condition not available

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

7.16 Force/pressure limiting with hydraulic speed-controlled axes with Q valve only (V4.0 and higher)

Force/pressure limiting on the hydraulic drive axis is available (V4.0 and higher).

With the hydraulic drive axis, the pressure limiting component is added to the velocity setpoint. The pressure limiting value is not subject to fine interpolation if **_enableForceLimitingValue()** or **_enable...LimitingProfile()** with derivativeLimitingMode=WITHOUT_LIMITING is set.

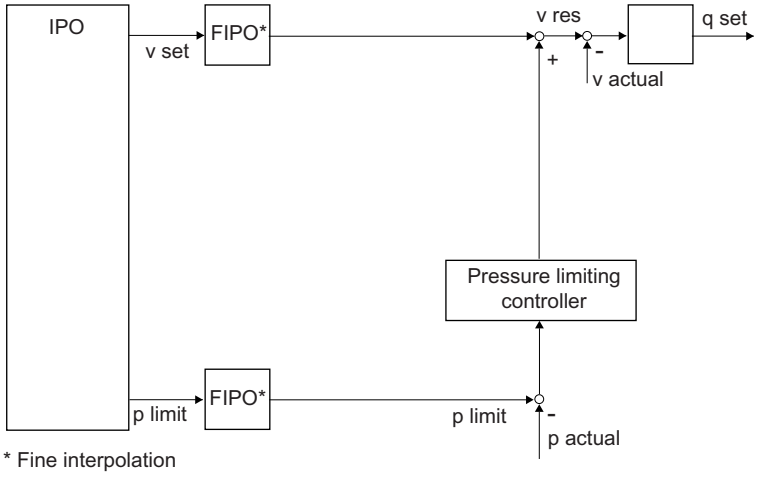


Figure 7-19 Overview of control structure for force/pressure limiting on hydraulic drive axis

Note

The *PDF documents* button can be used in the online help to locate the function charts with signal paths.

Features:

- Parallel pressure limiting
- Transition from pressure-limited to pressure-controlled
- Transition from pressure-controlled to speed-controlled (with/without pressure limiting)

The same controller is used as the pressure controller on the drive axis and as the force/pressure limiting controller.

On-the-fly switching between pressure control and speed control (open-loop and closed-loop) and vice versa is possible (as of V4.0).

Error responses / StopEmergency

If the FEEDBACK_EMERGENCY_STOP error response occurs in combination with the **_stopEmergency()** command with the STOP_WITH_COMMAND_VALUE_ZERO setting in active pressure control, the preassigned braking ramp is executed and the pressure limiting remains active. However, the pressure limiting can be deactivated via a command.

7.17 Velocity limiting with hydraulic axes

Velocity limiting on the hydraulic axis with Q valve only

This velocity limiting is available:

- On the positioning axis (V3.2 and higher)
- On the speed-controlled axis (V4.0 and higher)

Velocity limiting on the hydraulic axis with P valve and Q valve

The velocity limiting is output to the Q valve as velocity setpoints.

Part III Programming/Reference

8

8.1 Overview of commands

Note

You can also find information about the system functions in the *SIMOTION Cam Technology Package list manual*.

8.1.1 Overview of commands

Programming (execution of commands)

The axis is controlled by means of commands. These commands are used to enable and disable functions, specify and influence motions, specify data, and read out status information.

8.1.2 Commands for specification and control of motion on the axis

Table 8-1 Overview of commands for specification and control of motion on the axis

Command	Meaning	D	P	G
Enables, stop and continue commands, resets				
_enableAxis()	Enable axis	X	X	X
_enableQFAxis( )*	Enable hydraulic axis	X	X	X
_disableAxis()	Disable axis	X	X	X
_disableQFAxis( )*	Disable hydraulic axis	X	X	X
_enableForceControlByCondition()	Activating force/pressure control depending on switch-over conditions	-	X	X
_setForceCommandValue( )	Set force/pressure setpoint	-	X	X
_setQFAxisQCharacteristics*( )	Set characteristics for Q value	X	X	X
_setQFAxisPCharacteristics*( )	Set characteristics for P value	X	X	X
_stopEmergency()	Rapid stop with motion abort	X	X	X
_stop()	Stop axis motion with/without abort	X	X	X
_continue()	Resume an interrupted motion	X	X	X
_resetAxis()	Reset axis	X	X	X
_getAxisErrorNumberState()	Read out error number status	X	X	X
_resetAxisError()	Acknowledge axis error	X	X	X
_cancelAxisCommand( ), as of V4.1 SP1	Abort command with the specified CommandID	X	X	X
* Active only when the **TypeOfAxis** setting is **QFAxis**.				
Axis motions				

Command	Meaning	D	P	G
_homing()	Homing	-	X	X
_move()	Continuous turning (speed- or position-controlled)	X	X	X
_pos()	Positioning	-	X	X
_runTimeLockedVelocityProfile()	Rotational speed profile	X	X	X
_runTimeLockedPositionProfile()	Velocity profile	-	X	X
_runPositionLockedVelocityProfile()	Position-related velocity profile	-	X	X
_enableTorqueLimiting()	Switch on torque limitation	X	X	X
_disableTorqueLimiting()	Deactivate torque limitation	X	X	X
_enableAxisAdditiveTorque( )	Activate input interconnection for additive torque	X	X	X
_disableAxisAdditiveTorque( )	Deactivate input interconnection for additive torque	X	X	X
_enableAxisTorqueLimitPositive( )	Activate input interconnection B+	X	X	X
_disableAxisTorqueLimitPositive( )	Deactivate input interconnection B+	X	X	X
_enableAxisTorqueLimitNegative( )	Activate input interconnection B-	X	X	X
_disableAxisTorqueLimitNegative( )	Deactivate input interconnection B-	X	X	X
_enableVelocityLimitingValue()	Activate velocity limiting	X	X	X
_disableVelocityLimiting()	Deactivate velocity limiting	X	X	X
_enablePositionLockedForceLimitingProfile()	Activate force/pressure limiting with position-related force/pressure limiting profile	-	X	X
_enablePositionLockedVelocityLimitingProfile()	Activate velocity limiting with position-related limiting profile	-	X	X
_runTimeLockedForceProfile()	Starting the time-related force/pressure profile	-	X	X
_runPositionLockedForceProfile()	Starting the position-related force/pressure profile	-	X	X
_enableForceLimiting()	Enable force limiting	-	X	X
_disableForceLimiting()	Disable force limiting	-	X	X
_enableMovingToEndStop( )	Activate travel to fixed endstop	-	X	X
_disableMovingToEndStop()	Deactivate travel to fixed endstop	-	X	X
_enableTimeLockedVelocityLimitingProfile( )	Activate velocity limiting with time-related limiting profile	X	X	X
_enableTimeLockedForceLimitingProfile()	Activate force/pressure limiting with time-related limiting profile	-	X	X
Coordinate system				
_redefinePosition( )	Set actual value system	-	X	X
_setAndGetEncoderValue( )	Synchronize measuring systems	-	X	X
_enableMonitoringOfEncoderDifference()	Activate encoder difference monitoring	-	X	X
_disableMonitoringOfEncoderDifference()	Deactivate encoder difference monitoring	-	X	X
_getAxisUserPosition( )	Return user position for specified encoder value	-	X	X
_getAxisInternalPosition( )	Return encoder value for specified user position	-	X	X
Simulation				
_enableAxisSimulation()	Activate program simulation	X	X	X
_disableAxisSimulation()	Deactivate program simulation	X	X	X
Information functions / command buffers				
_getStateOfAxisCommand()	Reading the execution status of a motion command	X	X	X
_getMotionStateOfAxisCommand()	Reading current phase of motion	X	X	X
_bufferAxisCommandId()	Save CommandId	X	X	X

Command	Meaning	D	P	G
_removeBufferedAxisCommandId()	Remove buffered CommandId	X	X	X
_getStateOfMotionBuffer()	Read status of motion buffer	X	X	X
_resetMotionBuffer()	Reset motion buffer	X	X	X
_getProgrammedTargetPosition( )	Read the programmed absolute end position	-	X	X
_getAxisErrorNumberState()	Reading the status of a specific error on the axis	X	X	X
Data set commands				
_setAxisDataSetActive()	Activate data set	X	X	X
_setAxisDataSetParameter()	Write data set	X	X	X
_getAxisDataSetParameter( )	Read a data set	X	X	X
_setForceControlDataSetParameter()	Writing the force/pressure-specific data of a data set	-	X	X
_getForceControlDataSetParameter()	Reading the force/pressure-specific data of a data set	-	X	X
_resetAxisConfigDataBuffer( )	Clear the configuration data in the buffer without activation	X	X	X

D = drive axis

P = positioning axis

G = synchronized axis

PLCopen commands

The blocks listed below can be used in cyclic programs/tasks in SIMOTION.

They are used primarily in the LAD/FBD programming language. PLCopen blocks are available as standard functions (directly from the command library).

Table 8-2 SingleAxis functions for the axis (as of V4.0)

Function	Description
_MC_Power()	Enabling an axis
_MC_Stop()	Stopping the axes
_MC_Reset()	Resetting the axis
_MC_Home()	Homing axes
_MC_MoveAbsolute()	Absolute positioning axes
_MC_MoveRelative()	Relatively positioning axes
_MC_MoveVelocity()	Traversing axes at defined velocity (position axis travels position-controlled)
_MC_MoveAdditive()	Relative traversing of axes by a defined path additively to the remaining path
_MC_MoveSuperimposed()	Relative superimposing of a new motion on an existing motion
_MC_PositionProfile()	Traversing axis by a predefined and specified position/time profile
_MC_VelocityProfile()	Traversing axis by a predefined and specified velocity/time profile
_MC_ReadActualPosition()	Reading the actual position of axis
_MC_ReadStatus()	Reading the status of an axis
_MC_ReadAxisError()	Reading the error of an axis
_MC_ReadParameter()	Reading the axis parameter data type LREAL
_MC_ReadBoolParameter()	Reading the axis parameter data type BOOL
_MC_WriteParameter()	Writing the axis parameter data type LREAL

Function	Description
_MC_WriteBoolParameter()	Writing the axis parameter data type BOOL
Apart from the standard PLCopen functions, the following additional standard axis function is included:	
_MC_Jog()	Continuous or incremental jogging

Table 8-3 MultiAxis functions for the axis (as of V4.0)

Function	Description
_MC_CamIn()	Start camming between a master and a slave
_MC_CamOut()	End camming
_MC_GearIn()	Start gearing between a master and a slave
_MC_GearOut()	End gearing
_MC_Phasing()	Offsets the position of the slave axis relative to the master

Table 8-4 Functions for external encoder (as of V4.1 SP1)

Function	Description
_MC_Home()	Home external encoder
_MC_Power()	Enable external encoder
_MC_ReadStatus()	Read status of external encoder
_MC_ReadAxisError()	Read error of external encoder
_MC_Reset()	Reset external encoder
_MC_ReadParameter()	Read external encoder parameters of data type LREAL
_MC_ReadBoolParameter()	Read external encoder parameters of data type BOOL
_MC_ReadActualPosition()	Read actual position of external encoder

You can find additional information in the *PLCopen blocks* Function Manual and in the online help.

8.1.3 Command properties

Function parameters

Motion commands have the following function parameters:

- Information on the type of motion (**_pos()**, **_move()**, etc.) and any function parameters for specification (e.g. direction specification)
- Information regarding the response to active motions/commands (mergeMode)
- Parameters for the motion (dynamic response parameters)
- Parameters for advancing in the program (nextCommand)
- In the case of profile commands, information regarding the starting point within the profile, if required

The **dynamic response parameters** for motions and motion transitions (velocity, acceleration, and jerk) can be specified in the commands.

Additional properties

The following is applicable for profiles:

- The specified quantity corresponding to a cam-defined function is traversed
- Dynamic response parameters can be specified for travel to the start value in the cam or for travel from the end value
- If the profile is specified with a user-definable cam function, a value within the definition range of the cam must also be specified as the starting point from which the profile is to be applied
- The function in the cam is traversed to the end
- The definition range of the cam is evaluated in the user-defined time unit for time-related profiles and in the user-defined position unit for position-related profiles; the range is evaluated in the user-defined unit of the quantity to be traversed
- The direction of motion, velocity, and acceleration are calculated accordingly
- Behavior at the end of the profile can be configured
- The axis override function has no effect on the positioning profiles. The velocity override and acceleration override are included in the velocity profiles.

The status of motion commands can be seen from the **system variable** for the command; if necessary, additional information such as the braking distance and remaining distance for the positioning command is provided in the **posCommand** system variable.

Commands that contain a CommandId parameter allow the CommandId to be initialized to its defaults (value 0.0) (V3.1 and higher). This does not apply to **_getStateOf...CommandId()** commands, since initialization to defaults would not be practicable in this case. The CommandId should be explicitly specified here.

8.2 Enables, stop and continue commands, resets

8.2.1 Setting and canceling the axis enables

The commands for enabling and disabling the axis are:

- **_enableAxis(), _enableQFAxis()**
- **_disableAxis(), _disableQFAxis()**

The following enables can be set specifically for the electric axis using **_enableAxis()** / **_disableAxis()**:

- Controller enable
- Drive enable
- Power enable
- Follow-up mode (motion commands are not accepted/executed)
- Force/pressure enable

The following enables can be set specifically for the hydraulic axis using **_enableQFAxis()** / **_disableQFAxis()**:

- Controller enable
- Q output enable
- F output enable
- Follow-up mode (motion commands are not accepted/executed)
- Force/pressure enable

The axis enables for the virtual axis have exactly the same effect as for the electric axis.

Note

Drive and encoder error messages can be acknowledged, even when the error is still present. The same error is not to be signaled again.
The status of the actuator or sensor can be tested in the status indication of the corresponding system variable (**state**).

Activate/deactivate axis

- The **_enableAxis()** / **_disableAxis()** commands are used to enable/disable an axis as an electric axis where the **TypeOfAxis** is set to REAL_AXIS....
- The **_enableQFAxis()** / **_disableQFAxis()** commands are used to enable/disable an axis as a hydraulic axis where the **TypeOfAxis** is set to REAL_QFAXIS....

These commands can be issued while the axis is in motion. Canceling enables results in a technological alarm; the configured alarm response is executed.

Note

If, for example, **_disableAxis()** is called immediately after **_enableAxis()**, the commands can displace one another from the command buffer.

- With **_enableAxis()**, **_enableQFAxis()** the position controller in SIMOTION is enabled immediately, provided nothing different is set in the command parameters.
- The parameter **movingMode**=POSITION_CONTROLLED enables speed and position-controlled operation.
  With SPEED_CONTROLLED, speed-controlled operation is enabled. When activating in speed mode, command parameter **servoControlMode=**ACTIVE must be set so that the setpoint path is activated in the servo.
  In SPEED_CONTROLLED mode, the axis can be traversed if the encoder fails and there are no pending error responses (see section *Switching on the speed specification mode with _enableAxis (V4.0 and higher)*).
- As of V4.1 SP1, it is also possible to activate the hydraulic positioning axis in non-position-controlled mode using the **movingMode** = SPEED_CONTROLLED parameter with command **_enableQFAxis()**
  The closed-loop velocity-controlled hydraulic axis cannot be moved with open-loop velocity control via movingMode=SPEED_CONTROLLED.
- If, when activating the drive with **_enableAxis()**, the position controller is not enabled in SIMOTION or the axis remains in follow-up mode because user-defined functions such as motor identification, brake release, etc. are executed in the drive, settings options are available with **servoControlMode** = INACTIVE or **servoCommandToActualMode** = ACTIVE. Thus, the position controller does not position on the setpoint when **_enableAxis()** is activated.
  After the transition to the Operation or S4 drive state of the PROFIdrive state machine (displayed in the **actorMonitoring.power** = ACTIVE system variable), the position controller can then be enabled in SIMOTION with the **_enableAxis()** command and the **servoControlMode** = ACTIVE parameter setting or switched from follow-up mode with **servoCommandToActualMode** = INACTIVE.
- Axes can also be activated when the actual velocity is not equal to zero. You can use the configuration data item **decodingConfig.speedModeSetPointZero** to set the mode for getting the axis to zero velocity when the enables are set.
  - **decodingConfig.speedModeSetPointZero**=YES
    The velocity setpoint (and acceleration) of the axis are set directly to zero without a braking ramp.
    As of SCOUT V4.2, this is the standard setting when creating a new axis TO.
  - **decodingConfig.speedModeSetPointZero**=NO
    The axis is stopped via a braking ramp with the maximum dynamic values.
    See also Manipulated variable superimposition (Page 137).
    Please note that braking may be superseded by other motions which become effective immediately after the axis is activated.

Axis enable and follow-up mode

In the following tables, the behavior and displays with different configurations of **_enableAxis()** / **_enableQFAxis()** are described.

Table 8-5 Behavior and displays for axis enable with **traversing mode "position and speed controlled"**

	Follow-up mode ACTIVE (Follow-up setpoint)	**Follow-up mode INACTIVE (Do not follow-up setpoint)**
Position controller ACTIVE (**servoControlMode**=ACTIVE)	Special case	**STANDARD SETTING**
	• Traverse of the axis using setpoint superimposition is possible (see Setpoint superimposition (Page 128)) using **servosettings.additionalCommandValue** (e.g., for commissioning the position controller) • Axis is in position control (**servoMonitoring.controlState**= ACTIVE) • Motion commands CANNOT be performed (due to follow-up mode, **control**=INACTIVE) • Set position IPO is tracked • Servo set position is not tracked • Monitoring functions are not active	• Axis is in position control (**servoMonitoring.controlState**=ACTIVE) • Motion commands can be executed (**control**=ACTIVE)
Position controller INACTIVE (**servoControlMode**=INACTIVE)	Special case	Special case
	• The drive is only enabled for drive-autonomous motions and functions • Axis is NOT in position control (**~.controlState**=INACTIVE) • Motion commands CANNOT be executed • IPO set position and servo is tracked • Monitoring functions are not active	In real axes, "Do not follow up setpoint" is only effective if all the other enables have been assigned.

Table 8-6 Behavior and display for axis enable with traversing mode "**speed controlled**"

	Follow-up mode ACTIVE (Follow-up setpoint)	**Follow-up mode INACTIVE (Do not follow-up setpoint)**
Position controller ACTIVE (**servoControlMode**=ACTIVE)	Special case	**STANDARD SETTING**
	• Traversing the axis without position control using manipulated variable superimposition is possible (see Manipulated variable superimposition (Page 137) using **servosettings.additionalSetpointValue** • Axis is NOT in position control (but servo setpoint enable is active, so **~.controlState**=ACTIVE) • Motion commands CANNOT be performed (due to follow-up mode, **control**=INACTIVE) • IPO set position and servo are tracked using following error model (set and actual position are offset by following error) • Monitoring functions are not active	• Axis is NOT in position control (but servo setpoint enable is active, so **~.controlState**=ACTIVE) • Motion commands for speed-controlled procedures are possible (**control**=ACTIVE) • Set position in IPO and servo are tracked using the following error model (set and actual position are offset by the following error) • Standstill monitoring is not active (internal follow-up mode, as speed-controlled mode)
Position controller INACTIVE (**servoControlMode**=INACTIVE)	Special case	Special case
	• The drive is only enabled for drive-autonomous motions and functions • Axis is NOT in position control (**~.controlState**=INACTIVE) • Motion commands CANNOT be executed • IPO set position and servo is tracked • Monitoring functions are not active	In real axes, "Do not follow up setpoint" is only effective if all the other enables have been assigned.

Remove axis enable

In the command **_disableAxis()** or **_disableQFAxis()**, (Remove axis enable) the position controller, drive and power enable withdrawn, "Do not follow up setpoint" has no effect.

Enabling line infeed

Infeed must be enabled before activating the axis.

Infeed can be activated and deactivated, and basic diagnostics can be carried out, with the **_LineModule_control** function block (as of V4.2).

The following infeeds are supported by the **_LineModule_control** function block:

- Basic Line Modules (BLM)
- Smart Line Modules (SLM)
- Active Line Modules (ALM)

The **_LineModule_control** function block is part of the command library of the SIMOTION SCOUT engineering system. You can find the function block under **Drives > SINAMICS**.

For more detailed information about the **_LineModule_control** function block, see the SIMOTION SCOUT online help (standard function for Line Modules) or the Function Manual *Standard Function for SINAMICS S120 Line Modules*.

The **LineModule_control** function block can be used for firmware versions < V4.2. You can find the block under *Applications > Cross-sector applications > Function block for controlling Line Modules* in the *SIMOTION Utilities & Applications*.

Monitoring

The status of the current drive and power enables for an electric axis is displayed in the following system variables:

- **actormonitoring.drivestate** (drive enable)
- **actormonitoring.power** (power enable)

In a hydraulic axis, the occupation status of the Q final controlling element and F controlling element is displayed in the following system variables:

- **actorMonitoring.QOutputState** (Q output)
- **actorMonitoring.FOutputState** (F output)
  The Q valve enable is displayed in **actormonitoring.drivestate**.

If the axis is switched from follow-up mode, the **Control** system variable switches to ACTIVE. The **Control** system variable indicates whether the axis is able to generate motions.

The axis goes into follow-up mode after control startup. Enables are not mandatory in follow-up mode.

For position-controlled axes, disabling an electric drive results in automatic disabling of the position controller. The position control enable signal is ignored for speed-controlled axes. For a hydraulic axis, enabling or disabling of the Q output results in disabling of the controller.

Behavior of electric axis should a digital drive be missing or switched off

If a digital drive is missing or switched off, the **_enableAxis()** command and motion commands behave as follows:

- If a drive is missing, **_enableAxis()** is aborted with an error message
- If the drive is switched on but is not yet in cyclic operation, if commands are issued synchronously, the system waits for the command; the command is not aborted and the command job is executed even if it is an asynchronous command; the command remains active even with asynchronous programming
- The **cyclicInterface** system variable on the axis indicates whether the drive is in cyclic operation, i.e. **cyclicInterface**= ACTIVE; cyclic operation is determined using the life-sign
  Comment:
  - On analog onboard axes, **cyclic operation active** is always indicated.
  - If a drive error is present (e.g. temperature error), the system variable displays INACTIVE.

Force/pressure control

Force/pressure control can be explicitly enabled using the **forceControlMode** function parameter in the enable/disable command. This requires that the axis is at a standstill (standstill signal **motionStateData.motionState**=STANDSTILL).

When enabled, the last actual force/pressure value is taken as the force/pressure setpoint.

Force/pressure control is deactivated using the **_disableAxis()** or **_disableQFAxis()** command, or by deselecting force/pressure control in the **_enableAxis()** or **_enableQFAxis()** command.

For switching over during motion, the conditional switchover with switchover criterion with active pressure limitation is available.

System variables

Table 8-7 System variables

Variable	State	Meaning
.control	Active / Inactive	Axis active or inactive
.servomonitoring.controlstate	Active / Inactive	Position controller active or inactive
.actormonitoring.drivestate	Active / Inactive	Drive enable (drive ON) for electric axis Q valve enable for hydraulic axis
.actormonitoring.power	Active / Inactive	Power enable (pulse enable) (for electric axis only)
.actorMonitoring.qOutputState	Active / Inactive	Q output active (for axis with hydraulic functionality only)
.actorMonitoring.fOutputState	Active / Inactive	F output active (for axis with hydraulic functionality only)
.actorMonitoring.fOutputEnable	Enabled / Disabled	Status of F output enable (for axis with hydraulic functionality only)

Variable	State	Meaning
.cyclicInterface	Active / Inactive	Communication active (only for DP, this is always active for analog or onboard axes; if a drive error is present (e.g. temperature error), the system variable displays INACTIVE.)
.velocityControllerServoMonitoring.controlstate	Active / Inactive	Velocity controller on the hydraulic drive axis active or inactive

Setting enables individually

For PROFIdrive-coupled drives, the BY_STW_BIT parameter of the **_enableAxis()** and **_disableAxis()** functions provide an interface for setting and canceling of individual enables. This allows the users to implement their own state machine transitions.

Bits 0 to 6 of the ctrl word1 control word can be set and reset using the BY_STW_BIT parameter:

_enableAxis() sets the bits transferred in the parameter in the control word

_disableAxis() clears the bits transferred in the parameter in the control word

For non-PROFIdrive-coupled drives, these bits have no significance and are rejected with 'illegal command parameters'.

The statuses can be read from status word STW via the **driveData** system variable. The state display in the **actorMonitoring.driveState** and **actorMonitoring.power** system variable remains unchanged.

Entering control bits 0 to 6 directly using the **_enableAxis()** and **_disableAxis()** functions can be combined with specifying the command mode (DRIVE or POWER).

For detailed information about linking drives using PROFIdrive, refer to Coupling of digital drives (Page 56).

For information on defining the reaction to technological alarms, refer also to Section Adjustable response to RELEASE_DISABLE (Page 357).

Note

STW1 is initialized by the drive and the **actorMonitoring.driveState** and **actorMonitoring.power** system variables are generated in STW1 according to the drive status. (as of V3.2)

The status display in these system variables is thus delayed by one cycle clock relative to the control word specifications. (as of V3.2)

Enabling the speed specification mode with _enableAxis (as of V4.0)

By enabling speed specification mode (parameter **movingMode**=SPEED_CONTROLLED of **_enableAxis()**), it is possible to continue motion in the event of the failure of a selected encoder that is not involved in closed-loop control. When activating in speed mode, command parameter **servoControlMode=**ACTIVE must be set so that the setpoint path is activated in the servo.

In position-controlled operation, the axis is brought to a standstill if a selected encoder involved in the position control fails. The drive enable to be canceled can be specified (**driveControlConfig.releaseDisableMode** configuration data element).

After the axis is stopped and the error is acknowledged, the axis can be switched to activation mode with speed specification. This allows non-position-controlled axis motion in the case of an encoder failure.

Note

- Failure of an encoder involved in the closed-loop control for position-controlled positioning still causes the axis to be brought to a standstill.
- In differential position measurement, all involved encoders are in the active state.
- If a command is issued to an encoder which has an error, alarm 20005 is issued as previously. This can be the case, for example, with an encoder that has an error in the data set.
- In SIMOTION, the encoder selected for closed-loop control is specified in the data set.

See also

Command groups (Page 210)

Enabling force/pressure control depending on switchover conditions (Page 321)

Coupling of digital drives (Page 56)

Dynamic limiting functions (Page 160)

Positioning and standstill monitoring (Page 102)

8.2.2 Enabling force/pressure control depending on switchover conditions

The **_enableForceControlByCondition()** command causes a switch to force/pressure control when the switchover criterion defined in the command is satisfied.

The switchover criterion is checked in the servo. The switchover criterion can be a position, pressure, time, or digital input. The conditions are specified in the command and can be logically combined in the command in multiple stages. Reissuing the command before the conditions are met enables you to switch between conditions.

After the switchover to force/pressure control, the axis executes the function specified in the cam according to the profile specification. The rise in pressure for any necessary transitional motions, e.g. for entering and exiting the profile, can be programmed in the command.

Force/pressure, position, and time are stored at the time of switchover and are available in the system variables.

Velocity limiting can also be activated as a dependent condition in the switchover command, and a velocity limit value can be set.

The velocity limit is set directly, or the current set velocity (manipulated variable) or actual velocity can be retained as the velocity limit (as of V4.1 SP1).

A pressure/time profile or a pressure/position profile (V4.1 SP1 and higher) is enabled or the pressure value can be directly specified (as of V4.1 SP1) when the switchover occurs.

Axis with Q valve

The conditional switchover is triggered by the **_enableForceControlByCondition()** command.

Axis with P valve

Only the force limiting commands have any effect; the conditional switchover is triggered by the **_enableForceLimitingByCondition()** command.

8.2.3 Stopping motions with _stopEmergency()

The **_stopEmergency()** function stops the axis with a programmable stop mode. If a motion command is active, it is aborted and cannot be continued with a **_continue** command. The axis does not switch to follow-up mode. The axis is prevented from receiving additional motion commands. This status can be reset with **_resetAxis()** or **_disableAxis ()**. The command becomes active immediately.

The motion on the axis is stopped according to the behavior set in the command; the position control is not affected. Torque limiting, any active torque reduction (including when traveling to fixed stop) and force/pressure limiting are retained.

Active and pending motion commands on the axis are aborted and cannot be resumed.

Note

When a motion is stopped using **_stopEmergency()** and a smooth velocity profile with jerk specification is present during the axis acceleration phase, the velocity can continue to increase until the acceleration is reduced by means of the jerk which has been set.

In extreme cases, the velocity can rise still further, until it reaches the configured maximum velocity. The axis is only decelerated once acceleration has been reduced by the jerk.

With the parameter **abortAcceleration** (as of V4.2.1), you can set that a possible acceleration does not exceed the jerk limitation, but is immediately cleared.

Status

The **stopEmergencyCommand.state= ACTIVE** status is set. Motion commands on the axis are not active in this status.

Table 8-8 System variable for _stopEmergency command

Variable	State	Meaning
stopEmergencyCommand.state	ACTIVE	The **_stopEmergency()** command was triggered.
	INACTIVE	No **_stopEmergency()** command was triggered, or the command was acknowledged with **_resetAxis()** or **_disableAxis()**.

stopEmergencyCommand=ACTIVE status is canceled with **_disableAxis()** or **_resetAxis()**.

Note

The **stopEmergencyCommand**=ACTIVE status may last longer than the **_disableAxis()** command is active, so the **stopEmergencyCommand**=INACTIVE status should be checked explicitly before the enables are reset.

The axis enables are not deactivated.

The command has no effect if the axis is in follow-up mode; the status is not set.

A **_stopEmergency()** command with a higher-priority stop response overrides a lower-priority response.

The stop behavior can be set using the **stopDriveMode** function parameter.

The priority setting is defined in the stop behavior as follows:

- Specification of timing for stop motion (STOP_IN_DEFINED_TIME), irrespective of current velocity
- Stopping with maximum dynamic response values on the axis (STOP_WITH_MAXIMAL_DECELERATION)
- Stopping with dynamic response parameters (STOP_WITH_DYNAMIC_PARAMETER) This is set during configuration.
- Stop axis with setpoint zero (STOP_WITH_COMMAND_VALUE_ZERO)

StopEmergency with dynamic response parameters (as of V3.2)

The stopDriveMode=STOP_WITH_DYNAMIC_PARAMETER setting allows dynamic values to be entered and enabled directly in the **_stopEmergency()** command.

The defaults defined in the **userDefaultDynamics** system variable are used as default parameters.

See also

Stopping with preassigned braking ramp (Page 163)

8.2.4 Stopping motions with _stop()

The **_stop()** and **_continue()** commands can be used to stop and resume motions.

The **_stop** command is effective in the following technology object states: motion, motion stop without abort. It stops the entire motion or a portion of the motion of the axis using a programmed braking ramp.

The motion to be stopped can be interrupted or terminated. The portion of the motion to be stopped is specified using the **Command ID** or the type of motion.

An interrupted motion (entire motion or portion of motion) that was stopped with **_stop()**, stopMode=STOP_WITHOUT_ABORT and was not completely interpolated before the stop

command was issued can be resumed using the **_continue()** command. When a motion is resumed, the dynamic response parameters of the interrupted command, such as velocity profile, acceleration, etc., are used.

Motions interrupted by **_stop()** commands in **stopMode**=STOP_WITHOUT_ABORT can be overridden by motion commands with **mergeMode** IMMEDIATELY. The NEXT or SEQUENTIAL merge modes do not cause the motion to be executed immediately.

stopMode=STOP_AND_ABORT aborts the commands selected in the stop command. The aborted commands are returned with errors. When motion commands are aborted, the **Command Aborted** technology alarm is generated.

A command containing a position-controlled stop motion changes from ACTIVE to EXECUTED if the interpolation is completed, and the actual value is in the positioning window and the minimum dwell time has elapsed (MOTION_DONE). If the axis does not achieve this status, e.g., due to active torque limiting for deactivated following error monitoring and deactivated position monitoring (and therefore no canceling of a command due to a local alarm response), the command status remains ACTIVE.

Note

When a motion is stopped using **_stop()** and a smooth velocity profile with jerk specification is present during the axis acceleration phase, the velocity can continue to increase until the acceleration is reduced by means of the jerk which has been set.

In extreme cases, the velocity can rise still further, until it reaches the configured maximum velocity. The axis is only decelerated once acceleration has been reduced by the jerk.

With the parameter **abortAcceleration** (from V4.2.1), you can set that a possible acceleration does not exceed the jerk limitation, but is immediately cleared.

8.2.5 Stopping position-controlled axes in speed-controlled mode (V3.1 and higher)

- The **movingMode=SPEED_CONTROLLED** parameter of the **_stop()** and **_stopEmergency()** commands can be used to stop position-controlled axes in speed-controlled mode.
  The speed ramp takes effect immediately, and any pending following error is not removed. This also means that in force-controlled mode, force-limited mode, torque-limited mode, and velocity-limited mode, the axis is stopped immediately with the ramp via **_stopEmergency()** on switching over to speed-controlled mode.
- Axes can be stopped with position control in the **movingMode=POSITION_CONTROLLED** setting.
  With a drive axis, the **POSITION_CONTROLLED** setting is ignored.
- In the **movingMode=CURRENT_MODE** setting (default setting, compatibility mode), the axis is stopped in the last traversing mode set on the axis.
  In force-/pressure-controlled operation as well, the axis is stopped in the last traversing status (position-controlled or speed-controlled) set on the axis.

This enables a transition on the position-controlled axis with **_stopEmergency()** from each operating mode to speed-controlled mode and to position-controlled mode.

Comment:

Position-controlled commands cannot be resumed after the **_stop()** command is set with **movingMode= SPEED_CONTROLLED**; the motion command is aborted.

8.2.6 Resuming motions

The **_continue()** function resumes the complete motion or portion of a motion of the specified axis if it was stopped with **stop()** and STOP_WITHOUT_ABORT in the **stopMode** parameter.

The portion of the motion to be resumed is specified with the **Command ID** or by the type of motion.

When a motion is resumed, the dynamic response parameters of the interrupted command, such as velocity profile and acceleration, are used.

Resuming motions after _disableAxis() (V3.2 or higher)

With setting **TypeOfAxis.DecodingConfig.disableMotionOperation**=No, motion commands that are stopped with **_stop()** (stopMode=STOP_WITHOUT_ABORT) are not aborted by **_disableAxis()**, and may be resumed after **_enableAxis()** with **_continue()**.

The following restrictions apply:

- Only the main motion (basic motion) can be resumed.
- If motions are superimposed, continuation is only possible if the superimposed coordinate system is returned to while in the stop state.
  To do this, the **decodingConfig.transferSuperimposedPosition** configuration data element must be configured with TRANSFER_STANDSTILL.
- Relative positioning is started again, i.e. it is run through again in full after **_continue()**.

Note

_stopEmergency() is used to abort all motion commands; it is not possible to continue in this case.

8.2.7 Reset axis

The **_resetAxis()** command places the axis in a defined initial state.

- All active motions are stopped with a preassigned braking ramp, and the axis enables are retained.
- Commands in the motion buffer and commands pending at the motion buffer are deleted; synchronous commands are aborted. The **Command aborted** technology alarm is suppressed.
- Optionally, the axis can be restarted using a function parameter in the command.
- The command is executed synchronously.
- The command can also be executed asynchronously (V3.1 and higher).
- Pending errors on the axis are reset. The command is terminated with a negative acknowledgment if any errors occur that must not be acknowledged at this point.

- Any system variables changed by the program are reset to the configured values upon request. Actual values are retained.
- The homing status is retained.

Restart

When restart is activated on the technology object with the **activateRestart** parameter, the technology object performs a restart.

- An active motion on the axis is aborted without error messages or the like.
- The actual value system of the axis and, thus, the homing status is reset.
- All commands on the axis are aborted.
- All connections to other technology objects are removed and reestablished.

The **_resetAxis()** command stops any synchronous operation command that is in effect on this axis. The Synchronous Operation technology object on the synchronous axis is not reset.

Note

Setting **restartActivation** initiates a restart that is performed asynchronously. For this reason, it can take some time for the TO restart to begin.

If synchronous processing is required, **_resetAxis()** with the **ACTIVATE_RESTART** parameter must be used.

System variables

Table 8-9 System variables for the _resetAxis() command

Variable	State	Meaning
.reset	ACTIVE/INACTIVE	Reset command is active or not active
.restartActivation	ACTIVATE_RESTART	Setting the **restartActivation** variable to ACTIVATE_RESTART activates a restart of the technology object.
	NO_RESTART_ACTIVATION	Once the restart of the technology object is completed, the system resets the variable to NO_RESTART_ACTIVATION.

See also

Saving the Command ID (Page 345)

8.2.8 Resetting an axis error

The **_resetAxisError()** function resets a specified error or all errors on the axis. The command is terminated with a negative acknowledgment if any errors occur that must not be acknowledged at this point.

The command is asynchronous. The command can also be issued synchronously (V3.1 and higher). If necessary, the error is not reset until the local reaction triggered by the error has been completed.

Please also note the information in the **Motion Control Basic Functions**manual under *Display and acknowledge technological alarms*.

System variables

Table 8-10 System variables for the _resetAxisError() command

Variable	State	Meaning
.error	NO/YES	No errors/warnings are pending on the axis, or one or more errors/warnings are pending on the axis.
.errorReaction	See the **EnumAxisErrorReaction** data type as defined in the reference list for the system variables	Indicates the response that occurs based on one or more technological alarms

8.2.9 Canceling/deleting an axis command (as of V4.1 SP1)

The **_cancelAxisCommand()** function cancels the execution of the command or function with the CommandId specified in the **_cancelAxisCommand()** command and removes the command from the job list.

With **_cancelAxisCommand()**, commands such as **_homing()** with setting **homingMode**=PASSIVE_HOMING can be canceled too, as long as they are not the active motion.

The motion is cancelled with the maximum dynamic values. This also applies to superimposed motions, whereby the dynamic values from either motion cannot exceed the maximum values.

8.3 Commands for axis motions

8.3.1 Homing

The **_homing()** command homes the axis.

Various homing modes can be set by means of the **homingMode** parameter:

- Active homing (**ACTIVE_HOMING**)
  The axis is homed according to the sequence defined in the configuration.
- Setting of current position value (**DIRECT_HOMING**)
  The axis coordinate system is set to the value of the home position coordinate. The axis does not move.
- Offsets the actual position value (**DIRECT_HOMING_RELATIVE**)
  The axis coordinate system is offset by the value of the home position coordinate. The axis does not move.
- Absolute encoder adjustment
  - **homingMode=ENABLE_OFFSET_OF_ABSOLUTE_ENCODER**
    The configured value for the absolute encoder adjustment is compensated additively with the retentively stored offset. The total offset is stored in the NVRAM and is available after the control is switched off.
  - **homingMode=SET_OFFSET_OF_ABSOLUTE_ENCODER_BY_POSITION (as of V4.1 SP1)**
    The value in the *homePosition* parameter is set as the current position, and the resulting absolute encoder offset is calculated based on this. The total offset is stored retentively and is available after the control is switched off.

    Note

    Once a new project has been downloaded to the control, the stored offset is no longer available.

- Passive homing (**PASSIVE_HOMING**)
  The **_homing()** command with the PASSIVE_HOMING setting does not itself trigger an active axis motion; rather, homing occurs during the next axis motion. The axis is homed according to the sequence defined in the configuration.
  The motion command can be triggered before or after the **_homing()** command.
  The **_homing()** command is active in parallel until the axis is homed.
  Disabling the command in SIMOTION versions earlier than V3.2
  - **_resetAxis()**
  - **_disableAxis()**

  Disabling the command as of SIMOTION V3.2 with
  - **_resetAxis()**
  - **_stopEmergency()**

The sequences for active homing, **homingMode**=ACTIVE_HOMING, and the criteria for passive homing, **homingMode**=PASSIVE_HOMING, are set in the configuration.

The dynamic parameters for homing are programmable and refer to all phases of the homing procedure.

An axis has **homed** status when the axis coordinate system has been aligned with the homing signal. The status can be read out in the **positioningstate.homed** system variable.

System variables

Table 8-11 System variables for the _homing() command

Variable	State	Meaning
.userDefaultHoming	See the **StructAxisHomingDefault** data type as defined in the reference list of the system variables	User defaults for homing
.homingCommand	See the **StructAxisHomingCommand** data type as defined in the reference list of the system variables	Status of the homing sequence
.positioningstate.homed	YES/NO	Axis is/is not homed

See also

Homing (Page 84)

8.3.2 Moving

The **_move()** command can be applied to all axis types.

With the speed-controlled axis, the motion is implemented with speed specification.

With the positioning axis, the motion is implemented as velocity specification. In this case, the motion can be implemented with position-control (signal-integrated velocity) or only as a velocity specification without position control.

This selection is made using the **movingMode** parameter in the command.

- Position-controlled motion (POSITION_CONTROLLED)
- Speed-controlled motion (SPEED_CONTROLLED)

The **movingMode** parameter has no effect on the speed-controlled axis.

The **_move()** command can be applied as a superimposed command.

AS of SIMOTION V4.4, an existing acceleration or deceleration can be set to zero immediately when a motion is initiated by the **_move()** command. The existing acceleration or deceleration is not first reduced via the jerk. The **abortAcceleration** function parameter is available for this purpose.

Table 8-12 System variables for moving with _move()

Variable	State	Meaning
moveCommand.state	INACTIVE / ACTIVE_ABSOLUTE /ACTIVE_RELATIVE	Indicates that a **_move()** command is active on the axis. A distinction is made according to whether the **moveCommand.TargetVelocity** variable indicates an absolute or relative value.
moveCommand.TargetVelocity		Indicates the absolute or relative velocity.
moveCommand.relativeActual VelocityToTargetVelocity		Indicates the current velocity relative to the target velocity.

See also

Traversing the axis via velocity specifications (Page 163)

8.3.3 Positioning

- The **_pos()** motion command moves the axis to the programmed target position using an assignable velocity profile.
- The position can be specified as an absolute or relative position.
- The direction of motion can be specified for modulo axes because the target position can be attained in various directions with such axes. The various modes must be entered using the **direction** parameter (see table).
- **_pos()** commands have a special **blending** mode that links the _pos command to the previous _pos command.
- The **_pos** commands can be applied as superimposing commands.

Note

While the positioning motion is active, if the setpoint system is reset using **_redefinePosition()** or **_homing()** (DIRECT_HOMING / PASSIVE_HOMING), the motion is resumed as follows:

- For absolute positioning, the target position is approached in the newly-defined coordinate system.
- For relative motion, the relative path is traveled. In this case, offsetting the logical coordinate system has no effect.

AS of SIMOTION V4.4, an existing acceleration or deceleration can be set to zero immediately when a motion is initiated by the **_move()** command. The existing acceleration or deceleration is not first reduced via the jerk. The **abortAcceleration** function parameter is available for this purpose.

The following table provides an overview of the possible combinations of direction specification, axis type, and positioning mode.

Table 8-13 Possible specifications for direction in the _pos( ) command

	Non-modulo axis / absolute positioning	**Non-modulo axis / relative positioning**	**Modulo axis / absolute positioning**	**Modulo axis / relative positioning**
POSITIVE	Direction determined from the target position	Positive direction	Positive direction	Positive direction
NEGATIVE	Direction determined from the target position	Negative direction	Negative direction	Negative direction
BY_VALUE	Direction determined from the target position	Direction determined from the sign of the distance specification	Direction determined from the sign of the position specification	Direction determined from the sign of the distance specification
SHORTEST_WAY	Direction determined from the target position	Direction determined from the sign of the distance specification	Direction is selected to achieve shortest distance to target	Direction determined from the sign of the distance specification

System variables

Table 8-14 System variables for superimposed positioning

Variable	State	Meaning
posCommand	See the **StructAxisPosCommand** data type as defined in the reference list for system variables	Status of the positioning motion contains status, current target position, remaining distance, and target braking distance

See also

Positioning (Page 164)

Positioning with blending (Page 164)

8.3.4 Starting a time-related velocity profile

With the **_runTimeLockedVelocityProfile()** command, the axis is traversed via a user-definable time-related velocity profile.

The velocity profile is stored in a cam.

The profile is applied from a definable starting point to the end. The dynamic response parameters for any necessary transition motions, e.g. for entering the profile and exiting the profile, can be programmed on the command.

The traversing behavior at the end of the profile is set during configuration in **decodingConfig.behaviourAtTheEndOfProfile**.

For position-controlled axes, the **movingMode** parameter in the command can be used to specify whether the axis is to move with position control or at a defined velocity.

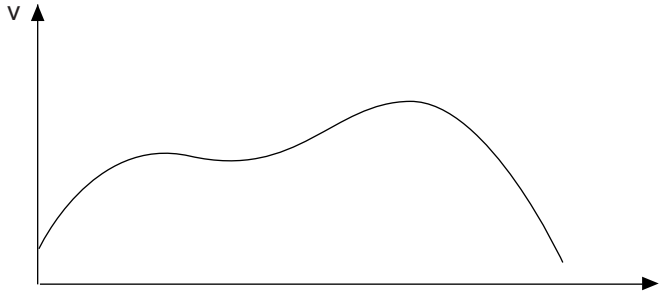


Figure 8-1 Example of a time-related velocity profile

The status and values are displayed in the elements of the **velocityTimeProfileCommand** system variable.

See also

Behavior at the end of the profile (V3.2 and higher) (Page 208)

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.5 Starting a position-related velocity profile

With the **_runPositionLockedVelocityProfile()** and **_runMotionInPositionLockedVelocityProfile()** commands, the axis is traversed via a user-definable position-related velocity profile.

The domain / the x coordinate of the cam corresponds to the absolute axis position (setpoint).

The profile is started at the current axis position. This must fall within the domain of the profile, otherwise the command is aborted and an alarm is triggered.

Note

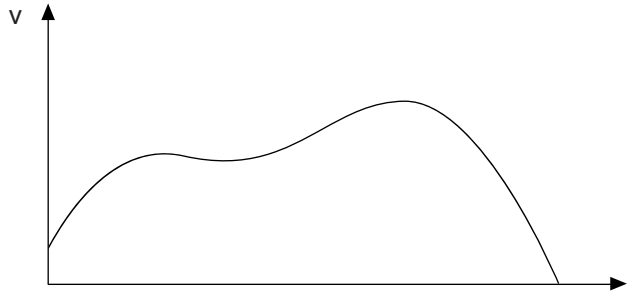
The set velocity must be a value other than zero at the starting point.

The dynamic response parameters for any necessary transition motions, e.g. for entering the profile and exiting the profile, can be programmed on the command.

The traversing behavior at the end of the profile is set during configuration in **decodingConfig.behaviourAtTheEndOfProfile**.

In addition, a tolerance for the detection of the end of the profile can be specified via the **TypeOfAxis.VelocityPositionProfile.endPositionTolerance** configuration data element. The tolerance is required if the above command has been preassigned with the WHEN_MOTION_DONE command transition.

The movingMode parameter in the command can be used to specify whether the axis is to move with position control or only at a defined velocity.



v [velocity unit of axis]
s [position unit of axis]

Figure 8-2 Example of a position-related velocity profile

The status and values are displayed in the elements of the **velocityPositionProfileCommand** system variable.

See also

Behavior at the end of the profile (V3.2 and higher) (Page 208)

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.6 Positioning with user-definable position profile

With the **_runTimeLockedPositionProfile()** and **_runMotionInPositionLockedVelocityProfile()** commands, the axis is traversed via a user-definable time-related position profile. A starting point can be specified in the cam.

The profile is applied from a definable starting point to the end. An absolute or relative position reference can be selected in the command.

The traversing behavior at the end of the profile is set during configuration in **decodingConfig.behaviourAtTheEndOfProfile**.

A transition profile is generated by the axis for discontinuous transitions. The transition profile is specified in a dynamic response parameter using the acceleration, jerk, and velocity profile parameters of the command.

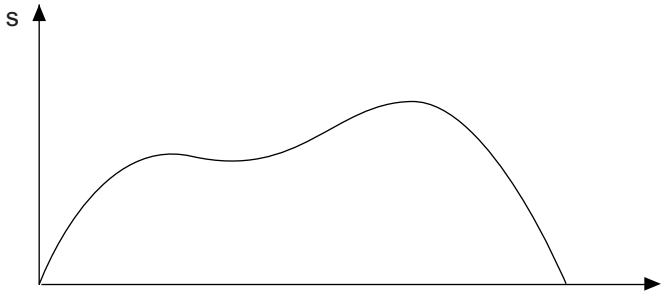


Figure 8-3 Example of a time-related position profile

The status and values are displayed in the elements of the **positionTimeProfileCommand** system variable.

See also

Behavior at the end of the profile (V3.2 and higher) (Page 208)

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.7 Enabling and disabling torque limiting

The **_enableTorqueLimiting()** command enables torque limiting in the drive, which takes effect immediately. The limiting value is specified in the command.

Active motion commands and synchronous operation relationships continue to be executed.

The limiting functions can be enabled before a motion or simultaneously with a motion and can be switched by reissuing the command.

The commands for travel to fixed endstop (**_enableMovingToEndStop()**) and torque limiting (**_enableTorqueLimiting()**) cannot be active at the same time.

The **_disableTorqueLimiting()** command cancels the torque limiting.

System variables

Table 8-15 System variables for torque limiting

Variable	State	Meaning
TorqueLimitingCommand.State	ACTIVE/INACTIVE	Indicates the torque limiting status
UserDefaultTorqueLimiting.TorqueLimit		Default for torque limiting
actualTorque		Actual torque on the axis See also **Technology data**.

See also

Overview of torque limiting via torque reduction (Page 171)

Technology data (Page 179)

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.8 Enable force/pressure limiting with position-related force/pressure limiting profile

With the **_enablePositionLockedForceLimitingProfile()** and **_enableMotionInPositionLockedForceLimitingProfile()** commands, the pressure limitation is activated with a position-related force/pressure limiting profile. The profile is set by means of a cam.

The domain of the cam to be used in the command is interpreted as a position, and the range as a force/pressure limiting value in the respective position and force/pressure units of the axis.

Behavior at the end of profile:

- Last value is still in effect
- Force/pressure limiting remains active

The profile is started at the current axis position. This must fall within the domain of the profile, otherwise the command is aborted and an alarm is triggered.

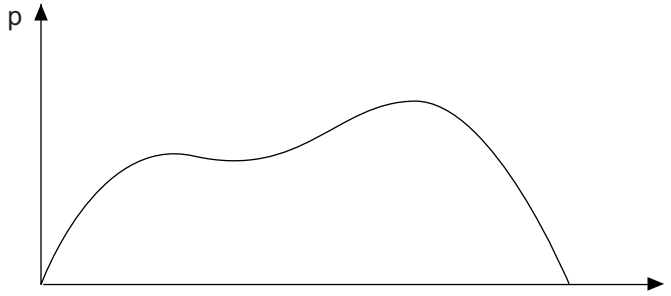


Figure 8-4 Example of a position-related force/pressure limiting profile

The status and values are displayed in the elements of the **forceLimitingCommand** system variable.

See also

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.9 Enabling force/pressure limiting with time-related motion profile

The **_enableTimeLockedForceLimitingProfile()** command activates pressure limiting with a time-related limiting profile. The profile is set by means of a cam.

The starting point can be specified in the cam.

Behavior at the end of profile:

- Last value is still in effect
- Force/pressure limiting remains active

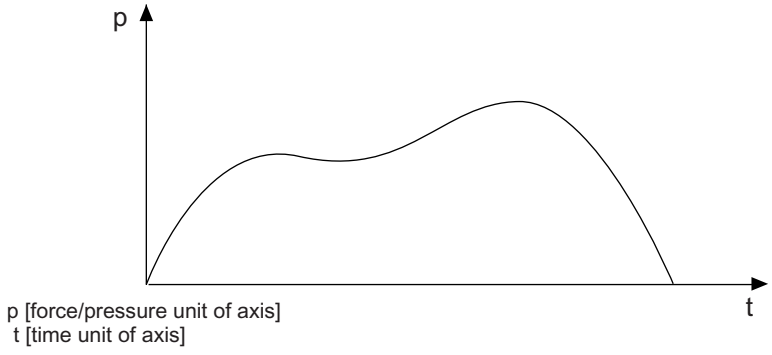


Figure 8-5 Example of a time-related limiting profile

With the axis setting **TypeOfAxis**=REAL_QFAXIS_WITH_OPEN_LOOP_FORCE_CONTROL, the profile is output as a manipulated variable on the F output.

See also

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.10 Starting the time-related force/pressure profile

The **_runTimeLockedForceProfile()** command starts the time-related force/pressure profile.

The axis executes the function specified in the cam as a force/pressure profile.

The domain of the cam to be used in the command is interpreted as time, and the range as a force/pressure in the respective time and force/pressure units of the axis.

The profile is applied from a definable starting point to the end.

Behavior at the end of profile:

- Last value is still in effect
- Force/pressure limiting remains active

The derivation of pressure for any necessary transition motions, e.g. for entering and exiting the profile, can be programmed in the command. The behavior at the end of the profile is set during axis configuration.

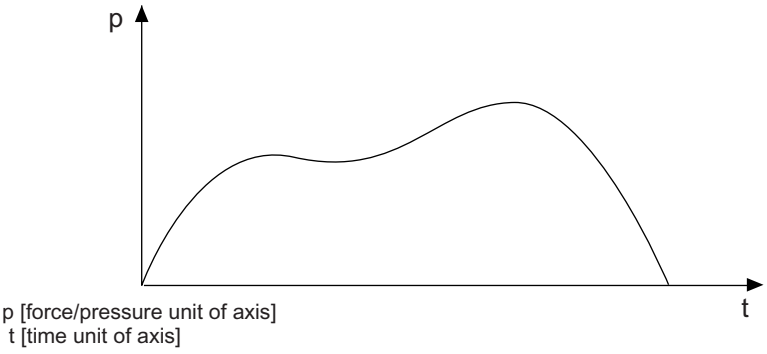


Figure 8-6 Example of a time-related force/pressure profile

The status and values are displayed in the elements of the **forceTimeProfileCommand** system variable.

If the step enabling condition is satisfied or the status is WHEN_INTERPOLATION_DONE, a pressure/time profile can switch from pressure control to position control.

See also

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.11 Starting the position-related force/pressure profile

Possible start commands:

- **_runPositionLockedForceProfile()**
- **_runMotionInPositionLockedForceProfile()**

The axis executes the function specified in the cam as a force/pressure profile.

The definition range of the cam to be used in the command is interpreted as a position and the value range as a force/pressure in the relevant position and force/pressure units of the axis.

The profile is started at the current axis position. This must fall within the definition range of the profile, otherwise the command is aborted and an alarm is triggered.

Behavior at the end of profile:

- Last value is still in effect
- Force/pressure limiting remains active

The derivation of pressure for any necessary transitions, e.g. for entering and exiting the profile, can be programmed in the command. The behavior at the end of the profile is set during axis configuration.

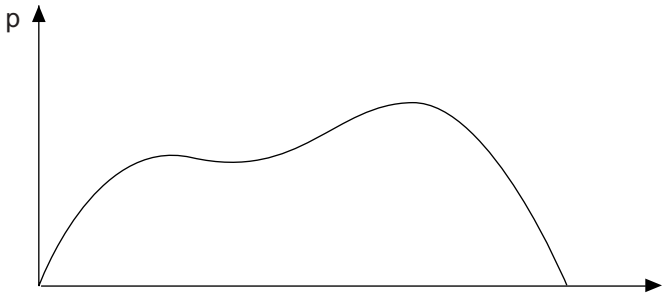


Figure 8-7 Example of a position-related force/pressure profile

The status and values are displayed in the elements of the **forcePositionProfileCommand** system variable.

See also

Traversing with user-defined motion and force/pressure profiles (Page 203)

8.3.12 Enabling/disabling force/pressure limiting

The **_enableForceLimitingValue()** command enables force/pressure limiting on the axis. The limit value is specified in the command.

Active motion commands and synchronous operation relationships continue to be executed.

The limiting functions can be enabled before a motion or simultaneously with a motion. They can be switched by reissuing the command.

Active limiting disables the following error and positioning monitoring.

The **_disableForceLimiting()** command cancels the force/pressure limiting.

With the axis setting **TypeOfAxis**=REAL_QFAXIS_WITH_OPEN_LOOP_FORCE_CONTROL, a manipulated variable is output or canceled on the F output with these commands.

System variables

Table 8-16 System variables for force/pressure limiting

Variable	State	Meaning
forceLimitingCommand.state	ACTIVE / INACTIVE	Activation status of a command for the force limitation on the axis

See also

Overview of force/pressure limiting (Page 196)

8.3.13 Enabling/disabling velocity limiting

The **_enableVelocityLimitingValue()** command enables velocity limiting on the axis. The limiting value is specified in the command.

If the velocity limit is reached, the **Anti-windup** functionality in the pressure controller is activated, i.e., the I component is stopped.

Velocity limiting can be enabled in parallel to motion commands.

In addition to a force/pressure specification, the velocity is limited in the case of dwell pressure, so that if an error occurs in the process, the velocity no longer increases, as is otherwise permitted.

In the event of an error, velocity limiting remains active except in the case of the RELEASE_DISABLE error reaction.

The **_disableVelocityLimitingValue()** command cancels velocity limiting.

System variables

Table 8-17 System variables for velocity limiting

Variable	Status	Meaning
velocityLimitingCommand.state	ACTIVE/INACTIVE	Status of the velocity limiting command on the axis

See also

Force/pressure control with velocity limiting (Page 195)

8.3.14 Enabling velocity limiting with position-related velocity limiting profile

Possible enabling commands:

- **_enablePositionLockedVelocityLimitingProfile()**
- **_enableMotionInPositionLockedVelocityLimitingProfile()**

This command activates velocity limiting with a position-related velocity profile. The profile is set by means of a cam. The velocity limit value is not implicitly deactivated at the end of the profile.

For position-related profiles, the absolute position reference of the profile is the starting point. This also applies when modulo axes are used, thus allowing the profile to remain valid beyond the end of the modulo range.

The profile is started at the current axis position. This must fall within the definition range of the profile, otherwise the command is aborted and an alarm is triggered.

Behavior at the end of profile:

- Last value is still in effect
- Force/pressure limiting remains active

See also

Force/pressure control with velocity limiting (Page 195)

8.3.15 Enabling velocity limiting with time-related velocity limiting profile

The **_enableTimeLockedVelocityLimitingProfile()** command activates velocity limiting with a time-related limiting profile. The profile is set by means of a cam. Force/pressure limiting is not implicitly disabled at the end of the profile.

The starting point can be specified in the cam.

Behavior at the end of profile:

- Last value is still in effect
- Force/pressure limiting remains active

See also

Force/pressure control with velocity limiting (Page 195)

8.3.16 Travel to fixed endstop

The **_enableMovingToEndStop( )** and **_disableMovingToEndStop( )** commands are available for the travel to fixed endstop function.

See also

Travel to fixed endstop (Page 176)

8.4 Commands for defining the coordinate system

8.4.1 Resetting the set and actual positions

The **_redefinePosition()** command is used to set/offset the axis coordinate system. Either the setpoint or actual value is specified. The value that is not specified is aligned to keep the following error constant. The servo values (actual values, setpoints) are not modified.

The **_redefinePosition** command is effective in the following technology object states: Active, Motion, and Motion stop without abort.

In the case of a virtual axis, the velocity and acceleration can also be set.

The setpoint and actual values of an axis can be changed using the **_redefinePosition()** command. The position value is specified as an absolute value or as a relative position offset. The behavior can be specified by means of the **RedefineSpecification** parameter:

- **COMMAND_VALUE** setting: The setpoint of the axis position is modified.
- **ACTUAL_VALUE** setting: The actual value of the axis position is modified.
- **COMMAND_VALUE_BASIC_MOTION** setting: Like COMMAND_VALUE, but with reference to the basic coordinate system
- **COMMAND_VALUE_SUPERIMPOSED_MOTION** setting: Like COMMAND_VALUE, but with reference to the superimposed coordinate system

The other value in each case is aligned.

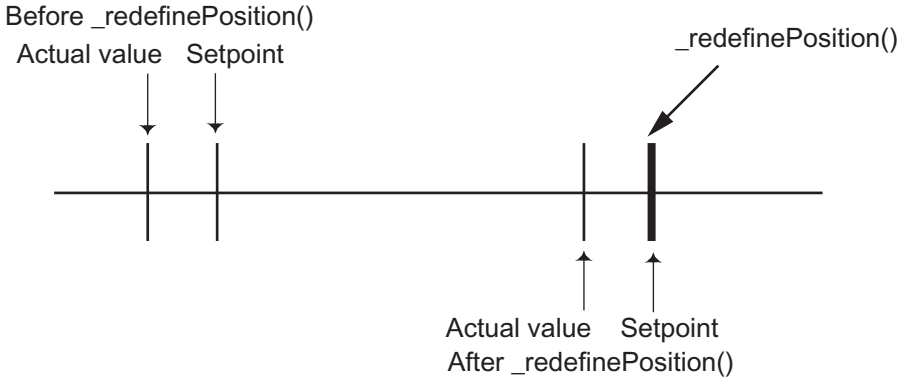


Figure 8-8 Absolute offset of axis position relative to setpoint

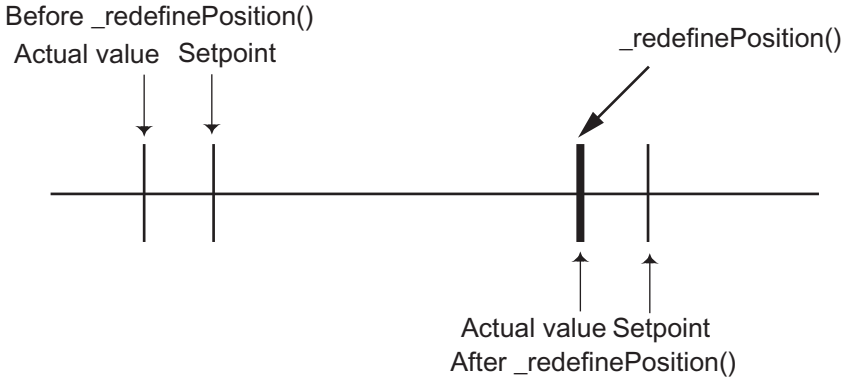


Figure 8-9 Absolute offset of axis position relative to actual value

If the set positions and/or actual positions of several axes must be re-set concurrently, this can be realized as follows:

- Programming of the **_redefinePosition()** commands for the various axes within the IPO-synchronous user task
  Requirement: For the associated axes, the appropriate IPO cycle clock must be set for this IPO-synchronous user task.
- Programming of the **_redefinePosition()** commands within a block for the synchronous processing
  (See **Motion Control Base Functions** manual, *Synchronous Start* section).

Basic coordinate system or superimposed coordinate system (as of V3.1)

_redefinePosition() can act on the basic coordinate system or the superimposed coordinate system. The behavior can be specified by means of the **RedefineSpecification** parameter.

See also

Superimposed motion (Page 169)

8.5 Simulation commands

8.5.1 Enabling/disabling program simulation

The **_enableAxisSimulation()** command sets the axis to program simulation. If the axis is in motion, it is stopped using the preassigned braking ramp. As for a virtual axis, the actual values are taken from the setpoints. The axis enables and setpoint generation remain active, as do the active motion commands.

While simulation is active, the command variable is calculated and the actual value of the interpolator is set to the setpoint of the interpolator, irrespective of the real actual value of the axis. The real actual value of the axis is not affected by the program simulation.

If the axis is operated in simulation mode, a program with command variable calculation can be executed; in this case, motions are not executed by the axis, but are only simulated.

Terminate simulation.

The **_disableAxisSimulation()** command resets the axis out of program simulation. The real actual values then continue to be accepted. All other states remain unchanged. Any existing following error is removed immediately.

The following examples describe how to avoid the following error being removed immediately.

Example: Travel onwards from the actual position of axis

1. Switch the module off and on again.
2. Deactivate the simulation.

Before deactivation, the setpoint has been set to the actual value.

Example: Travel onwards from the new setpoint

1. Determine the different between the setpoint and actual value.
2. Set the setpoint to the actual value using **_redefinePosition()**.
3. Deactivate the simulation.
4. Travel the difference between the setpoint and actual value.

The **_resetAxis()** command also terminates the program simulation of the axis.

Effectiveness and status

The **_enableAxisSimulation()** and **_disableAxisSimulation()** commands are effective in the following TO states: Inactive, Active, Motion, and Motion stop without abort.

The current simulation mode can be scanned by means of the **simulation** system variable (program simulation).

Note

The outputting of setpoint values may be blocked by the simulation.

Axis errors, such as "Drive not connected" or "Encoder fault" cannot be avoided in this way; the axis must be functional same as a non-simulated axis.

System variables

Table 8-18 System variables for program simulation

Variable	State	Meaning
Simulation	ACTIVE/INACTIVE	Simulation status

See also

Setting as a real axis without drive (axis simulation) (Page 60)

8.6 Information functions / command buffers

8.6.1 Overview of information functions/command buffer

The information commands on the axis can be used to read out the command and motion status of axis commands. It is possible to store the command status for longer than the period during which the command is in effect on the axis. The commands are identified with the **CommandId**. To scan the status, you need to specify the axis as well as the **CommandId** assigned to the command.

8.6.2 Reading the execution status of a motion command

The **_getStateOfAxisCommand()** command returns the execution status of a motion command. The status can be scanned while the command is being executed. Upon completion, the **CommandId** is deleted.

If a **CommandId** scan is required beyond this time period, it can be stored in a buffer and then cleared at any time, see **_bufferAxisCommandId** and **_removeBufferedAxisCommandId**.

The following statuses are reported:

- Command is being executed
- **CommandId** not known or command already completed
- Command is decoded, execution has not yet started
- Command is decoded, waiting for a synchronous start
- Execution of the command has been completed
- Execution of the command has been aborted

The "Execution of the command has been completed" and "Execution of the command has been aborted" statuses are only returned when the command status is stored. The **_bufferAxisCommandId( )** command can be used to store the **CommandId** and command status in the system beyond the end of the command.

For the "Execution of the command has been aborted" status (ABORTED), the reason for aborting is also given (V3.2 and higher)

System variables

None

See also

Saving the Command ID (Page 345)

8.6.3 Reading current phase of motion

The **_getMotionStateOfAxisCommand()** command returns the current phase of the motion.

The following statuses are reported:

- **NOT_EXISTENT**
  CommandId not known or command already completed
  See also Saving the Command ID (Page 345)
- **BUFFERED**
  Command is in the command queue
- **IN_EXECUTION**
  Command is being executed
- **IN_ACCELERATION**
  The motion generated by the command is in the acceleration phase
- **IN_CONSTANT_MOTION**
  The motion generated by the command is in the constant motion phase
- **IN_DECELERATION**
  The motion generated by the command is in the deceleration phase
- **AXIS_HOMED**
  Axis synchronized
- **INTERPOLATION_DONE**
  The setpoint interpolation of the command is completed
- **SYNCHRONIZING**
  Synchronizing
- **DESYNCHRONIZING**
  Desynchronizing
- **SYNCHRONIZED**
  Synchronous operation
- **MODIFICATION_ACTIVE**
  Compensating movement is active with scaling or offset in synchronous operation
- **EXECUTED**
  Execution of the command has been completed
  See also Saving the Command ID (Page 345)
- **ABORTED**
  Execution of the command has been aborted

System variables

None

See also

Saving the Command ID (Page 345)

8.6.4 Saving the Command ID

The **_bufferAxisCommandId()** command causes the command status and the **CommandId** to be stored beyond the command execution time. This permits a command status to be queried even if the command has already been completed.

In V3.2 and higher, the command includes an additional parameter that can suppress deletion of the **CommandId** on resetting.

Note

The maximum number of **CommandIds** with command status that can be stored is specified in the **decodingConfig.numberOfMaxBufferedComandId** configuration data element. This must be managed by the user program. If the buffer overflows, the command will be rejected with a return value.

8.6.5 Canceling saving of Command ID

The **_removeBufferedAxisCommandId()** command clears the specified **CommandId** and the command status from the buffer. Alternately, all stored **CommandIds** can be deleted.

8.6.6 Reading the status of a specific error on the axis

The **_getAxisErrorNumberState()** command is used to read the status of a specific error on the axis.

8.6.7 Reading out pending alarms (V4.0 and higher)

The error status, alarm number, and alarm parameters of up to eight pending alarms can be read out with the **_getAxisErrorState()** command.

8.6.8 Reading the status of the motion buffer on the axis

The information command **_getStateOfMotionBuffer()** on the **MotionBuffer** can be used to manage the buffer on the axis. This can be used to check whether the axis is ready to accept motion commands before those commands are issued. All **sequentially** effective commands are stored in the motion buffer.

The **_getStateOfMotionBuffer()** command returns the status of the motion buffer on the axis: EMPTY, FULL, or WRITEABLE.

8.6.9 Clearing the motion buffer on the axis

The information command and the reset command on the motion buffer can be used to manage the motion buffer on the axis. This can be used to check whether the axis is ready to accept motion commands before those commands are issued.

The **_resetMotionBuffer()** command deletes all commands in the motion buffer and the commands pending for the motion buffer. The current command is not deleted. The **_resetMotionBuffer()** command is executed synchronously. The command is active in all axis states.

8.6.10 Activating data sets

Data set commands can be used to read, write, and activate data sets on the axis.

The **_setAxisDataSetActive()** command sets the data set specified in the function parameter to active.

System variables

Table 8-19 System variables for activating data sets

Variable	State/type	Meaning
dataSetMonitoring	See the **StructDataSetMonitoring** data type as defined in the reference list of the system variables	Information for data set changeover • State • Requested data set • Current data set

Note

If the active measuring system is switched via a data set changeover, the **_setAndGetEncoderValue()** system function should be used to synchronize the two measuring systems before the switchover takes place. This will prevent unwanted compensating motions of the position controller if differences in position are identified.

The data sets must have the same control loop structure and must not contain a change that requires a restart. This means that when DS_1 is with DSC, DS_2 must also be with DSC.

Settings that require a restart cannot be activated via a data set changeover.

See also

Data sets (Page 200)

8.6.11 Writing a data set

The **_setAxisDataSetParameter()** command can be used to overwrite an inactive data set on the axis. The data set number is specified in the command.

See also

Data set overview (Page 200)

8.6.12 Reading a data set

The **_getAxisDataSetParameter()** command can be used to read any data set on the axis. The data set number is specified in the command.

The command reads the data that can be written with the **_setAxisDataSetParameter()** command.

See also

Data set overview (Page 200)

8.6.13 Writing the force/pressure-specific data of a data set

The **_setForceControlDataSetParameter()** command can be used to overwrite the force/pressure-related data of an inactive data set. The data set number is specified in the command.

The following force-/pressure-related data is contained in the data set:

- Monitoring of the control deviation of the force controller
- Force controller data
  Force controller setting
  Limitations of the manipulated variable of the force controller
  PID controller setting
  Controller type
  Effective direction of the manipulated variable (can be switched with V4.2 and higher)
  Inversion of the force controller manipulated variable

Detailed information about the parameters can be found in the TP Cam System Functions Reference List.

Note

The force/pressure-specific data of the data set cannot be written on the speed-controlled axis.

8.6.14 Reading the force/pressure-specific data of a data set

The **_getForceControlDataSetParameter()** command can be used to read the force/pressure-related data in the data set. The data set number is specified in the command.

The command reads the data that can be written with the **_setForceControlDataSetParameter()** command.

Note

The force-/pressure-specific data of the data set cannot be read on the drive axis.

8.6.15 Writing the data of the data set (hydraulic functionality only)

The **_setQFAxisDataSetParameter()** command can be used to overwrite the specific parameters for the axis with hydraulic functionality in an inactive data set. The data set number is specified in the command.

The following specific parameters of the axis with hydraulic functionality are contained in the data set:

- Inversion of F output
- Dynamic response adaptation to the axis with hydraulic functionality
- Data for the speed controller on the drive axis with hydraulic functionality

Detailed information about the parameters can be found in the TP Cam System Functions Reference List.

Note

The specific data for the axis with hydraulic functionality in the data set cannot be written on the drive axis.

8.6.16 Reading the force/pressure-specific data of the data set (hydraulic functionality only)

The **_getQFAxisDataSetParameter()** command can be used to read the specific parameters for the axis with hydraulic functionality in the data set. The data set number is specified in the command.

The command reads the data that can be written with the **_setQFAxisDataSetParameter()** command.

Note

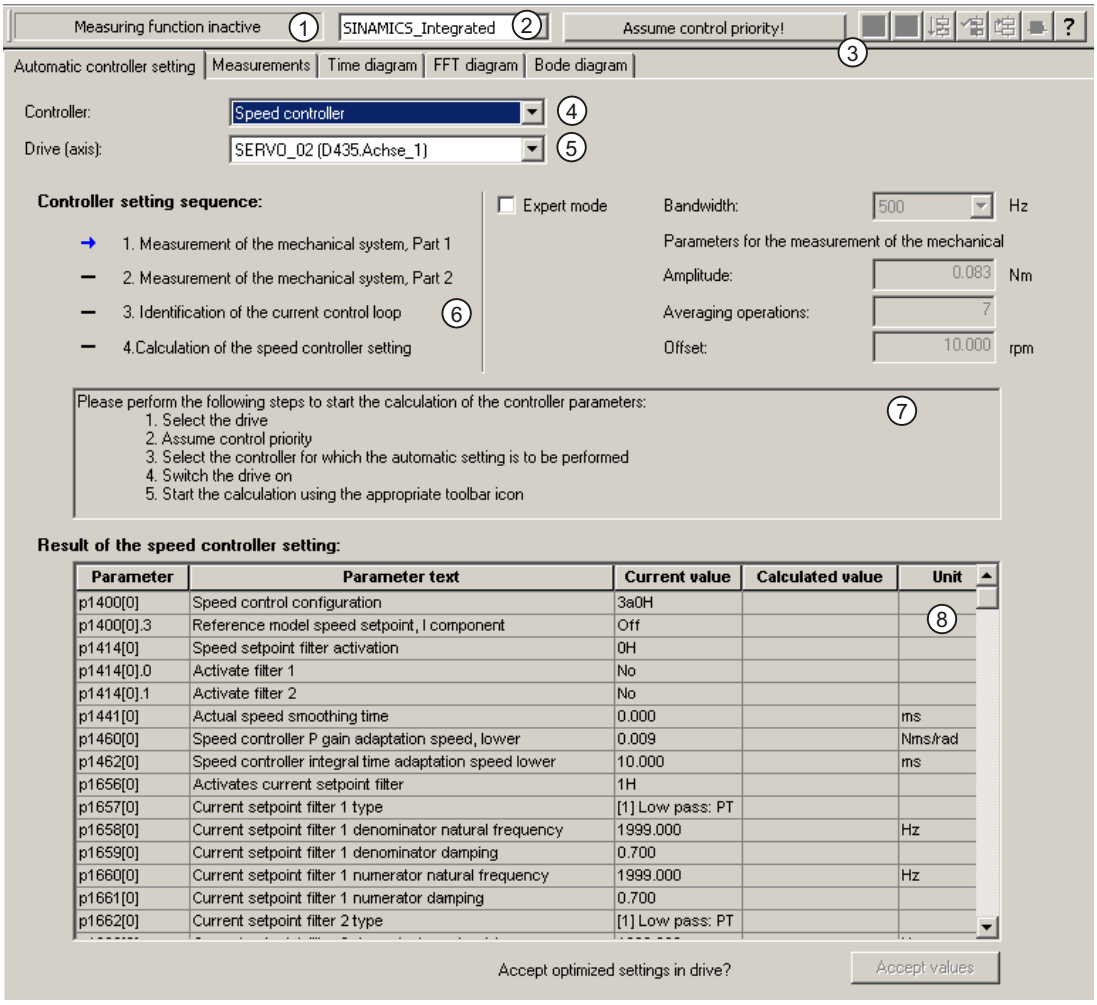
The specific data for the axis with hydraulic functionality in the data set cannot be read on the speed-controlled axis.

8.6.17 Command for calculating a braking distance

With the **_getAxisStoppingData()** command, the system calculates the braking distance as a function of a specified actual velocity, actual acceleration, motion profile, and dynamic response parameters.

8.7 Assigning automatic controller optimization

Here, you operate the automatic controller optimization. To perform the automatic controller setting, you must have an ONLINE connection to the drive device of the relevant drive.



Parameter	Parameter text	Current value	Calculated value	Unit
p1400[0]	Speed control configuration	3a0H		
p1400[0].3	Reference model speed setpoint, I component	Off		
p1414[0]	Speed setpoint filter activation	0H		
p1414[0].0	Activate filter 1	No		
p1414[0].1	Activate filter 2	No		
p1441[0]	Actual speed smoothing time	0.000		ms
p1460[0]	Speed controller P gain adaptation speed, lower	0.009		Nms/rad
p1462[0]	Speed controller integral time adaptation speed lower	10.000		ms
p1656[0]	Activates current setpoint filter	1H		
p1657[0]	Current setpoint filter 1 type	[1] Low pass: PT		
p1658[0]	Current setpoint filter 1 denominator natural frequency	1999.000		Hz
p1659[0]	Current setpoint filter 1 denominator damping	0.700		
p1660[0]	Current setpoint filter 1 numerator natural frequency	1999.000		Hz
p1661[0]	Current setpoint filter 1 numerator damping	0.700		
p1662[0]	Current setpoint filter 2 type	[1] Low pass: PT		

1 Measuring functions status display
2 Drive unit
3 Operator buttons
4 Controller selection
5 Drive selection
6 Progress message box
7 Description of current step
8 Result display

Figure 8-10 Automatic optimization - Example: speed controller

The following operator input options and displays are available:

Field/Button		Meaning/Instruction
Measuring functions status display		This area shows whether or not the drive is active in the step. The following information is displayed: • Measuring function active Drive/axis is moving • Measuring function inactive Drive/axis is not moving
Drive unit		This combo box offers all the SINAMICS drive units in the open project that contain drives operated in the "SERVO" closed-loop control type and that the motor measuring system uses as encoders. Selection is possible provided the master control has not yet been assumed.
Assume control priority/ **Give up control priority**		This allows you to assume master control in order to perform automatic controller setting. Master control can be released only once the drive has been switched off.
Operator buttons		
		**Drive ON** This button can be pressed if master control has already been assumed. This enables the infeed and assigns the drive enable. This is not the same as a POWER ON of the hardware. Afterwards, the controller setting can be started. To cancel and then restart the function, perform the following operating input sequence: Switch off drive -> Switch on drive
		**Drive OFF** You can use this button to reset the functions initiated by *Drive ON*. This enables the infeed and inhibits the drive enable. This is not the same as a POWER OFF of the hardware. If it is detected that calculated parameters are present (the last step has been completed) that have not yet been transferred to the drive/axis, a confirmation prompt will be issued, asking whether the controller setting is now complete and, as a result, the calculated parameters should be discarded.
		**Automatic execution** Automatically performs all steps of the controller setting starting with the current selection on the step display. This button will be deactivated once the last possible step has been performed.
		**Execute step** The single step of the controller optimization selected on the step display will be performed and the step display then incremented by one step. This button will be deactivated once the last possible step has been performed.
		**Step back** Sets the step display back one step. If the step display is at step 1, the button is deactivated.
		**Cancel step** Cancels the execution of the current step; the step display remains at the canceled step.
		**Help** Opens the help for this screen form.

Field/Button		Meaning/Instruction
Controller selection		In **Controller selection**, you can select which controller to set. The view on the screen form will then show the appropriate controller. The combo box contains the following selection options: • "Speed controller" (always available) • "Position controller (DSC)" (offered only if SCOUT is installed on the PC). Before the setting is applied, a few call conditions are checked and any unfulfilled conditions are displayed. The selection can be changed at any time, provided no step is currently being executed.
Drive selection		In **Drive selection**, a drive corresponding to the setting in the **Drive unit** field can be selected**.** The following rules apply for the "Drive" input field: • The field is labeled "Drive". If SCOUT is installed on the PC, it is called "Drive (axis)". • If SCOUT is installed and the drive is connected to at least one axis, the connected axis is appended to the drive name in brackets, in the form "(<device>.<axis name>)". If the drive is to be connected to multiple axes, "..." is appended to the first axis name. • The tool tip for the Drive field contains all names in the syntax shown above for the axes connected with the drive. The field can only be operated provided the master control has not yet been assumed. As soon as the "Drive selection" field registers a change, the parameters of the newly selected drive are registered in the "Current value" column in the result display for a cyclic update.
Progress message box		The step control presents the status overview of the steps of the "automatic controller setting" on the left.
		The step has been executed.
		Selection of the step that will be executed next or is currently being executed.
		Step has not yet been executed.
	Expert mode	Checkbox If the checkbox for **Expert mode** is selected, the **Amplitude**, **Bandwidth**, **Averaging operations**, and **Offset** input fields are displayed. The user can enter special settings for the next optimization step in these fields. The checkbox is deselected as the default setting. If the checkbox is deactivated, the parameter settings determined by the optimization tool are used for the measurement.
Kv factor (controller = position controller)		As a result of step 2 of the position controller optimization ("Calculation of the position controller setting"), the determined **Kv factor** is indicated. Provided no new value has been calculated, the display shows "–". This is also the standard display when the screen form starts.
Select axes and data sets button "..." (controller = position controller)		This button can be used to call a dialog that shows the possible axes (in accordance with the relevant requirements and supplementary conditions) and their axis data sets in which the Kv factor can be accepted.

Field/Button	Meaning/Instruction
Result display	This table displays the parameters whose values were determined during the controller optimization. The **Current value** column displays the current value of the parameter read out cyclically from the target device. The **Calculated value** column displays the value of the parameter determined from the controller setting. Provided no new value has been calculated, the "–" character is displayed instead of the value. This is also the standard display when the screen form starts.
Accept (controller = speed controller)	The **Accept** button is active only when step 4 of the speed controller optimization has been completed successfully. The calculated values are applied to the drive.
Accept (controller = position controller)	The **Accept** button is active only under the following conditions: • Step 2 for the position controller optimization must have completed successfully. • An online connection to the associated SIMOTION device must exist. • The axis is online-consistent. • The axis data set table contains at least one data set. The calculated values are applied to the drive.

Note

Emergency cancelation of automatic setting with <spacebar>

The following actions are performed:

- The step currently being executed is canceled
- The drive is disabled

See also

Overview of automatic controller setting (as of V4.1 SP1) (Page 230)

8.8 Technological alarms

8.8.1 Alarm reactions

The reactions of the axis to **technological alarms** (local reaction) are implemented by the system.

The local reactions are preset on an alarm-specific basis or they can be set in the alarm configuration.

The following local reactions on the axis are available:

- **NONE**
  - No reaction
  - The responses can be specified by the user in the TechnologicalFaultTask.
- **DECODE_STOP**
  - Command processing aborted
  - The current motion (thus, for example, also the MotionIn command) and commands in the motion buffer are still executed.
  - New motion commands are rejected.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.
- **END_OF_MOTION_STOP**
  - Abort at the end of the command causing the error
  - Stop of the motion at the end of the command
  - The current motion command (thus also the MotionIn command) is still executed.
  - New motion commands are rejected.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.
- **MOTION_STOP**
  - Controlled motion stop with programmed ramp values.
  - The current motion command (thus also the MotionIn command) is stopped.
  - New motion commands are rejected.
  - Motion commands are not canceled and are resumed when the error reaction is acknowledged.
  - No more new motion commands are accepted.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.
- **MOTION_EMERGENCY_STOP**
  - Controlled motion stop with maximum ramp values/limits for the axis.
  - The current motion command (thus also the MotionIn command) is stopped.
  - New motion commands are rejected.
  - Motion commands are not canceled and are resumed when the error reaction is acknowledged.
  - No more new motion commands are accepted.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.

- **MOTION_EMERGENCY_ABORT**
  - Controlled motion stop with maximum axis dynamic values (starting from the current setpoint).
  - Active commands (IPO) are aborted and signaled back to the user program with an error code.
  - The drive remains active.
  - The position controller remains active.
  - The IPO remains active.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.
  - After the error is acknowledged with **_resetError()**, the axis can travel again. (provided that the cause of the error no longer exists)
- **FEEDBACK_EMERGENCY_STOP**
  - Motion stop with preassigned braking ramp.
    See Stopping with preassigned braking ramp (Page 163).
  - Manipulated variable superimposition not effective.
  - Active commands (IPO) are aborted and signaled back to the user program with an error code.
  - The drive remains active.
  - The position controller remains active.
  - The IPO goes into follow-up mode, preassigned braking ramp servo (time specification in the servo for preassigned braking ramp)
    IPO and servo are then active again.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.
  - After the error is acknowledged with **_resetError()** and the axis is reset with **_resetAxis()**, the axis can be traversed further (provided the cause of the problem no longer exists).

- **OPEN_POSITION_CONTROL**
  - Motion stop with speed setpoint zero and abort
  - The position control loop is decoupled from the setpoint circuit.
  - Active commands (IPO) are aborted and signaled back to the user program with an error code.
  - The position controller enable is canceled.
  - The control unit transmits the setpoint to the drive.
  - The drive stops (because of setpoint 0).
  - The drive remains active.
  - The Ipo goes into follow-up mode.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.
  - After the error is acknowledged with **_resetError()** and the drive and position controller are activated with **_enableAxis()**, the axis can travel again (provided that the cause of the problem no longer exists).
- **RELEASE_DISABLE**
  - Motion disable with controller disable and abort of all commands
  - The drive enables are canceled.
    See also Coupling of digital drives (Page 56) and Adjustable response to RELEASE_DISABLE (Page 357).
  - The controller enables are canceled.
  - Active commands (IPO) are aborted and signaled back to the user program with an error code.
  - Position controller and Ipo go into follow-up mode.
  - Further reactions can be specified by the user in the TechnologicalFaultTask.
  - After the error is acknowledged with **_resetError()** and the drive and position controller are activated with **_enableAxis()**, the axis can travel again (provided that the cause of the problem no longer exists).

  Note

  Active commands are commands in the IPO or commands that are being executed; these commands are no longer in the motion buffer.

  When you acknowledge an error with **_resetAxisError()**, you must delete any commands in the motion buffer with **_resetMotionBuffer()**.

  The preset responses of the technology alarms can be overwritten by higher-priority stop responses.

  Global alarm reactions can be set on the technology object (execution system, FaultTasks).

See also

Jerk limitation for local stop response (V3.2 or higher) (Page 168)

8.8.2 Adjustable response to RELEASE_DISABLE

The drive response to the RELEASE_DISABLE alarm response can be set (with a digital drive link).

See Coupling of digital drives (Page 56).

See also

Alarm reactions (Page 353)

8.8.3 Toleration of the failure of an encoder not involved in closed-loop control (V4.0 and higher)

If an encoder that is not involved in closed-loop control fails, the system should neither stop the axis nor disable the drive.

In the following cases, an encoder is not involved in closed-loop control:

- If the encoder is not selected.
  That is, one of the 8 possible encoders of the axis is configured and active (running) but it is not the selected encoder at the moment.
- If the encoder is selected but not involved in closed-loop control.

This behavior is set using the
typeOfAxis.numberOfEncoders.Encoder_1.sensorControlConfig.tolerateSensorDefect
configuration data element.

As of SINAMICS V2.6.2, the failure of an encoder not involved in closed-loop control is indicated by alarm 20015.

Part IV External Encoder - Description

9

9.1 External encoder overview

The External Encoder technology object ("ExternalEncoder") records a position and provides the position to the control. The determined position can be evaluated in the user program.

The reference of the encoder position to a defined position is produced by parameterization of the mechanical properties and a homing procedure.

The position is stated independently of the encoder type and depending on the selected unit system:

- **Linear unit system**
  The position is given as a length dimension, e.g. millimeters (mm).
- **Rotary unit system**
  The position is given as an angle dimension, e.g. degrees (°).

Information from the External Encoder technology object can be used as input variables for other technology objects, such as the Synchronous Operation technology object, Output Cam technology object, or Measuring Input technology object, or to display the actual value of an external motion.

The External Encoder technology object can be used with

- Incremental encoders
- Absolute encoders
- Resolvers
- Encoders on direct value/analog value
- Encoders on count value
- ...

An external encoder can be used only on one position-related encoder.

The external encoder is referred to by the data type **externalEncoderType** in reference lists and when programming.

From V3.2 onwards, the External Encoder technology object is included in the Cam and Cam_ext technology packages.

Interconnections

The External Encoder technology object can be interconnected with the following technology objects:

- **Synchronous Operation technology object**
  The External Encoder technology object is used to provide a master value.
- **TO output cam**
  The External Encoder technology object is used to provide a reference value.

- **TO cam track**
  The External Encoder technology object is used to provide a reference value.
- **TO measuring input**
  The External Encoder technology object is used to provide a reference value.

Programming commands/functions for the External Encoder technology object

The MCC and ST programming languages are available for programming external encoders.

See the *SIMOTION MCC* Programming Manual or the *MCC programming language* in the online help.

External Encoder Fundamentals 10

10.1 Actual values for the external encoder technology object

Each position actual value possible on the technology object type can be used as the actual value of the external encoder technology object.

These are, for example:

- Actual value (encoder1 or encoder2) from a PROFIdrive axis telegram
  - Encoder from a PROFIdrive axis telegram to a SINAMICS drive object
    With SINAMICS it is possible to configure multiple encoders, from which a maximum of two encoder values can be transmitted in each axis telegram.

    Note

    An encoder in a PROFIdrive axis telegram can only be used for an external encoder TO or for the encoder of a TO axis in cyclical operation if a TO axis is also created on the PROFIdrive telegram and is not deactivated.

  - Encoder from a PROFIdrive axis telegram to ADI4 (at least one axis must be created on the ADI4)
  - Encoder from a PROFIdrive axis telegram to IM174 (at least one axis must be created on the IM174)
  - Encoder from a PROFIdrive axis telegram to a non-SINAMICS drive
- Actual value from a PROFIdrive encoder telegram
  - Encoder from a PROFIdrive encoder telegram to a SINAMICS encoder object, e.g. to an SMC module
  - Encoder from a PROFIdrive encoder telegram to a PROFIBUS encoder module, e.g. SIMODRIVE sensor
  - Encoder from a PROFIdrive encoder telegram to a PROFINET encoder module, e.g. MC encoder
  - Encoder from a PROFIdrive encoder telegram to any other encoder module
- Actual value as a direct value in the I/O area
- Only C2xx: Actual value via encoder signal evaluation directly at the SIMOTION CPU

For further information, refer to the relevant hardware manuals.

10.2 Encoder mounting type

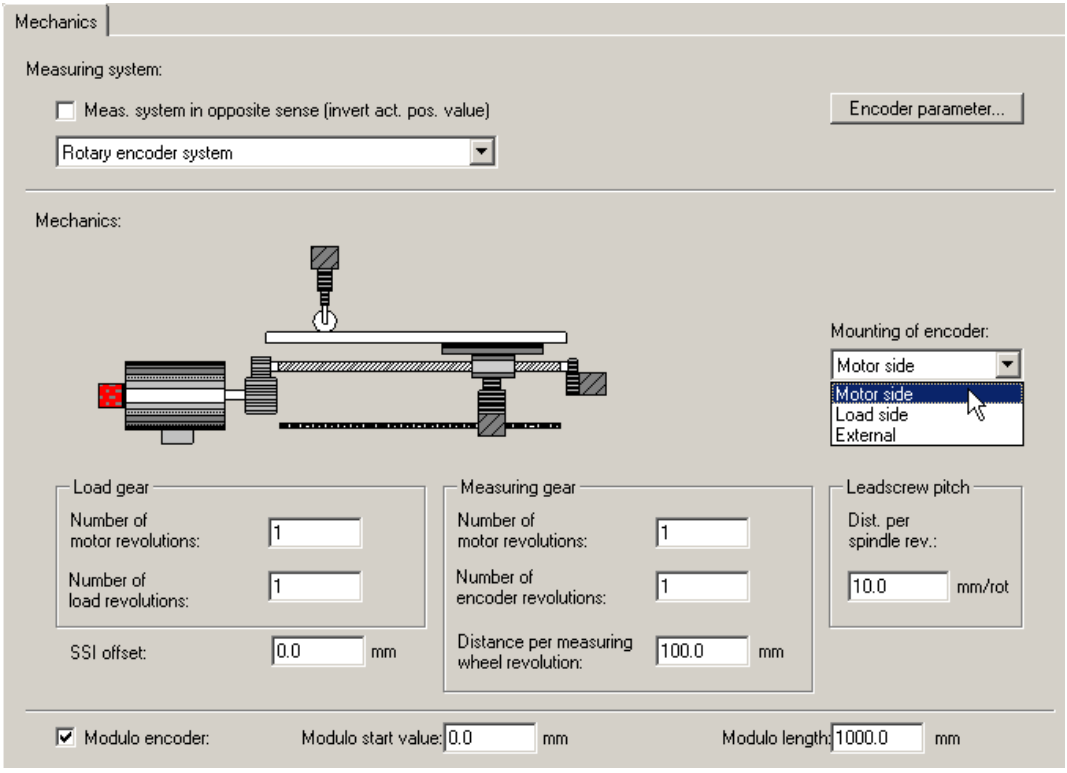


Figure 10-1 Definition of the mounting type of an external encoder in SIMOTION SCOUT

Mounting types for the external encoder

- On the drive side
- On the load side
- External

The transmission ratios must be entered via configuration data:

- For "drive side" mounting using the **adaptdrive** configuration data element
- For "load side" mounting using the **adaptload** configuration data element
- For "external" mounting using the **adaptextern** configuration data element
- For linear encoder system mounting, the transmission ratio is always 1.

10.3 Encoder for position

The controller uses an encoder to detect the axis position.

From a technological standpoint, the following encoder types can be differentiated:

- Incremental encoder
  On the controller side, only the difference between two read encoder values is evaluated. Values are read out in the servo cycle clock at equal intervals. In order to determine the mechanical axis position, the axis must be homed each time the system is switched on. Position zero is displayed after it is switched on.
- Absolute encoder
  This encoder supplies the absolute value or, in the case of an absolute value in the PROFIdrive message frame, the absolute value is read one time after the system is switched on. After this, the actual value is processed in the same way as with the incremental encoder.
  The absolute value supplied by the encoder is assigned to the associated mechanical axis position by means of the absolute encoder adjustment. The absolute encoder adjustment occurs only one time, and the controller remembers the compensation value/absolute encoder offset even after it is powered on/off.
  Certain situations, such as an encoder failure, a restart, etc., can require an absolute encoder re-adjustment. For more information, refer to the chapter titled *Absolute encoder homing/ Absolute encoder adjustment*.
  The following absolute encoder types are differentiated:
  - Absolute encoder with absolute encoder setting
    The measuring range of this encoder is greater than the axis traversing range.
    The axis position results directly from the current encoder value since this can be uniquely mapped.
    An offset may be entered - it is not necessary to hold overflows within the controller.
    There are no overflows of the absolute actual value stored when SIMOTION is switched off. The next time it is switched on, the actual position value is generated from the absolute actual value only.
  - Absolute encoder with cyclic absolute encoder setting
    The axis traversing range is greater than the range of values measured by the encoder, and the encoder returns an absolute value within its measuring range.
    The controller also counts the number of measuring ranges internally in order to hold a unique actual axis position beyond the range of measured values.
    When SIMOTION is switched off, the overflows of the absolute actual value are stored in the retentive memory area of SIMOTION. The next time it is switched on, the stored overflows are taken into account for the calculation of the actual position value.
    The actual position of the axis is passed internally in a 64-bit integer variable.
    Example of a single-turn encoder with 4,096 increments:
    The position of the encoder is mapped in bits 0 to 11, and the number of overflows of the encoder range in bits 12 to 63.
    Example of a multi-turn encoder with 4,096 x 4,096 increments:
    The position of the encoder is mapped in bits 0 to 23, and the number of overflows of the encoder range in bits 24 to 63.
    The overall position of the axis is retained after the controller is switched off. When the controller is switched on again, if the actual encoder value does not match the actual position stored in the controller, it will be corrected to a maximum of ± ½ the encoder measuring range.

    Note

    If the axis/encoder is moved by more than one half the encoder measuring range with the controller switched off, the actual value in the controller no longer matches the real axis.

See also

Absolute encoder homing / absolute encoder adjustment (Page 95)

10.4 Encoder for velocity

Encoders for detection and display of the speed/velocity can only be applied to speed-controlled axes.

Options are:

- Incremental encoders/absolute encoders with number of increments or pulses/revolution (for electric axes)
- Interval counters (for hydraulic axes)
- Encoders providing the velocity as a direct value in the I/O area (for hydraulic axes)
- Read-out of the speed from the PROFIdrive message frame and provision for technological functionality, e.g. velocity monitoring

10.5 Encoder assignment and terminology

The encoder type is specified with the encoder mode.

Table 10-1 Definable encoder mode depending on the encoder type

Encoder mode	Encoder type		
	Absolute encoder	Cyclic absolute encoder	Incremental encoder
PROFIdrive (with adaptation [1]) (as of V4.2)	x	x	x
Endat (encoder data interface)	x	x	x
SSI (synchronous serial interface)	x	x	-
Sinusoidal	-	-	x
Rectangle	-	-	x
Resolver	-	x [2]	x
Analog encoder (value in the I/O area)	x	-	-

[1] The adaptation setting (set by the system by default with V4.2 or higher and SINAMICS) makes changes based on the encoder set in SINAMICS. The encoder resolution and fine resolution are adopted during runtime.

[2] Possible with single-pole-pair resolver only

When the encoder signals are evaluated in an encoder module or drive, the actual values are transferred to the controller in the normalized format according to the PROFIdrive profile. The type of measured value acquisition is dependent on the drive or encoder module, e.g. Sinus, SSI or Endat. The PROFIdrive encoder mode can therefore be set in the controller (as of V4.2).

As of SIMOTION V4.2, the system makes the settings for communication between SIMOTION and SINAMICS drive (encoder). The project setting **Use symbolic assignment** is activated by

default for new projects as of V4.2. Telegrams are set automatically. Encoder characteristics such as fine resolution, grid spacing, and the absolute value for data width are adapted automatically when the system ramps up.

For encoder data, refer to the data sheet or type plate of the encoder. With SINAMICS, encoder data can be transferred from the drive.

The rest of this chapter is only relevant if the project default setting **Use symbolic assignment** is deactivated as of V4.2, or if you are working with a project which uses SIMOTION V4.1.

Encoder pulses per revolution

The encoder pulses per revolution is specified on the encoder type plate as the number of signal periods per revolution (incremental encoder: Pulses/rev; absolute encoder: Pulses/rev; resolver: Number of pole pairs (for SINAMICS and MASTERDRIVES)).

Configuration data:

- **AbsEncoder.absResolution**
- **IncEncoder.incResolution**

Grid line spacing (linear encoder system)

The grid line spacing is specified on the type plate of the encoder as the distance between lines on the linear measuring system (linear scale).

Configuration data:

- **Resolution.distance**

Fine resolution

The fine resolution of the actual value is the interpolation result of a signal period of an encoder pulse.

The fine resolution steps are generated by the measuring electronics from the raw signal of the encoder pulses. Factors as a multiple of 2 are possible.

Example:

- A rectangle signal has a fine resolution of 1
- Two rectangle tracks (TTL signal) offset by 90° have a fine resolution of up to 4
- Depending on the measuring electronics, a sinusoidal signal can have any fine resolution, in principle, e.g. 2,048

Depending on the defined encoder type, the default value 0 is interpreted differently in SIMOTION (see table: *Default settings for fine resolution in SIMOTION*).

In SIMOTION, the multiplication factor is specified, rather than the shift factor/number of bits (x).

The actual value, including the fine resolution, is indicated in the **sensorData.incrementalPosition** system variable.

Configuration data:

- **AbsEncoder.absResolutionMultiplierCyclic**
- **IncEncoder.incResolutionMultiplierCyclic**

Data width of absolute value (without fine resolution) for absolute encoders

The data width of the absolute value (without fine resolution) is a result of the sum of the bits for representing the number of encoder pulses and the bits for representing the maximum number of revolutions that can be registered by the encoder according to the type plate.

Example:

4,096 pulses/rev (= 2^{12}) and a maximum of 4,096 revolutions that can be registered yields a 12 + 12 = 24-bit data width of the absolute value.

Configuration data:

- **AbsEncoder.absDataLength**

Fine resolution of absolute value in Gn_XIST2.

This parameter for the format of the Gn_XIST_2 is only relevant for encoders via PROFIdrive telegram (for more information, see *Encoder interconnection via PROFIdrive telegram*).

Configuration data:

- **AbsEncoder.absResolutionMultiplierAbsolute**

The fine resolution of the absolute value in Gn_XIST2 must be less than or equal to the fine resolution of the absolute value in Gn_XIST1.

Default setting for fine resolution.

Depending on the encoder mode, the default settings are evaluated by the system as described in the table below. The default settings are used if 0 is assigned to the value.

Table 10-2 Default settings for fine resolutions in SIMOTION

Encoder type	Encoder mode	Fine resolution (Gn_XIST1)	Fine resolution on the absolute value (Gn_XIST2)
Onboard C2xx			
Incremental encoder	Rectangle	4	-
	Stepper motor	1	-
Absolute encoder	SSI	1	1
Encoder in PROFIdrive axis telegram (applies to SINAMICS, SIMODRIVE 611U and MASTERDRIVES)			
Incremental encoder	Rectangle	2048	-
	Sinusoidal	2048	-
	Resolver	2048	-
	Endat	2048	-
Absolute encoder	Endat	2048	512
	SSI	1	1
...Cyclic absolute	Resolver (only pole pair number 1 possible)	2048	512
	Endat	2048	512
	SSI	1	1
PROFIBUS absolute encoder in PROFIdrive encoder telegram			
Absolute encoder	SSI	$2^{(32 - \text{Number of data bits})}$	1

These settings are aligned to the default parameter settings of the corresponding Siemens devices. If the behavior is different, alignment with the encoder should be performed by making a corresponding entry in the configuration data of the technology object or in the parameters of the drive or encoder. Depending on the device it may be necessary to assign the corresponding parameters in the drive or encoder with the value of the exponent (shift factor).

Device-specific features for Masterdrives with Endat encoders:

For Masterdrives with Endat encoders, Endat or SSI can be selected in the encoder mode. However, the fine resolution must always be configured in the axis wizard of the Axis or External Encoder technology object differently from the default settings (see Encoder list (Page 367)).

Default setting:

- Fine resolution (**~Encoder.~ResolutionMultiplierCyclic**) = 0
- Fine resolution of the absolute encoder in Gn_XIST2 (**AbsEncoder.absResolutionMultiplierAbsolute**) =0
- If encoderMode=PROFIDRIVE, the values are evaluated as available, since the correct values are adapted by the drive. A default setting of 0 is not permitted for this setting.

See also

Encoder list (Page 367)

10.6 Encoder list

For the most up-to-date list of encoders you can use with SIMOTION in conjunction with SINAMICS, SIMOVERT-MASTERDRIVES, and SIMODRIVE 611U, go to **http://support.automation.siemens.com/WW/view/de/18769911**.

The encoder list is also documented under *FAQs > Drives > Parameters of the connectable encoders* in the *SIMOTION Utilities & Applications* and in the online help (look for encoder parameter assignment in the index). *SIMOTION Utilities & Applications* is part of the scope of delivery of SIMOTION SCOUT.

10.7 Onboard encoder interface on SIMOTION C2xx

Incremental encoders with TTL signal and absolute encoders with SSI protocol can be connected directly to C230-2 or C240. (See the C230-2 and C240 operating instructions and Encoder list (Page 367))

See also

Encoder assignment and terminology (Page 364)

Encoder list (Page 367)

10.8 Encoder interface using the PROFIdrive message frame

As of SIMOTION V4.2, the system sets up communication between SIMOTION and SINAMICS drive (encoder). For SINAMICS drives and encoders with V4.2 and higher, encoder resolution data is transferred directly from the drive during runtime. Telegrams are set automatically.

The rest of this chapter is only relevant if the project default setting **Use symbolic assignment** is deactivated as of V4.2, or if you are working with a project which uses SIMOTION V4.1.

The encoder values are transmitted in the PROFIdrive telegram (see Table *Telegram types* in Section *Setting as a real axis with digital drive coupling*).

Encoder forced values, status values, and actual values are transmitted in the PROFIdrive telegram.

The encoder behavior on SIMOTION is set as represented in the PROFIdrive protocol.

The encoder parameters are defined via the drive wizard during drive configuration (either user-defined or by selecting the encoder).

Encoder parameters that are entered subsequently in the SIMOTION axis wizard must match the encoder parameters in the drive.

Note

For SINAMICS drives with earlier versions than SIMOTION V4.2, it is possible to transfer the encoder parameters from the drive. When assigning encoders in the axis wizard, click **Data transfer from the drive**.

If you are using a DRIVE-CLiQ component with electronic type plate (e.g., SMI motor, DRIVE-CLiQ encoder) you must first upload the parameters from the drive and save them in the project (online commissioning). If the online commissioning is carried out at a later point, you can work with the default settings of the axis wizard in the meantime during offline configuration. Once online commissioning is complete, upload the drive parameters, save them in the project, run the axis wizard again, and perform the **Data transfer from drive** function.

If you change the encoder data in the drive, you must perform an alignment again in the axis wizard.

Further sources of information:

- Encoder list (Page 367)
- User Manual *SIMODRIVE sensor Absolute encoder with PROFIBUS-DP*
- Operating Instructions *Absolute encoder with PROFINET IO* (Chapter *Operating with SIMOTION*)

Encoder value via PROFIdrive axis telegrams

For further information, refer to the commissioning manuals for the drives.

The first and (if present) second encoder of the PROFIdrive axis telegram can be assigned freely to an External Encoder technology object or to the encoder of an axis.

Encoder value via PROFIdrive encoder telegram 8x

PROFINET/PROFIBUS encoders can be set in accordance with the current specifications of the encoder profile with telegram type 81 and as of V4.2, with telegram type 83. These encoders can be assigned freely. See also **PROFINET/PROFIBUS absolute encoder via PROFIdrive encoder telegram** in the next chapter.

Inconsistent configuration

In the event of errors or inconsistencies between the configuration data in SIMOTION and the parameter settings for the encoder in the drive, a technological alarm is triggered as soon as an online connection is established between the control and the drive/encoder.

As off SIMOTION Runtime V4.4: **Technological alarm 20025** with applicable reason

In SIMOTION Runtime < V4.4: **Technological alarm error 20005: Device type:2, log.address:1234 faulty. (Bit:0, reason: 0x80h)**

For PROFIdrive encoders and encoders on the axes according to PROFIdrive, a comparison of the parameter assignment takes place via the following drive/encoder parameters:

P979 (SensorFormat) according to PROFIdrive, which contains information about the type, resolution, and shift factors.

For drives or encoders that do not support parameter P979, the configuration data is evaluated as valid without alarm message.

Actual value Gn_XIST1

The incremental actual value is transferred cyclically with the defined fine resolution in Gn_XIST1. The incremental actual value in Gn_XIST1 is steadily continued according to the actual value change and reset when the data width of Gn_XIST1 is exceeded. If operating with incremental and absolute encoders, the control evaluates the incremental actual value in Gn_XIST1 according to the settings made for encoder pulses per revolution and fine resolution, or grid line spacing for linear scaling.

When the controller is switched on, the fine resolution value within one encoder signal period is indicated correctly in Gn_XIST1. The initial value for the number of signal periods is set by the drive/encoder, and the actual value can then be steadily continued from this initial value.

In the PROFIdrive profile, the fine resolution is given as "shift factor" (x).

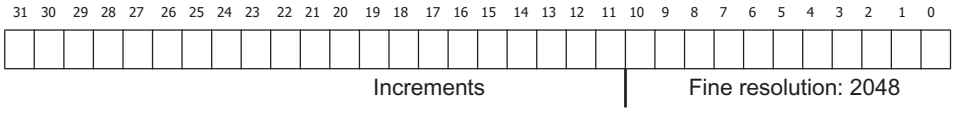


Figure 10-2 Example composition of the 32-bit encoder data of the cyclic actual value Gn_XIST1

Example of for an encoder with number of encoder pulses = 2048 (data width, 11 bits)

The fine resolution in SIMOTION in the **Inc/AbsResolutionMultiplierCyclic** configuration data element is set to the default setting 0 and is thus evaluated as a default fine resolution of 2048 (the default value depends on the encoder mode setting; see Table *Default settings for fine resolution in SIMOTION*).

SIMODRIVE 611U:

Table 10-3 Settings

SIMOTION		611U	
Encoder pulses per revolution [1]	=2048	P1007	=2048
Fine resolution [2]	=0 (≡ 2048)	P1042	=11

[1] Inc/AbsEncoder.Inc/AbsResolution

[2] Inc/AbsEncoder.Inc/AbsResolutionMultiplierCyclic

SINAMICS:

Table 10-4 Settings

SIMOTION		SINAMICS	
Encoder pulses per revolution [1]	=2048	P408	=2048
Fine resolution [2]	=0 (≡ 2048)	P418	=11

[1] Inc/AbsEncoder.Inc/AbsResolution

[2] Inc/AbsEncoder.Inc/AbsResolutionMultiplierCyclic

Note the information about the SINAMICS alignment.

Actual value Gn_XIST2

If the positions for the measuring input or homing functions are transferred to Gn_XIST_2 (n = 1 or 2, number of encoder), they are transferred with the fine resolution defined for the encoder. When the absolute value is read, the value in Gn_XIST_2 is evaluated based on the settings for the data width of the absolute value (without fine resolution) in **AbsEncoder.absDataLength** and the fine resolution absolute value in Gn_XIST2 is evaluated in **AbsEncoder.absResolutionMultiplierAbsolute**.

The fine resolution of the absolute value in Gn_XIST2 indicates the fine resolution factor included in the absolute value transfer. This can match the fine resolution of the actual value, but it can also be smaller, for example, if the 32-bit data width in Gn_XIST2 is not sufficient for the entire fine resolution factor as a result of the data width of the absolute value (without fine resolution).

Example:

Encoder pulses per revolution = 2048 (11 bits) and multi-turn resolution of 4,096 revolutions (12 bits)

Thus, the data width of the absolute value without fine resolution is 11 bits + 12 bits = 23 bits.

Therefore, 9 bits remain for the fine resolution in Gn_XIST2 (32 bits - 23 bits = 9 bits). The setting 0 for the fine resolution of the absolute value in Gn_XIST2 is thus evaluated by the system as 512 (= 9 bits).

Table 10-5 Setting the encoder data

Encoder pulses per revolution [1]	2048
Data width of absolute value (without fine resolution) [2]	23

Fine resolution of absolute value in Gn_XIST2 [3]	0 (= 512)
Fine resolution [4]	0 (= 2048)

[1] AbsEncoder.AbsResolution

[2] AbsEncoder.absDataLength

[3] AbsEncoder.absResolutionMultiplierAbsolute

[4] AbsEncoder.AbsResolutionMultiplierCyclic

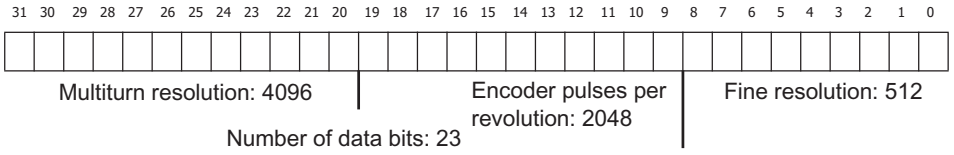


Figure 10-3 Example composition of the 32-bit encoder data of the absolute actual value Gn_XIST2

The number of bits resulting from the data width of the absolute value (without fine resolution) and the number of data bits for the fine resolution of the absolute actual value must not exceed 32. If it is less than 32, leading zeroes are added in Gn_XIST2.

Resolver in PROFIdrive axis telegram

For SINAMICS and MASTERDRIVES, parameters are assigned for the pole pair number of the resolver rather than the encoder pulses per revolution (example: 8-pole resolver = 4 pole pairs → input value = 4).

With SIMODRIVE, parameters are assigned for the encoder pulses per revolution based on parameter P1011.2

As of V4.1 SP1, the resolver with pole pair number 1 is supported as an absolute encoder with the cyclic absolute setting. (encoder pulses per revolution = 1, data width of the absolute value = 0, default value evaluation: fine resolution = 2048, fine resolution Gn_XIST2 = 512)

When a 1-pole resolver is used as Endat encoder, the p418(XIST1) and p419(XIST2) parameters must be set to "11" to prevent information loss of the absolute position. (Settings on the axis: absolute encoder, cyclic absolute, Endat, line count = 1, fine resolution = 2048, fine resolution absolute value = 2048, data width = 0)

See also the Encoder list (Page 364).

PROFINET/PROFIBUS absolute encoder via PROFIdrive encoder telegram

The data width of the encoder value in the technology object configuration data in SIMOTION must match the parameter settings for the PROFINET/PROFIBUS absolute encoder in HW Config.

See also the Encoder list (Page 364).

Example:

Parameter settings of a PROFIBUS absolute encoder in HW Config with 24-bit data width of the absolute value.

The 'SIMODRIVE isochronous sensor' PROFIBUS absolute encoder in HW Config is defined according to the default setting for 24-bit data width and encoder pulses per revolution of 4096: measuring steps per revolution = 4096
24-bit data width for the total resolution yields 0x01000000 (32-bit HEX number). This number, represented separately in HighWord and LowWord, equals 0x0100 in the HighWord and 0x0000

in the LowWord. The decimal values of these two parts (0x0100 = 256 decimal) are to be entered as follows:
total resolution (high) = 256
total resolution (low) = 0

This results in the following consistent configuration for the technology object:
the encoder value is transferred left-justified to Gn_XIST1; the unused bits of the fine resolution are set to 0 according to PROFIdrive, but must be specified in the fine resolution of the actual value. This results in a fine resolution of 8 bits (32 bits - 24 bits = 8 bits) (2^8 = 256 as a factor).

According to the setting above, the absolute value in Gn_XIST2 has a right-justified alignment and therefore a fine resolution of the absolute value in Gn_XIST2 of 0 bits (2^0 = 1 as a factor).

Encoder via PROFIdrive axis telegram on ADI4 and IM174

At least one electric or hydraulic axis must be configured on ADI4/IM174.

The defined update rate (BaudRate) for SSI encoders must be supported by the encoder.

See also the Encoder list (Page 364).

For more information about configuration and operation, refer to the *ADI4 - Analog Drive Interface for 4 Axes Manual* and *Distributed I/Os IM 174 PROFIBUS Module Manual*. These documents are provided on the *SIMOTION Documentation* DVD.

Note

An encoder in a PROFIdrive axis telegram can only be used for an external encoder TO or for the encoder of a TO axis in cyclical operation if a TO axis is also created on the PROFIdrive telegram and is not deactivated.

See also

Encoder assignment and terminology (Page 364)

Encoder list (Page 367)

10.9 Encoder interface as a direct value in the I/O area

Encoders can be used that

- Provide actual value information directly as absolute value in the input/output area.
- Provide a counter value in the peripheral area (as of V4.0).
- Provide an actual velocity in the peripheral area.

Actual value information directly as absolute value

These encoders must be parameterized and operated as absolute encoders, for example with respect to homing.

The following settings are provided to set the **adaptation to the properties of the measured value**:

- The **justification of the measured value** in the **NumberOfEncoders.Encoder_n.AnalogSensor.DriverInfo.format** configuration data element
  - Signed left-justified (VALUE_LEFT_MARGIN)
  - Signed right-justified (VALUE_RIGHT_MARGIN)
  - Unsigned left-justified (VALUE_LEFT_MARGIN_WITHOUT_SIGN)
  - Unsigned right-justified (VALUE_RIGHT_MARGIN_WITHOUT_SIGN)
- The **data width of the measured value** without the sign bit in the **NumberOfEncoders.Encoder_n.analogSensor.DriverInfo.resolution** configuration data
- The upper limit and lower limit, the **maximum limits of the measured value** in the following configuration data
  - **NumberOfEncoders.Encoder_n.AnalogSensor.DriverInfo.maxValue**
  - **NumberOfEncoders.Encoder_n.AnalogSensor.DriverInfo.maxValue**

Example: Use of an ET 200S, SSI module, or an analog input.

Justification of the measured value

The measured value is mapped to an internal 32-bit wide signed data value of the DINT type in accordance with the justification specification. The mapped value is then tested with actual value limitation against the maximum limits and evaluated using the weighting factor for the **NumberOfEncoders.Encoder_n.AnalogSensor.ConversionData.factor** direct value that specifies the technological resolution or assignment of the LSB.

⚠ WARNING

Weighting factor for analog sensors

The scaling or conversion in the value of the configuration data **ConversionData.factor** for the position of the analog sensor has changed in SIMOTION V4.4.

As of V4.4, the following applies:

The values are entered directly in inches and no longer converted from mm to inches for the configured Inch unit system. This can result in a higher velocity being detected (by a factor of 25) if it is an axis that works in inches.

The following applies for SIMOTION up to V4.4:

The values were entered in mm and then converted to inches (1 inch corresponds to approx. 25 mm). Please check the values entered for **ConversionData.factor** in older projects and correct them when necessary.

Regardless of the sign, the encoder resolution/measured value width is limited to maximum 31-bit, because the configuration data of the maximum limits are signed data values of the DINT type.

For the **Left-justified/Right-justified without sign** setting, the measured value for the setting of the encoder resolution/measured value width is processed as follows:

- At ≤16 bits the measured value left-justified/right-justified is mapped to the least-significant byte 1 and byte 2 of an internal data value of the DINT type. In this process, the missing 16 bits minus the measured value width of the least-significant byte 1 and byte 2 are right/left-padded with zeros and the most-significant byte 3 and byte 4 are filled with zeros.
- > At 16 bits the measured value left-justified/right-justified is mapped to an internal data value of the DINT type. In this process, the missing 32 bits minus the measured value width are right/left-padded with zeros.
  As this mapped measured value is tested against the maximum limits of the data type DINT, the encoder resolution/measured value width is limited to maximum 31 bits. This means the measuring range is limited to maximum 50% of the measured value width.

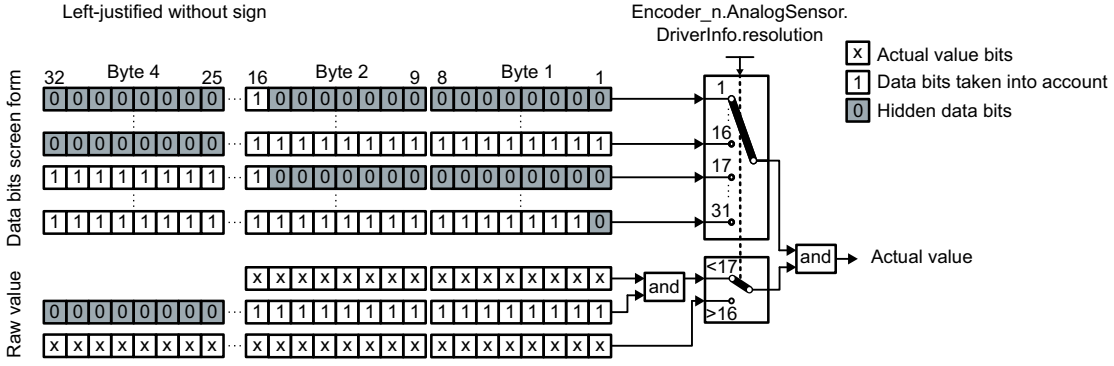


Figure 10-4 Masked reading of the analog IO value left-justified without sign

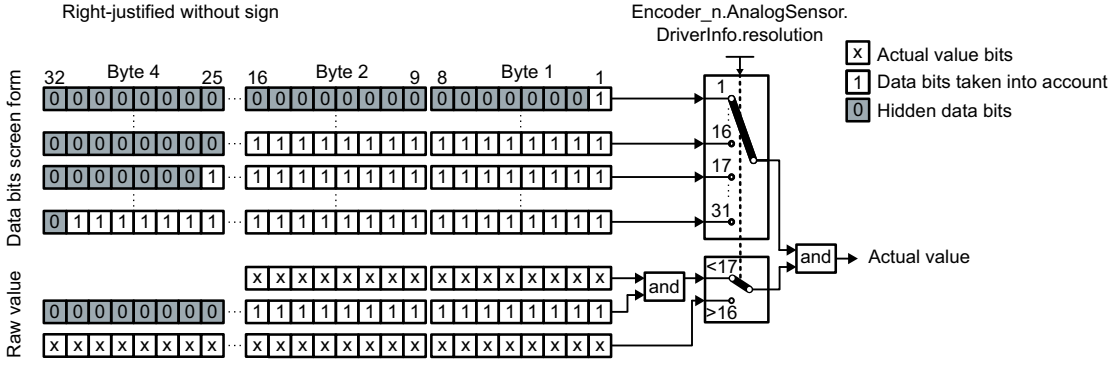


Figure 10-5 Masked reading of the analog IO value right-justified without sign

For the **Left-justified/Right-justified without sign** setting, the measured value for the setting of the encoder resolution/measured value width is processed as follows:

- At <16 bits the measured value left-justified/right-justified is mapped to the least-significant byte 1 and byte 2 of an internal data value of the DINT type. For left-justified, the missing 15 bits minus measured value width of the least-significant byte 1 and byte 2 are right-padded with zeros and the sign bit is extended left to bit 32. For right-justified, the sign bit is padded left to bit 32.
- At ≥16 bits the measured value left-justified/right-justified is mapped to an internal data value of the DINT type. For left-justified, the missing 31 bits minus measured value width are right-padded with zeros. For right-justified, the sign bit is padded left to bit 32.

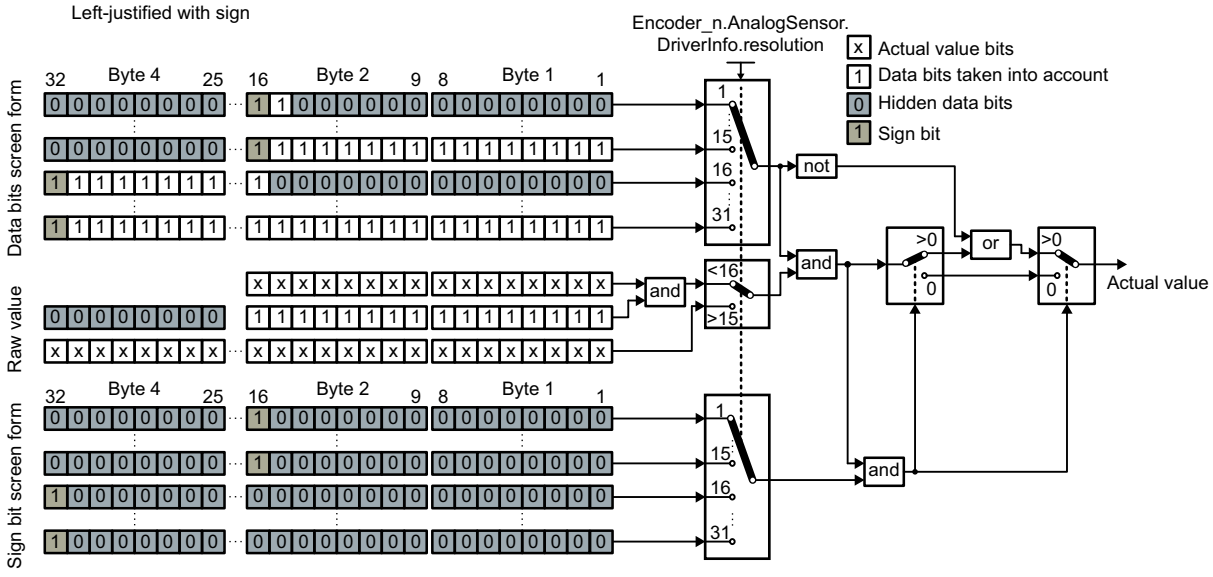


Figure 10-6 Masked reading of the analog IO value left-justified with sign

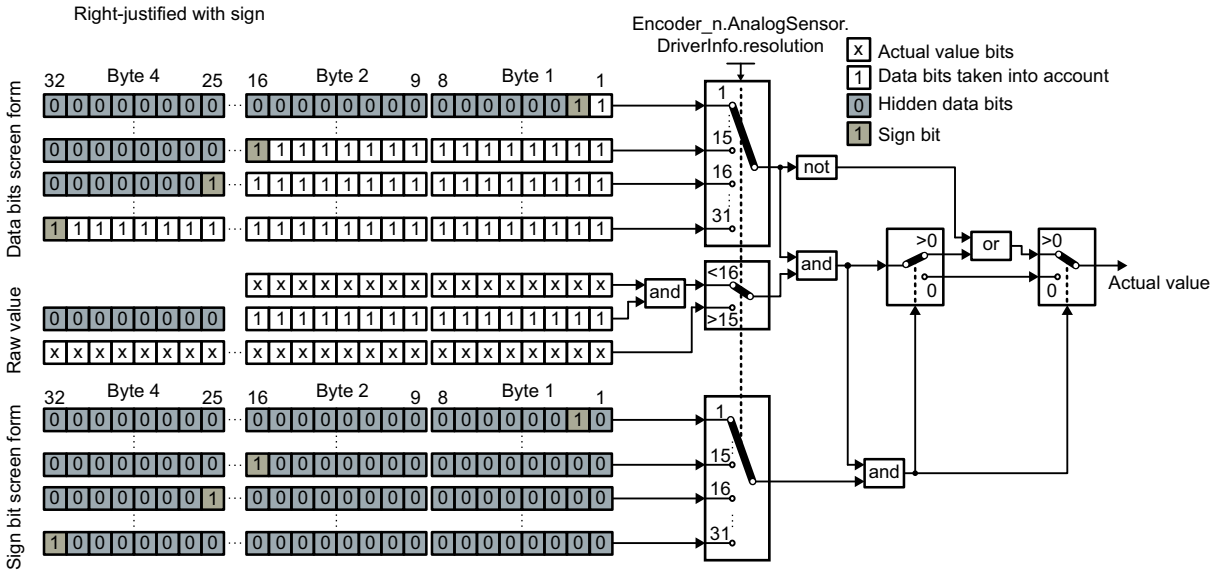


Figure 10-7 Masked reading of the analog IO value right-justified with sign

Counter value (as of V4.0)

The encoder is set as an incremental encoder. 16 bits or 32 bits can be set as counter value width.

Example: Use of an ET 200S, COUNT module

Actual velocity

The actual value information can include the number of pulses between two scans or, alternatively, the time interval between two sequential pulses. This type of encoder is used, for example, to measure actual velocities with the hydraulic axis functionality.

Example: Use of an ET 200S, COUNT module

Direct value as absolute value in I/O area (as of V4.1 SP1)

The following bits can be configured for the direct value as absolute value in the I/O area:

- **Ready bit** via the elements of the **analogSensor.readyStateMonitoring** configuration data element
- **Error bit** via the elements of the **analogSensor.errorStateMonitoring** configuration data element

These can be used for evaluation on the Axis technology object of a *ready identifier* and an *error identifier*, which are available from the peripheral module in addition to the measured value.

In SIMOTION V4.1 SP1, this configuration data is defined directly in the expert list.

If the *not ready* status or an error is displayed in these identifiers during operation, and the **ready bit** or **error bit** is configured, technology alarm 20005 is issued with the *sensor error* identifier.

If the Axis technology object is ready during ramp-up, but the direct value in the I/O area is not yet in ready status, the *WAIT_FOR_VALID Sensor* status is displayed on the encoder. (**sensorData[n].state** system variable)

As of V4.1 SP1, the direct value in the I/O area does not have to be updated in every equidistant communication cycle (for example, if an encoder connected to the peripheral module is not able to supply a new measured value in each cycle during a fast communication cycle clock due to measurement reasons or processing time reasons). In such cases, the actual value is extrapolated by the control.

The control supports the following options:

- The peripheral module displays the new measured value set in an update bit/update counter. The update bit or update counter is defined in the **analogSensor.UpdateCounter** configuration data element.
  UpdateCounter configuration: The update counter can be one (toggle) bit or several (counter) bits in width.
- The refresh cycle of the actual value in the peripheral module is known and is defined directly in the **analogSensor.UpdateCounter.updateCycle** configuration data element.
  Default setting with refresh cycle = 1 (default behavior for updating in every cycle clock)

10.10 Actual value system

Actual value processing for the external encoder is described using the actual value processing on the axis.

10.11 Overflow bit for modulo counting

The number of modulo revolutions (V3.2 and higher) is described in Chapter Actual value measurement / Actual value system.

See also

Actual value measurement / actual value system (Page 130)

10.12 Actual value smoothing

The actual values are read in the position control cycle clock.

Other quantities, such as the velocity, are calculated from these data. The system variables for **SensorData** are calculated in the position control cycle clock, whereas the system variables for **MotionState** are calculated in the IPO cycle clock.

The filter for the velocity in the position control cycle clock is set in **typeOfAxis.Encoder_1.filter**.

The filter for the velocity in the interpolator cycle clock is set in **typeOfAxis.SmoothFilter**.

See also

Actual value measurement / actual value system (Page 130)

10.13 Actual value extrapolation

In the case of synchronous operation with master value reference to external encoder position values, a setting can be made during configuration to specify whether the actual values are to be generated using extrapolation (see "Actual value coupling" in the Synchronous Operation description of functions).

The velocity is filtered / averaged separately using a filter that is set via **extrapolation.Filter**.

The extrapolated actual values are displayed in **extrapolationData.position** and **extrapolationData.velocity**.

See also

Actual value measurement / actual value system (Page 130)

10.14 Standstill signal

The standstill signal **motionState.stillstandVelocity** is ACTIVE when the current velocity is less than a configured velocity threshold for at least the duration of the delay time.

The **StandStillSignal** configuration data element can be used to specify when the standstill signal is to be output.

10.15 Monitoring functions

- **Zero mark monitoring** for incremental encoders
  You can activate the function for monitoring the number of increments between two encoder zero marks.
- Permissible **changes to the actual value** of an absolute encoder
  The user can activate the function for monitoring permissible changes to the actual value for an absolute encoder.
- **Limiting frequency**
  The encoder limit frequency can be monitored.
- **Current velocity**
  The permissible maximum value of the actual velocity can be monitored. If the maximum value is exceeded, the system variable **sensordata.sensorMonitoring.velocity** is output as **limitexceeded**. The velocity is not limited to this value.

- **Cyclical data exchange** (as of V3.1)
  The system variable **sensorMonitoring.cyclicInterface** indicates whether cyclic data exchange is active on the active encoder. The cyclical data exchange is determined using the sign-of-life.
  Application: The encoder may be located on a different drive than the actuator of the axis, or it may even have its own protocol (PROFIBUS/PROFINET encoder).
- **Number of the active encoder** (as of V3.1)
  The number of the active encoder is output directly via system variable **sensorMonitoring.actualSensor**.

10.16 Synchronization / Homing

10.16.1 Overview of synchronization / homing

SIMOTION supports different synchronization/homing modes for the external encoders.

You can set the reference position of the external encoder using the **_synchronizeExternalEncoder()** function.

10.16.2 Synchronization/homing with incremental encoders

- **Direct homing / Set home position**
  Direct homing is set with the function parameter synchronizingMode=DIRECT_HOMING.
  The current actual position of the encoder is set to the value of the home position coordinate (**syncPosition** system variable) without a traversing motion having taken place.
- **Relative direct homing**
  Relative direct homing is set with the function parameter synchronizingMode=DIRECT_HOMING_RELATIVE.
  The current actual position of the encoder is offset by the value of the home position coordinate
  (**syncPosition** system variable), without a traversing motion having occurred.
- **Passive homing/flying homing**
  Passive homing is set with the function parameter synchronizingMode=PASSIVE_HOMING.
  At the synchronization point, the external encoder is set to the value of the home position coordinate.
  IncHomingEncoder.passiveHomingMode can be set as synchronization event in the configuration data:
- Encoder zero mark (ZM_PASSIVE)
- External zero mark (CAM_PASSIVE)
- Next encoder zero mark after homing output cam (CAM_AND_ZM_PASSIVE)
  The path traveled after the homing output cam is exited up until the encoder zero mark is reached can be monitored.
- Encoder zero mark or external zero mark system setting dependent on the encoder type (DEFAULT_PASSIVE).

See also

Passive homing/on-the-fly homing (Page 93)

Direct homing/setting the home position (Page 94)

Relative direct homing/relative setting of home position (V3.2 and higher) (Page 94)

10.16.3 Synchronization/homing with absolute encoders

Direct homing/Set home position

Direct homing is set with the function parameter synchronizingMode=DIRECT_HOMING.

The current actual position of the encoder is set to the value of the home position coordinate (**syncPosition** system variable) without a traversing motion having taken place.

Relative direct homing

Relative direct homing is set with the function parameter synchronizingMode=DIRECT_HOMING_RELATIVE.
The current actual position of the encoder is offset by the value of the home position coordinate.

(**syncPosition** system variable), without a traversing motion having occurred.

Absolute value conversion

The value of the external encoder is set equal to the encoder value + absolute encoder offset using the **_synchronizeExternalEncoder()** command and the setting of the function parameter synchronizingMode=ENABLE_OFFSET_OF_ABSOLUTE_ENCODER. The absolute encoder offset is set in the **absHomingEncoder.absshift** configuration data.

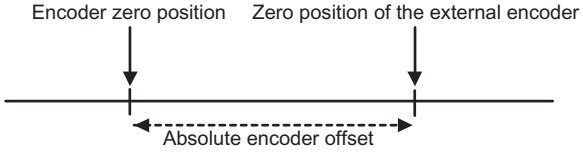


Figure 10-8 The external encoder value is the encoder zero position plus the absolute encoder offset.

The absolute encoder offset (as of V3.2) may be set as an additive or absolute value.

This absolute encoder offset is stored in the NVRAM and remains in effect until the next absolute encoder adjustment. This function must therefore be executed once when the control is commissioned.

The offset value and its calculation are set during **configuration**.

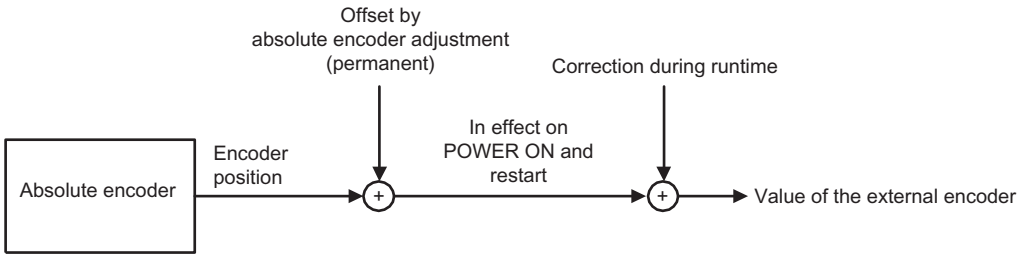


Figure 10-9 Incorporating the absolute encoder offset

A value may be set as a total offset using the **absHomingEncoder.setOffsetOfAbsoluteEncoder** and **absshift** configuration data.

Setting an additive offset

Setting **absHomingEncoder.setOffsetOfAbsoluteEncoder**=RELATIVE (default behavior):

- Actual external encoder value = actual encoder value + (previous offset already in effect + absshift)
- (new) offset = previous offset + absshift

absHomingEncoder.absshift is added to the existing absolute encoder offset whenever the **_synchronizeExternalEncoder()** function is called.

Setting an absolute offset (as of V3.2)

Setting **absHomingEncoder.setOffsetOfAbsoluteEncoder**=ABSOLUTE (as of V3.2):

absHomingEncoder.absshift is set as the absolute encoder offset whenever the **_synchronizeExternalEncoder()** function is called.

- Actual external encoder value = actual encoder value + **absshift**

Setting the value of the external encoder to a predefined position (as of V4.1 SP1)

When the synchronizingMode=SET_OFFSET_OF_ABSOLUTE_ENCODER_BY_POSITION function parameter is set on the **_synchronizeExternalEncoder()** command, the value in the syncPosition parameter will be set as the current position. The resulting absolute encoder offset is calculated with this value, indicated in the **absoluteEncoder[n].totalOffsetValue** system variable, and stored as a retain variable in the system.

The value in the **absHomingEncoder.absshift** configuration data element is not changed.

Displaying the offset

The offset can be read out. (as of V3.1)

The complete offset is indicated in the **absoluteEncoder.totalOffsetValue** system variable and the activation status of the complete offset indicated in the **absoluteEncoder.activationState** variable.

In addition, the status indicates whether the **_synchronizeExternalEncoder()** function with synchronizingMode= ENABLE_OFFSET_OF_ABSOLUTE_ENCODER has been executed at least once after the project was downloaded.

States that require a new setting of the external encoder value

- Once a new project has been downloaded to the control, the stored offset is no longer available.
  If the control already contains a project before the new project is downloaded, and if the technology object name is not changed, the stored offset is retained (as of V4.1 SP1). This behavior also applies for an upgrade, i.e. it is not version-dependent.
- After power is cycled off and on, the offset is deleted unless the project was saved to the ROM.
- After a memory reset.

See also

Absolute encoder homing / absolute encoder adjustment (Page 95)
Direct homing/setting the home position (Page 94)
Relative direct homing/relative setting of home position (V3.2 and higher) (Page 94)

Programming External Encoders/Reference 11

11.1 Commands

The External Encoder technology object can be addressed in the user program using the following commands:

- **_enableExternalEncoder()**
  Activate external measuring system; enables updating of actual values in the IPO.
- **_disableExternalEncoder()**
  Deactivate external measuring system; disables updating of actual values in the IPO. Actual values are frozen in the IPO, they remain unchanged until the next activation of the measuring system.
- **_resetExternalEncoder()**
  Reset External Encoder technology object.
- **_resetExternalEncoderError()**
  Reset an error on the External Encoder technology object.
- **_synchronizeExternalEncoder()**
  Homing of the measuring system by setting the synchronous position directly or by enabling the synchronous position with the next external zero mark / encoder zero mark / homing output cam and encoder zero mark or absolute encoder adjustment.
  The command is only performed in the **active** TO state.
- **_resetExternalEncoderConfigDataBuffer()**
  This function clears the configuration data collected in the buffer since the last activation without activating them.
- **_bufferExternalEncoderCommandID()**
  Enables the saving of commandId and the associated command status beyond the execution period of the command.
- **_removeBufferedExternalEncoderCommandId()**
  Stops the saving of commandId and the associated command status beyond the execution period of the command.
- **_enableMonitoringOfEncoderDifference()**
  Activates the monitoring of the maximum permissible difference between the measuring systems specified in the command.
- **_disableMonitoringOfEncoderDifference()**
  Deactivates the encoder monitoring.
- **_getStateOfExternalEncoderCommand()** (as of V3.1)
  Read out command status
- **_getExternalEncoderErrorNumberState()** (as of V3.1)
  Readout of error number status

- **_redefineExternalEncoderPosition()** (as of V3.2)
  Sets the encoder coordinate system
- **_cancelExternalEncoderCommand()** (as of V4.1 SP1)
  Cancels the command with the specified CommandID.

Note

You can also find information about the system functions in the *SIMOTION Cam Technology Package parameters manual*.

11.2 Technological alarms

11.2.1 Possible alarm reactions

- **NONE**
  No response.
- **DECODE_STOP**
  Command preparation aborted. The current function remains active.
  Processing on the technology object can be continued after **_resetExternalEncoder()** or **_resetExternalEncoderError()**.
- **SIMULATION_STOP**
  Calculation of the simulation value by a function is stopped. The simulation function is not aborted. Simulation can be continued with **_resetExternalEncoderError()**.
- **SIMULATION_ABORT**
  Calculation of the simulation value by a function is stopped. The simulation function is aborted.
- **ENCODER_DISABLE**
  Stops and aborts all commands.

Note

Local alarm responses are specified by means of the system.

Index

A

Absolute actual value, external encoder, 70, 370
Absolute encoder, 95, 380
    Synchronization, 380
Absolute encoder adjustment, 95, 380
Absolute value conversion, 380
Acceleration ramp, 206
Active homing
    With encoder zero mark, 91
    With external zero mark, 89
    With reference cam and encoder zero mark, 87
Actual value acquisition, Axis technology object
    Filter, 132
Actual value extrapolation for external encoder, 378
Actual value smoothing for external encoder, 377
Adaptation, 39
ADS, 200
Alarm reactions
    Axis, 353
Alarm responses
    External Encoder, 384
Analog drive link, 38
Automatic configuration, 19
Automatic controller optimization
    Settings for automatic optimization, 350
Automatic controller setting
    Position controller, 235
    Speed controller, 231
Axis
    Automatic configuration, 19
    Canceling/deleting command, 327
    Homing, 84
    No drive assignment, 36
    Real, 36
    Reset error, 327
    Resetting, 325
    Virtual, 36
    With analog drive link, 38
    With C2xx stepper drive, 58
    with digital drive link, 53
    with encoder signal simulation, 59
    with force/pressure control, 36
Axis command
    _bufferAxisCommandId(), 345
    _cancelAxisCommand(), 327
    _continue(), 325
    _disableAxis(), 313
    _disableAxisSimulation(), 341
    _disableForceLimiting(), 337
    _disableMovingToEndStop(), 176
    _disableQFAxis(), 313
    _disableTorqueLimiting(), 333
    _disableVelocityLimiting(), 338
    _enableAxis(), 313
    _enableAxisSimulation(), 341
    _enableForceControlByCondition(), 321
    _enableForceLimiting(), 337
    _enableMotionInPositionLockedForceLimitingProfile(), 334
    _enableMotionInPositionLockedVelocityLimitingValue(), 338
    _enableMovingToEndStop(), 176
    _enablePositionLockedForceLimitingProfile(), 334
    _enablePositionLockedVelocityLimitingValue(), 338
    _enableQFAxis(), 313
    _enableTimeLockedForceLimitingProfile(), 335
    _enableTimeLockedVelocityLimitingValue(), 339
    _enableTorqueLimiting(), 173, 333
    _enableVelocityLimitingValue(), 338
    _getAxisDataSetParameter(), 347
    _getAxisErrorNumberState(), 345
    _getAxisErrorState(), 345
    _getAxisStoppingData(), 349
    _getForceControlDataSetParameter(), 348
    _getMotionStateOfAxisCommand(), 344
    _getQFAxisDataSetParameter(), 348
    _getStateOfAxisCommand(), 343
    _getStateOfMotionBuffer(), 345
    _homing(), 327
    _move(), 329
    _pos(), 330
    _removeBufferedAxisCommandId(), 345
    _resetAxis(), 325
    _resetAxisError(), 327
    _resetMotionBuffer(), 346
    _runMotionInPositionLockedForceProfile(), 336
    _runMotionInPositionLockedVelocityProfile(), 332
    _runPositionLockedForceProfile(), 336
    _runPositionLockedVelocityProfile(), 332
    _runTimeLockedForceProfile(), 335
    _runTimeLockedPositionProfile(), 333
    _runTimeLockedVelocityProfile(), 331
    _setAxisDataSetActive(), 346
    _setAxisDataSetParameter(), 347
    _setForceControlDataSetParameter(), 347, 348

_setQFAxisDataSetParameter(), 348
_stop(), 323
_stopEmergency(), 322
Properties, 312
Specifying and controlling motions, 309
Axis data set, 200
Axis functions - Overview, 21
Axis setting for hydraulic functionality, 283
Axis settings, 36
Axis simulation, 341
Axis status, 218
Axis technology, 21
Path axis, 24
Positioning axis, 23
Speed-controlled axis, 22
Synchronous axis, 24
Axis types
Electric, 27
Hydraulic, 29
Linear axis, 26
Modulo rotary axis, 26
Rotary axis, 26
TypeOfAxis configuration data, 31
Virtual, 30
Axis-drive relationship, 25

B

Backlash compensation, Axis technology object, 139
Balancing filter, 114
Balancing filters
Setting, 152
Behavior at the end of profile, Axis technology object, 208
Brake control via the axis, 50
Braking ramp, Axis technology object, 163
Limiting, 206

C

C2xx stepper drive
Connecting an axis, 58
Circularity test, 242
Command buffer, Axis technology object, 211, 343
Properties, 211
Command groups, Axis technology object, 210
Commissioning
Position controller for position axis, 147
Connectable encoders (external encoders), 361
Cyclic actual value, external encoder, 69, 369
Cyclic continuation, Axis technology object, 198

D

Data set changeover, Axis technology object, 200
Data types
External Encoder, 359
TO axis, 17
Defaults, Axis technology object, 84
Diagnostics, external encoder, 78
Digital drive link, 53
Additive data block, 179
Stepper drives, 58
Telegram types, 56
Direct homing, 94
External encoder, 379, 380
DPV1 services, 57, 222
Drive assignment
Analog, 38
digital, 53
For the axis, 36
Hydraulic functionality, 283
Drive axis
With force/pressure limiting, 306
Drive Safety Data Block (DSDB), 245, 254
DSC, 118
Basic principles, 118
DSC with spline, 122
DSDB (Drive Safety Data Block), 245, 254
Dynamic response
Setting defaults, Axis technology object, 159
Dynamic response parameters, 158
Dynamic Servo Control (DSC), 118

E

Encoder
absolute, 380
Absolute, 95
Encoder pulses per revolution, 65, 365
Encoder assignment
Adaptation, 64, 364
Non-exclusive, 78
Encoder failure
On the axis, 357
Encoder list, 67, 367
Encoder pulses per revolution, 65, 365
Encoder signal output
Via TM41, 145
Encoder signal simulation
Via TM41, 59

Encoder switchover
Axis technology object, 200
Expert list, Axis technology object
Using, 229
External encoder
Data type, 359
External Encoder, 359
Interconnecting, 359
External encoder command
_bufferExternalEncoderCommandID(), 383
_cancelExternalEncoderCommand(), 384
_disableExternalEncoder(), 383
_disableMonitoringOfEncoderDifference(), 383
_enableExternalEncoder(), 383
_enableMonitoringOfEncoderDifference(), 383
_getExternalEncoderErrorNumberState(), 383
_getStateOfExternalEncoderCommand(), 383
_redefineExternalEncoderPosition(), 384
_removeBufferedExternalEncoderCommandId(), 383
_resetExternalEncoder(), 383
_resetExternalEncoderConfigDataBuffer(), 383
_resetExternalEncoderError(), 383
_synchronizeExternalEncoder(), 383
External encoder synchronization, 379

F

Final controlling element characteristic for hydraulic functionality, 300
Fine interpolation, Axis technology object, 126
Fine resolution of absolute value, external encoder in Gn_XIST2, 70, 370
Fine resolution, external encoder, 65, 365
Default settings, 66, 366
Flying homing
External encoder, 379
Force control for hydraulic functionality, 304
Force control, Axis technology object, 187
Activating, 321
Force limitation on the axis, 337
Force limiting for hydraulic functionality, 304
Force limiting profile on the axis, 334
Force limiting profile, Axis technology object, 206
Force profile, 203, 206
Position-related, 336
Time-related, 335
Force/pressure control, Axis technology object, 190
Commissioning, 195
Control Structure, 187
Emergency strategies, 192
Limits, 192
Monitoring, 192
Setpoint specification, 195

G

Gear parameters, Axis technology object
Preventing overflows, 83
General state diagram in SIMOTION, 53
Overview of commands, 218
Gn_XIST1 external encoder, 69, 369
Gn_XIST2 external encoder, 70, 370

H

Hardware limit monitoring, 107
Homing, 84
Absolute encoder, 95, 380
Absolute encoder adjustment, 97
Differential position measurement, 99
Direct, 94
Displaying the offset, 96, 381
Homing mark monitoring, 97
Relative direct, 94
Setting an additive offset, 96, 381
Setting the axis to a predefined position, 96, 381
Setting the offset as a total value, 96, 381
Homing axes - Terms
Home position, 85
Home position coordinate, 85
Homing output cam, 85
Reference mark, 85
Reference point offset, 85
Synchronization point, 85
Homing of external encoder, 379
Hydraulic final controlling element characteristic, 300
Hydraulic functionality
Additive offset, 299
Additive sliding-friction compensation, 299
Offset application, 299
Pressure difference measurement, 291
Sliding-friction compensation, 299

I

Incremental encoder
Homing, 379
Synchronization, 379
Information functions on the axis, 343
Interpolator
Motion command, 216

J

Jerk limiting for local stop response, 168

L

Leadscrew pitch for linear axes, 176
Limits
  Acceleration ramp, 206
  Backstop, 106
  Braking ramp, 206
  Dynamic limitation functions, 160
  Force, 337
  Force/pressure control, 192
  Manipulated variable, 106
  Pressure limiting, 196
  Printing, 337
  Technological, 160
  Velocity limiting, 195
Linear axis, 26
Linking digital drives, 225
Local alarm response
  Axis, 353
  TO External Encoder, 384
Long-term accuracy
  Modulo axis, 81
Long-term stability
  Modulo axis, 81

M

Man. var. filter, 137, 298
Manipulated variable filter, 136
Manipulated variable preparation
  Electrical axis, 136
Maximum velocity, 162
Measuring function, 238
Mechanics, Axis technology object
  Gear parameters, 83
  Settings, 79
Modulo rotary axis, 26
Monitoring
  Encoder limit frequency, 109
  Following error, 100
  Force limiting, 196
  Force/pressure control, 192
  Hardware end positions, 107
  Increase monitoring, 105
  Limits, 108
  Manipulated variable, 105
  Measuring system differential, 110
  Positioning, 102
  Reference mark, 97
  Slip monitoring, 110
  Standstill (zero-speed) monitoring, 103
  Standstill signal, 104
  Velocity error, 110
Monitoring functions for external encoder, 378
  Current velocity, 378
  Life-sign, 378
  Limiting frequency, 378
  Permissible changes to the actual value of an absolute encoder, 378
  Zero marks for incremental encoders, 378
Motion command, Axis technology object
  Interpolator, 216
  Load, 216
Motion profiles for the hydraulic functionality, 293
Motion profiles, Axis technology object, 203
Motion transition, 216
MotionIn interface, Axis technology object
  Motion vectors, 167
  Traversing, 167
Mounting type for external encoders, 362

O

Operating mode
  Axis simulation mode, 22
  Follow-up mode, axis, 22
  Program simulation mode, axis, 22
  Setpoint mode, axis, 22
  Simulation mode, axis, 22
Overview of commands
  Axis motion commands, 309
  External Encoder, 383

P

P valve, 283
Passive homing
  External encoder, 379
  with encoder zero mark, 94
  with external zero mark, 93
  with homing output cam and encoder zero mark, 93
Path axis
  Data type, 18
PLCopen commands for the axis, 311
Position axis
  Commissioning the position controller, 147

Position controller
Automatic setting, 235
Positioning axis, 23
Data type, 17
Position control, 111
Precontrol, 112
Position-related profiles, Axis technology object, 205
PQ valve, 283, 304
Preparation of manipulated variables for hydraulic functionality, 297
Pressure control for hydraulic functionality, 304
Pressure control, Axis technology object, 187
Activating, 321
Pressure limitation
Axis technology object, 337
Hydraulic functionality, 199
Pressure limiting
Hydraulic functionality, 304
Pressure limiting profile on the axis, 206, 334
Pressure profile, 203, 206
Position-related, 336
Time-related, 335
Pressure profile for the hydraulic functionality
Increase limiting, 199
Profile
Force limiting profile, 206
Force profile, 206
Position profile, 206
Position-related, 205
Pressure limiting profile, 206
Pressure profile, 206
Time-related, 205
Velocity limiting profile, 205
Velocity profile, 205

Q

Q-valve, 283

R

Recording the characteristic, 301
References, 4
Relative direct homing, 94
Retraction, 109
Rotary axis, 26

S

Safety channel, 251
Safety Information Data Block (SIDB), 257
Safety on the Axis technology object, 245
SBT, 272
SDI, 269
SLP, 270
SLS, 267
SOS, 267
SS1, 264
SS2, 266
STO, 264
Safety on the TO axis
SCA, 271
Safety status information for DSDB, 254
Safety status word for SIDB, 259
Set reference point, 94
SIDB (Safety Information Data Block), 257
Signal output, Axis technology object
TM41 module, 146
SIMOTION status diagram
Overview of commands, 218
Software limit monitoring, 108
Speed controller
Automatic setting, 231
Speed precontrol, 123
Speed-controlled axis, 22
Data type, 17
With force/pressure control, 305
Standstill signal for external encoder, 378
State model/axis status, 218
Static friction compensation, Axis technology object, 138
Status diagram
PROFIBUS, 53
Status displays during profile processing, Axis technology object, 206
Stepper drives, Axis technology object
Behavior, 144
Digital drive link, 58
Overload range, 144
Torque characteristic, 144
Stepper motor axis, 228
Superimposed motions, 169
Switching between final controlling elements for hydraulic functionality, 303
Switchover smoothing filter, Axis technology object, 200
Symbolic assignment, 39
Synchronization of absolute value encoder, 380
Synchronization of incremental encoder
Direct homing, 379
Flying homing, 379
Passive homing, 379

Synchronous axis, 24
Data type, 18
System default, 225
System variable for external encoder
MotionState, 377
SensorData, 377
sensordata.sensormonitoring.velocity, 378

T

Technological alarms
External Encoder, 384
TO Axis, 353
Technology data, 179
Technology data block, 179
Telegram types for digital drive link, 56
tfm unit of force, Axis technology object, 33
tfs unit of force, Axis technology object, 33
Time-of-day synchronization, 57
Time-related profiles, Axis technology object, 205
torque limitation
Axis, 182
Torque precontrol, 123
Travel to fixed stop, 176

U

Unit of force, Axis technology object
tfm, 33
tfs, 33
Units
TO axis, 33

V

Valve type
P valve, 283
PQ valve, 283
Q-valve, 283
Velocity controller
Drive axis with hydraulic functionality, 295
Velocity limiting profile, Axis technology object, 205
Velocity profile, 205
Motion control, 153
Position-related, 332
Smooth, 153
Time-related, 331
Trapezoidal, 153