SIEMENS

SIMOTION

Motion Control
TO Axis Electric / Hydraulic, External Encoder

Function Manual

05/2009

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

℗ 05/2009

Preface

Contents

This document is part of the Description of System and Function Documentation Package.

Area of application

This manual is valid for SIMOTION SCOUT in connection with the Technology Package SIMOTION Cam or Cam_ext for Product Version V4.1 SP3.

Chapters in this manual

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

Part I Axis

- Overview

  This chapter provides the user with an overview of the Axis Technology Object.
- Axis Fundamentals

  This chapter explains the basic setting options and functions of the Axis Technology Object.
- Configuring an Axis

  This chapter explains the configuration procedure with reference to various tasks.

Part II Hydraulic Functionality

- Overview

  This chapter provides the user with an overview of the hydraulic functionality of the Axis Technology Object.
- Fundamentals of Hydraulic Functionality

  This chapter explains the basic setting options and functions concerning the hydraulic functionality of the Axis Technology Object.

Part III Programming / Reference

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

Part IV External Encoder

- Description

  This chapter provides the user with an overview of the External Encoder Technology Object.
- External Encoder Fundamentals

  This chapter explains the basic setting options and functions of the External Encoder Technology Object.

- **Configuring an External Encoder (online help only)**

  This chapter explains the configuration procedure with reference to various tasks.
- **Programming External Encoders / Reference**

  This chapter explains the commands and functions in greater detail.

Index

- Keyword index for locating information.

An overview of the SIMOTION documentation can be found in a separate list of references.

This documentation is included as electronic documentation with the supplied SIMOTION SCOUT.

The SIMOTION documentation consists of 9 documentation packages containing approximately 80 SIMOTION documents and documents on related systems (e.g. SINAMICS).

The following documentation packages are available for SIMOTION V4.1 SP3:

- SIMOTION Engineering System
- SIMOTION System and Function Descriptions
- SIMOTION Diagnostics
- SIMOTION Programming
- SIMOTION Programming - References
- SIMOTION C
- SIMOTION P350
- SIMOTION D4xx
- SIMOTION Supplementary Documentation

Technical support

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

	Europe / Africa
Phone	+49 180 5050 222 (subject to charge)
Fax	+49 180 5050 223
Internet	http://www.siemens.com/automation/support-request

	Americas
Phone	+1 423 262 2522
Fax	+1 423 262 2200
E-mail	mailto:techsupport.sea@siemens.com

	Asia / Pacific
Phone	+86 1064 719 990
Fax	+86 1064 747 474
E-mail	mailto:adsupport.asia@siemens.com

Note

Country-specific telephone numbers for technical support are provided under the following Internet address:

http://www.siemens.com/automation/service&support

Calls are subject to charge, e.g. 0.14 €/min. on the German landline network. Tariffs of other phone companies may differ.

Questions about this documentation

If you have any questions (suggestions, corrections) regarding this documentation, please fax or e-mail us at:

Fax	+49 9131- 98 63315
E-mail	mailto:docu.motioncontrol@siemens.com

Siemens Internet address

The latest information about SIMOTION products, product support, and FAQs can be found on the Internet at:

- General information:
  - http://www.siemens.de/simotion (German)
  - http://www.siemens.com/simotion (international)
- Product support:
  - http://support.automation.siemens.com/WW/view/en/10805436

Additional support

We also offer introductory courses to help you familiarize yourself with SIMOTION.

Please contact your regional training center or our main training center at D-90027 Nuremberg, phone +49 (911) 895 3202.

Information about training courses on offer can be found at:

www.sitrain.com

Contents

Part I Axis - Overview 1

1.1 General information about axes

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.). From V3.2 onwards, the **Axis technology object (TO)** is included in the Cam 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 implemented through 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 positioning axis is referred to by the data type **posAxis** in reference lists and when programming.

- Following axis

  The synchronous axis creates a grouping of the slave 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 the use of the synchronous object.

  The slave 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 Interpolation* manual describes how to use the path axis with the path object.

  The path axis is referred to by the data type _pathAxis 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 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 Interpolation, 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 limiting 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 acknowledgements of encoder information.

Note

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

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.

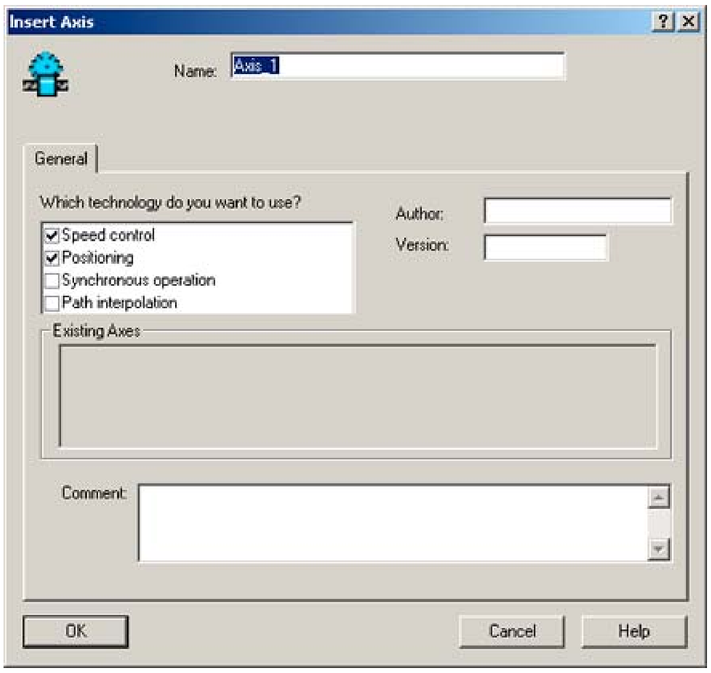
See the *SIMOTION MCC Motion Control Chart*, *SIMOTION ST Structured Text* and *PLCopen* Programming Manuals.

Axis fundamentals 2

2.1 Axis technologies

2.1.1 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.


Insert Axis

Name: Axis_1

General

Which technology do you want to use?

- [✓] Speed control
- [✓] Positioning
- [ ] Synchronous operation
- [ ] Path interpolation

Author:

Version:

Existing Axes

Comment:

Figure 2-1 Axis technology setting in SIMOTION SCOUT

Possible axis functions in relation to the axis type

Table 2- 1 Axis functions - Overview

Function	Drive axis	Position axis	Following axis	Path axis
Speed or velocity specification	X	X	X	X
Travel with torque limitation	X	X	X	X
Motion according to the specifications on the MotionIn interface	X	X	X	X
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

- Follow-up mode

  The setpoint is corrected to the actual value in follow-up mode. The actual position and actual speed values will be updated. The axis can then also be tracked if it is moved by external effects.

  Motion commands are not accepted/executed.

- Program simulation mode

  The axis and position controller are active in 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.

  Program simulation mode is used to test the programmed sequences in the control and the interaction between various axes on the basis of trace recordings without moving the axis.

  This operating mode is only useful for real axes.

  See also Enabling/disabling program simulation (Page 298).

- Axis simulation mode

  It is possible to switch a real axis to axis simulation status even if the drive is not connected.

- Setpoint mode

  Setpoint mode is the "normal" mode of the axis, in which motion commands are accepted and executed.

The operating modes are set via commands.

Speed-controlled axis

The drive axis type is used when the position of the axis is of no importance. Only the speed of rotation 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 of rotation or velocity specified via
  - Programmable velocity profile
  - 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 is used to specify, control, and monitor 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/direction of 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.

Operating modes

As for drive axis, plus

- Position control

Functions

- Speed of rotation or velocity specified via
  - Programmable velocity profile
  - Free velocity profile (time- or position-related)
- Travel with torque limitation

- Position specified via
  - Programmable velocity profile
  - 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)

Following axis

The synchronized axis type is used to determine the axis setpoint from a master value according to conversion rules.

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

The "Axis" and "Synchronous Operation" technology objects have a reciprocal effect on each other in terms of their operating modes 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 that are available via the synchronous object:

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

Other functions associated with the Synchronous Operation 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 modes 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 that are available via the path object:

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

The path axis contains the functionality of the following axis.

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

See also

Setting as a real axis without drive (axis simulation) (Page 45)

Enabling/disabling program simulation (Page 298)

Setting and canceling the axis enables (Page 269)

2.1.2 Axis-drive relationship

The Axis technology object provides the user with technological functionality and the interface to the drive/actuator. It executes control and motion commands and indicates statuses and actual values.

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).

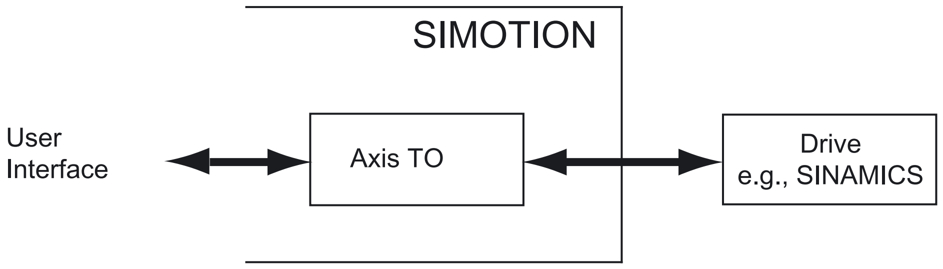


Figure 2-2 Axis-drive relationship

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.

Digital drives or interface modules for analog drives and/or stepper motors can be connected via PROFIBUS/PROFINET using the PROFIdrive profile.

Setpoint specifications and feedback signals (incl. encoder information) for drives connected to a field bus are provided via standardized protocols (standard message frames in accordance with the PROFIdrive profile).

2.2 Axis types

2.2.1 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

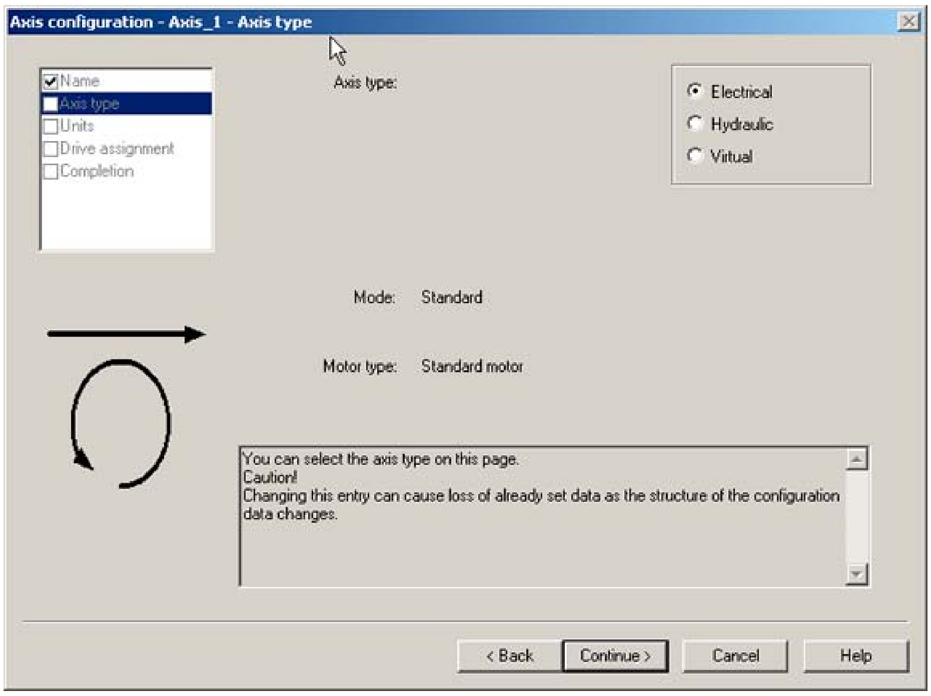
- **Electrical**
- **Hydraulic**
- **Virtual**

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

See also

Actual value measurement / actual value system (Page 110)

2.2.2 Setting for the electric axis type



Axis type: [✓ Electrical] Hydraulic Virtual

Mode: Standard

Motor type: Standard motor

You can select the axis type on this page.
Caution!
Changing this entry can cause loss of already set data as the structure of the configuration data changes.

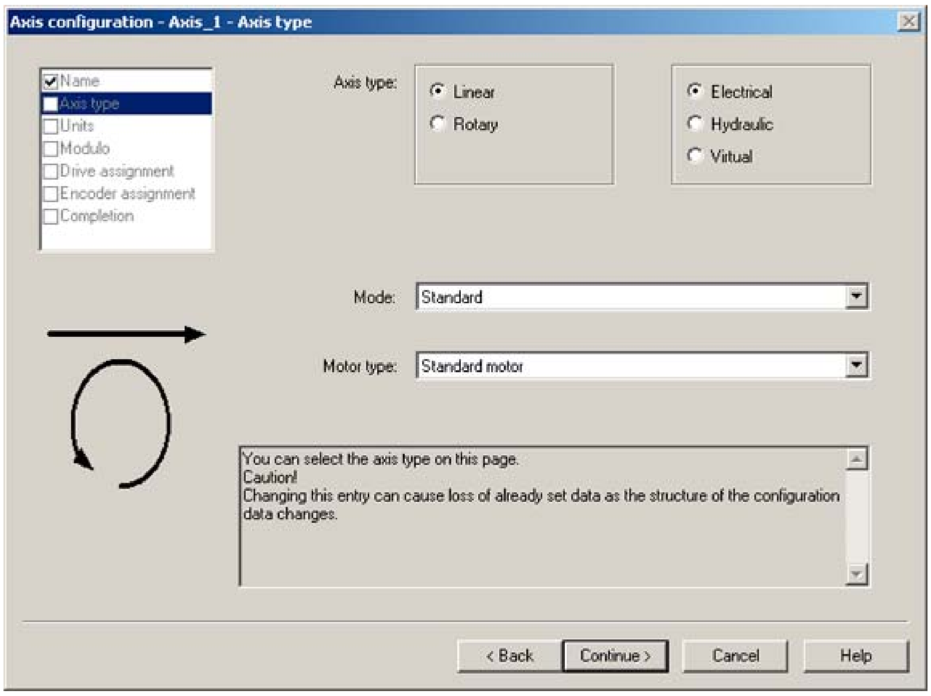
Figure 2-3 Axis type setting for an electric drive axis in SIMOTION SCOUT

Table 2- 2 Motor type options

Motor type	Description
Standard motor	Rotary motor whose characteristics are described using the corresponding physical units, such as speed of rotation and torque.
Linear motor	Motor whose characteristics are described using the corresponding physical units, such as velocity and force.

Note

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


Axis configuration - Axis_1 - Axis type

Axis type: [✓ Linear] Rotary

[✓ Electrical] Hydraulic Virtual

Mode: Standard

Motor type: Standard motor

You can select the axis type on this page.
Caution!
Changing this entry can cause loss of already set data as the structure of the configuration data changes.

Figure 2-4 Axis type setting for an electrical position or following axis in SIMOTION SCOUT

Table 2- 3 Axis type options

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

Table 2- 4 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

Table 2- 5 Motor type options

Motor type	Description
Standard motor	Rotary motor whose characteristics are described using the corresponding physical units, such as speed of rotation and torque.
Linear motor	Motor whose characteristics are described using the corresponding physical units, such as velocity and force.

See also

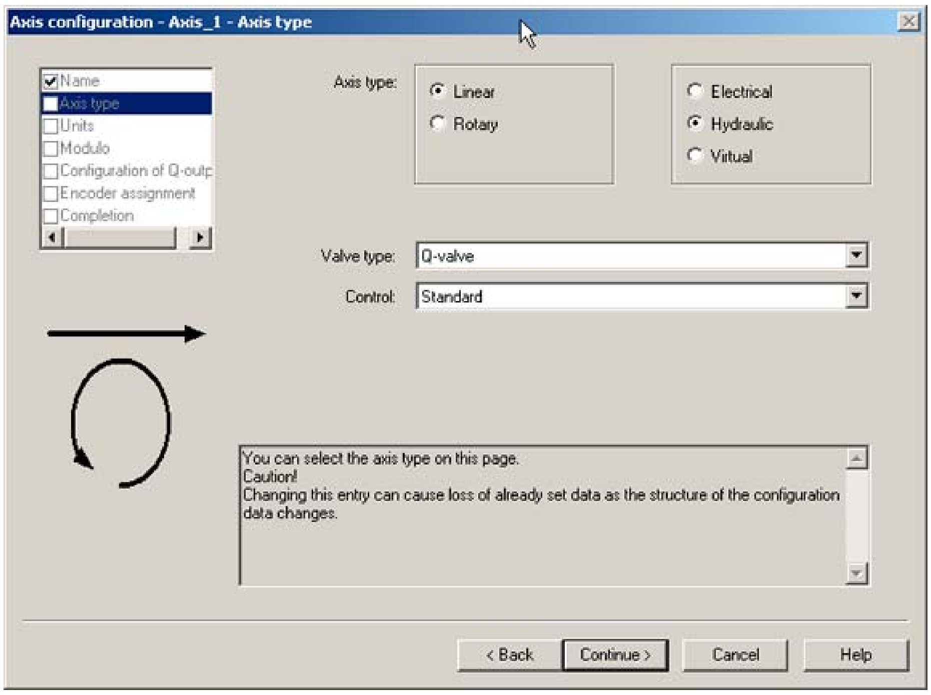
Overview of force/pressure control (Page 165)

Overview of force/pressure limiting (Page 174)

Overview of axis types (Page 24)

TypeOfAxis configuration data (Page 29)

2.2.3 Setting for the hydraulic axis type



Axis type: [✓ Linear] Rotary
Electrical [✓ Hydraulic] Virtual
Valve type: Q-valve
Control: Standard

You can select the axis type on this page.
Caution!
Changing this entry can cause loss of already set data as the structure of the configuration data changes.

Figure 2-5 Axis type setting for a hydraulic position or following axis in SIMOTION SCOUT

Table 2- 6 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 2- 7 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 165)

Overview of force/pressure limiting (Page 174)

Hydraulic functionality overview (Page 233)

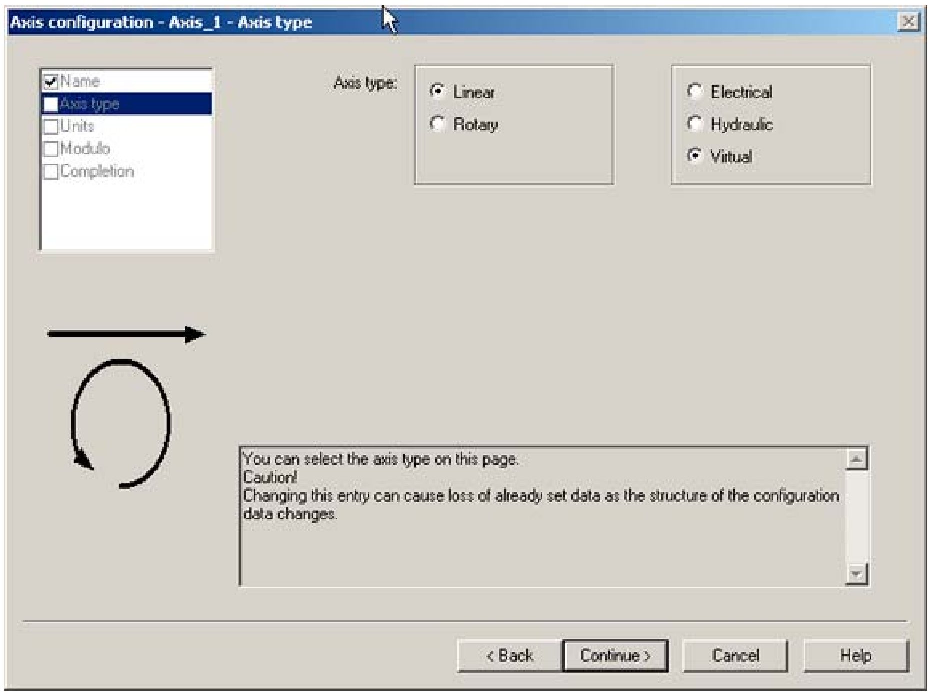
TypeOfAxis configuration data (Page 29)

Setting as a real axis with Q valve only (Page 237)

Setting as a real axis with Q valve + P valve/F output (Page 240)

Setting an axis as a real axis with P valve only (V3.2 and higher) (Page 241)

2.2.4 Setting for the virtual axis type


Axis configuration - Axis_1 - Axis type

☑ Name
☐ Axis type
☐ Units
☐ Modulo
☐ Completion

Axis type: [✓ Linear] Rotary

Electrical Hydraulic [✓ Virtual]

You can select the axis type on this page.
Caution!
Changing this entry can cause loss of already set data as the structure of the configuration data changes.

Figure 2-6 Axis type setting for a virtual position or following axis in SIMOTION SCOUT

2.2.5 TypeOfAxis configuration data

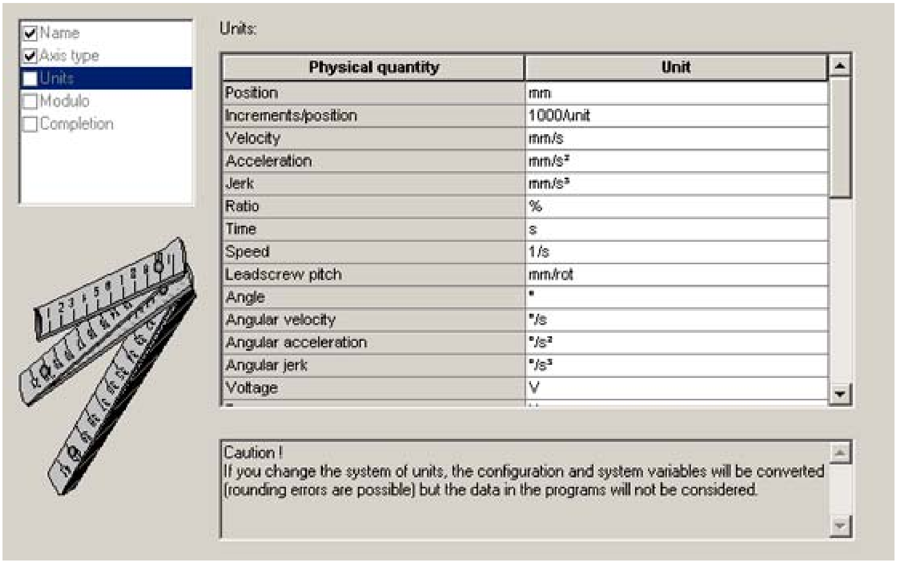
The axis type is entered according to the axis type parameter assignments under the **TypeOfAxis** configuration data in the axis wizard. The following table shows which axis wizard parameterization corresponds to which value under **TypeOfAxis**.

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

	Axis type					Closed-loop control			Valve type [d]		
TypeOfAxis	Linear [c]	Rotary	Electric	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

2.3 Units and accuracies



Units:

Physical quantity	Unit
Position	mm
Increments/position	1000/unit
Velocity	mm/s
Acceleration	mm/s²
Jerk	mm/s³
Ratio	%
Time	s
Speed	1/s
Leadscrew pitch	mm/rot
Angle	°
Angular velocity	°/s
Angular acceleration	°/s²
Angular jerk	°/s³
Voltage	V

Caution !
If you change the system of units, the configuration and system variables will be converted (rounding errors are possible) but the data in the programs will not be considered.

Figure 2-7 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), or tonne force (metric), metric unit
  - tfs

    ton force (short) or ton force (US), US unit

The defined units are used to 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, e.g., controller gains, the units may differ from the units of the individual variables, e.g., 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 these intermediate values.

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 controller 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/unit (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/unit (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: 10.333 mm
- Effective modulo length: 20.333 mm

2.4 Axis settings / drive assignment

2.4.1 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).

2.4.2 Real and virtual axes

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.

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
    Position control and monitoring of axes, drive communication, and I/O processing
  - 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 the fast responses in the motion control, in exception cases, the processing cycle clock can also be set to servo, see also Motion execution/interpolator (Page 187).

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

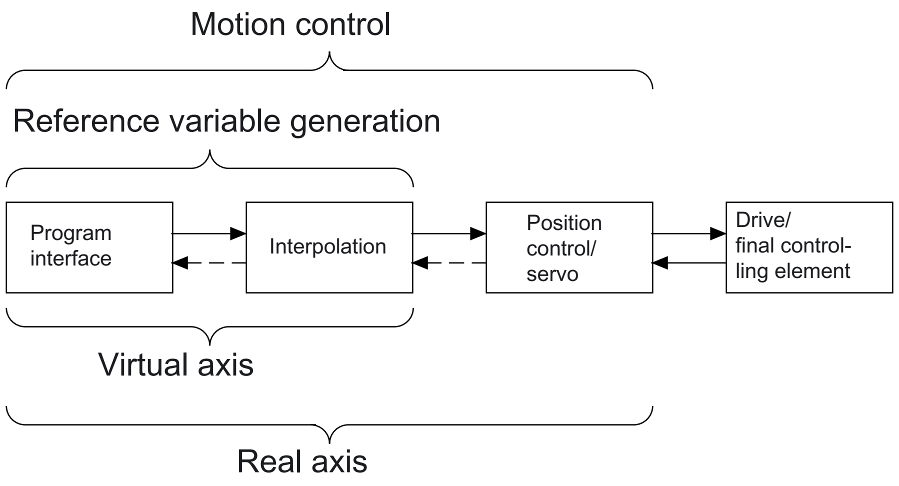


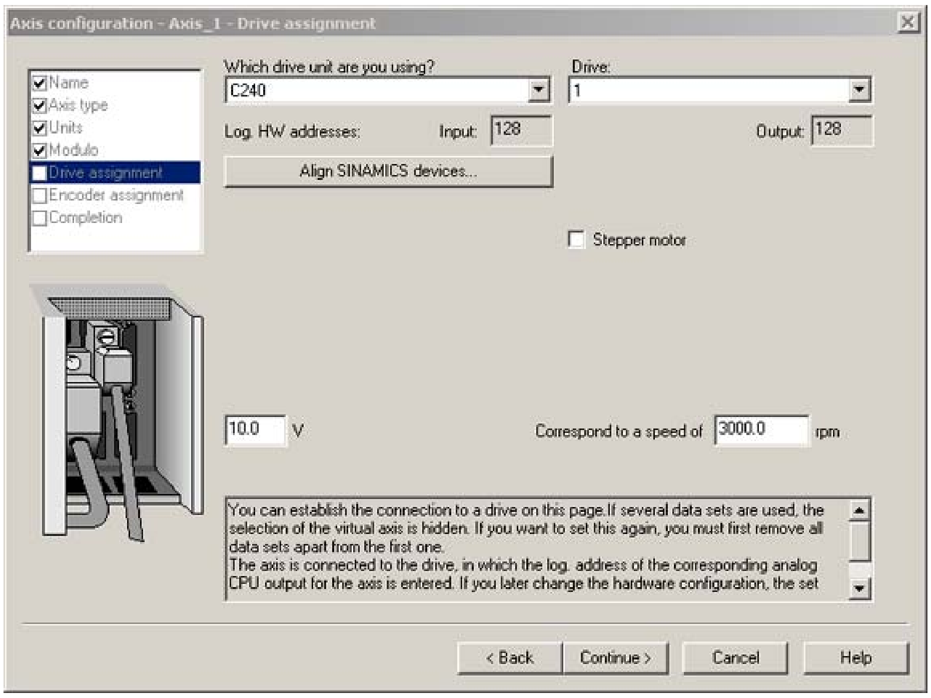
Figure 2-8 Difference between real and virtual axis (positioning axis example)

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).

2.4.3 Setting as a real axis with analog drive link



Which drive unit are you using? C240

Drive: 1

Log. HW addresses: Input: 128 Output: 128

☐ Stepper motor

10.0 V Correspond to a speed of 3000.0 rpm

You can establish the connection to a drive on this page.If several data sets are used, the selection of the virtual axis is hidden. If you want to set this again, you must first remove all data sets apart from the first one.
The axis is connected to the drive, in which the log. address of the corresponding analog CPU output for the axis is entered. If you later change the hardware configuration, the set

Figure 2-9 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.

The enabling 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

Setting as a real axis with digital drive coupling (Page 36)

2.4.4 Setting as a real axis with digital drive coupling

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 message frame 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 message frame, the maximum velocity is twice the reference velocity.

Control bits in STW1

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

The _enableAxis() and _disableAxis() commands for the axis TO simplify the use of the PROFIdrive state machine in accordance with the General State Diagram shown in the figure below.

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.

From V3.2 onwards, 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 status, in accordance with PROFIdrive, bit 0 of the initial status 0 for STW1 is required (pulse suppression and ready-to-start status). If STW1 bit 0 in the S1 status is not set to 0, for example as result of the enables using the single bit assignment or for the removal of not all enables for RELEASE_DISABLE alarm response, STW1 bit 0 using single bit assignment in the _disableAxis() command or using _disableAxis() with disableMode:=ALL must be set to 0. 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. In V3.2 and higher, 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 by the application

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 the 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 movement size 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, where the acknowledge can be made using SCOUT / HMI or a user program.
- The TO axis no longer performs any monitoring for drive failure / independent disable of the drive. Other monitoring remains active, e.g. encoder monitoring, operational time monitoring, 20005 drive error: 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 places to the control signals those last transferred 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.

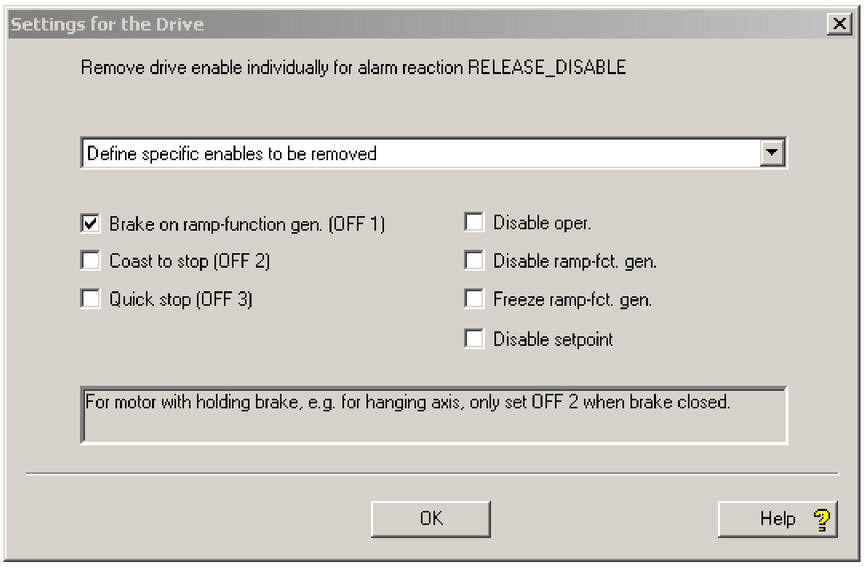
Brake Control

In V4.0 and higher, it is possible to cancel the enables specifically with the local alarm response RELEASE_DISABLE. When implementing a brake control in the drive, for example, OFF3 (STW1 bit 2) should be canceled first for **_disableAxis()** and for RELEASE_DISABLE. It is then possible to cancel the power (STW1 bit 1 cleared) when the brake is closed. For information on defining the reaction to technological alarms, refer also to Section Adjustable response to RELEASE_DISABLE (Page 312).

Information on brake control can also be found on the Utilities & Applications CD under "FAQs > Drives".

Removal of drive enables in a fault situation

For the local RELEASE_DISABLE alarm response, all enables to the drive will be removed for system setting. Before removing the enables in STW1 bit 0 - STW1 bit 3, in accordance with the general state diagram in SIMOTION (see figure below), ZSW1 bit 10 = 0 (n = 0 in the drive) will be awaited.
This system setting can be changed in the **driveControlConfig.releaseDisableMode** configuration data.


Settings for the Drive

Remove drive enable individually for alarm reaction RELEASE_DISABLE

Define specific enables to be removed

- [✓] Brake on ramp-function gen. (OFF 1)
- [ ] Coast to stop (OFF 2)
- [ ] Quick stop (OFF 3)
- [ ] Disable oper.
- [ ] Disable ramp-fct. gen.
- [ ] Freeze ramp-fct. gen.
- [ ] Disable setpoint

For motor with holding brake, e.g. for hanging axis, only set OFF 2 when brake closed.

Figure 2-10 Settings for the drive in SCOUT

You can open the **Settings for the Drive** dialog by selecting *Axis* - *Configuration* and the *Settings for the drive* button.

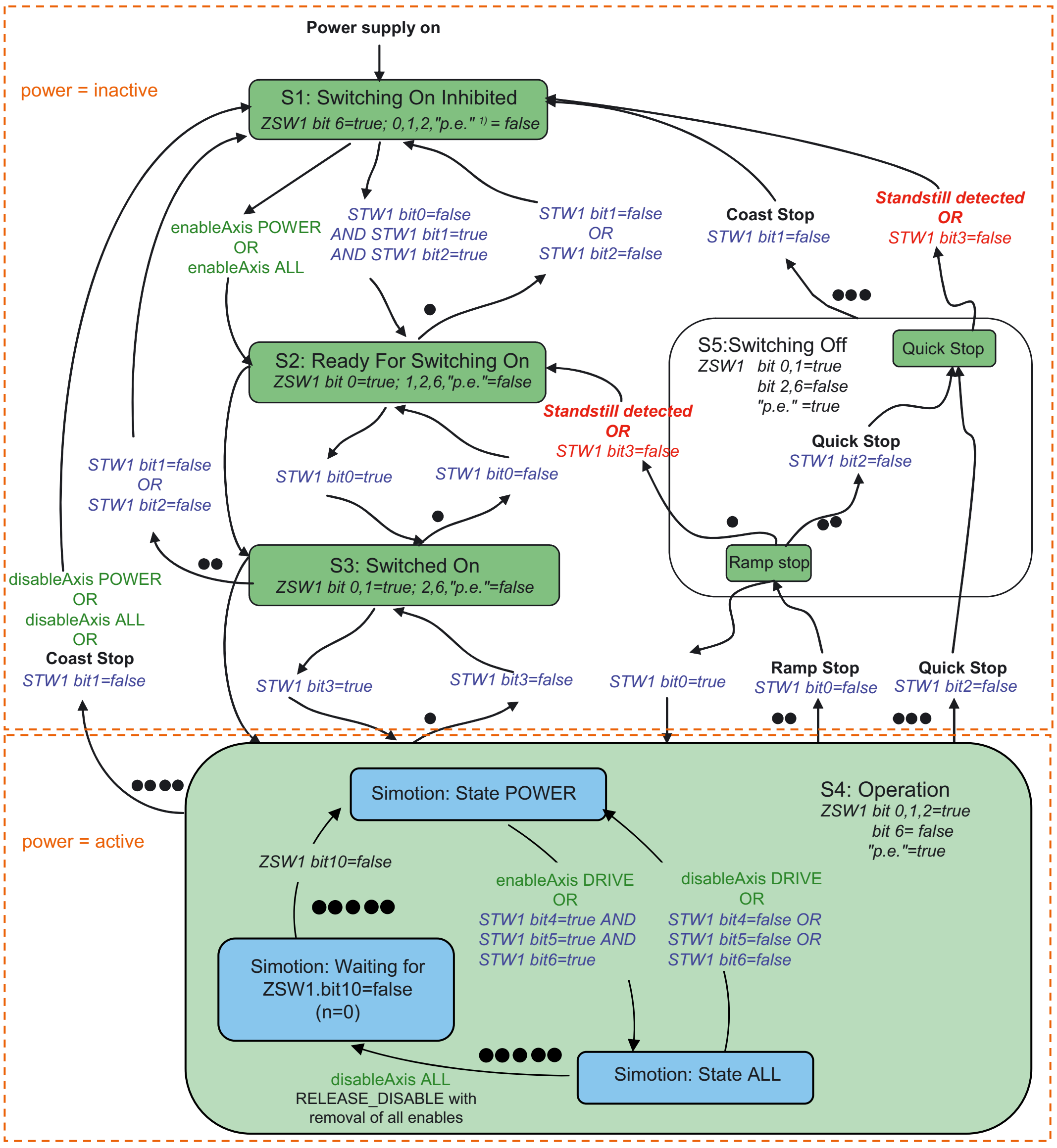
The bits selected in the screen form will be reset in the control word for the RELEASE_DISABLE local alarm response.

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.

Status diagram


Power supply on
power = inactive
S1: Switching On Inhibited
ZSW1 bit 6=true; 0,1,2,"p.e." 1) = false
enableAxis POWER OR enableAxis ALL
STW1 bit0=false AND STW1 bit1=true AND STW1 bit2=true
STW1 bit1=false OR STW1 bit2=false
Coast Stop STW1 bit1=false
Standstill detected OR STW1 bit3=false
S2: Ready For Switching On
ZSW1 bit 0=true; 1,2,6,"p.e."=false
Standstill detected OR STW1 bit3=false
S5:Switching Off
ZSW1 bit 0,1=true bit 2,6=false "p.e." =true
Quick Stop
Quick Stop STW1 bit2=false
Ramp stop
STW1 bit1=false OR STW1 bit2=false
STW1 bit0=true
STW1 bit0=false
S3: Switched On
ZSW1 bit 0,1=true; 2,6,"p.e."=false
disableAxis POWER OR disableAxis ALL OR Coast Stop STW1 bit1=false
STW1 bit3=true
STW1 bit3=false
STW1 bit0=true
Ramp Stop STW1 bit0=false
Quick Stop STW1 bit2=false
power = active
Simotion: State POWER
S4: Operation
ZSW1 bit 0,1,2=true bit 6= false "p.e."=true
ZSW1 bit10=false
enableAxis DRIVE OR STW1 bit4=true AND STW1 bit5=true AND STW1 bit6=true
disableAxis DRIVE OR STW1 bit4=false OR STW1 bit5=false OR STW1 bit6=false
Simotion: Waiting for ZSW1.bit10=false (n=0)
Simotion: State ALL
disableAxis ALL RELEASE_DISABLE with removal of all enables

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 event 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 TO
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 2-11 General state diagram in SIMOTION

Table 2- 8 Semantic of the STW1 bit 0 – STW1 bit 6 control word bits

STW1	Bit = 0 meaning	Bit = 1 meaning	Notes
Bit 0	OFF, Brake along the ramp generator (OFF1)	ON	Brake along the ramp generator deceleration ramp followed by pulse disable
Bit 1	Coast stop (OFF2)	No coast stop (no OFF2)	Pulse inhibit
Bit 2	Quick stop (OFF3)	No quick stop (no OFF3)	Brake along the OFF3 deceleration ramp followed by pulse disable
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 parameters manuals.

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.

Technologies and message frame types

Table 2- 9 Technologies and message frame types supported for real axis with digital drive link

Drive	Technology	Frame 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
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, 102, 103, 105, 106
SINAMICS integrated (SIMOTION D)	All	1 to 6, 102, 103, 105, 106
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.

Table 2- 10 Message frame types

Frame type	Product Brief
Standard message frames	
1	n-set interface, 16-bit, without encoder
2	n-set interface, 32-bit, without encoder
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
Siemens message frames	
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

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

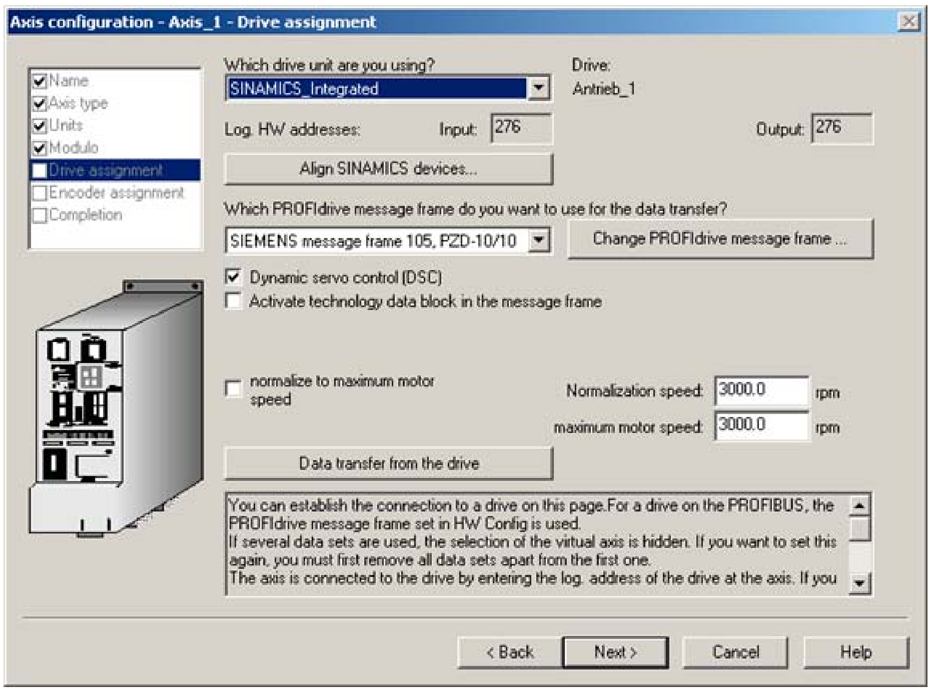
Separation of reference value and maximum value (as of V4.0)

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.


Axis configuration - Axis_1 - Drive assignment

[✓] Name
[✓] Axis type
[✓] Units
[✓] Modulo
[ ] Drive assignment
[ ] Encoder assignment
[ ] Completion

Which drive unit are you using? SINAMICS_Integrated
Drive: Antrieb_1
Log. HW addresses: Input: 276 Output: 276
Align SINAMICS devices...
Which PROFIdrive message frame do you want to use for the data transfer? SIEMENS message frame 105, PZD-10/10
Change PROFIdrive message frame ...
[✓] Dynamic servo control (DSC)
[ ] Activate technology data block in the message frame
[ ] normalize to maximum motor speed
Normalization speed: 3000.0 rpm
maximum motor speed: 3000.0 rpm
Data transfer from the drive

You can establish the connection to a drive on this page.For a drive on the PROFIBUS, the PROFIdrive message frame set in HW Config is used.
If several data sets are used, the selection of the virtual axis is hidden. If you want to set this again, you must first remove all data sets apart from the first one.
The axis is connected to the drive by entering the log. address of the drive at the axis. If you

< Back | Next > | Cancel | Help

Figure 2-12 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 Support of SINAMICS Safety Integrated Extended Functions on the Axis technology object (as of V4.1 SP1) (Page 201).

See also

Alarm reactions (Page 309)

Technology data (Page 159)

Support of SINAMICS Safety Integrated Extended Functions on the Axis technology object (as of V4.1 SP1) (Page 201)

2.4.5 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.

2.4.6 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

Setting as a real axis with digital drive coupling (Page 36)

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

With this setting, no actuator is connected via the Axis technology object. An angle signal is output directly to a third-party controller 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 124)

2.4.8 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 62)

2.4.9 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.

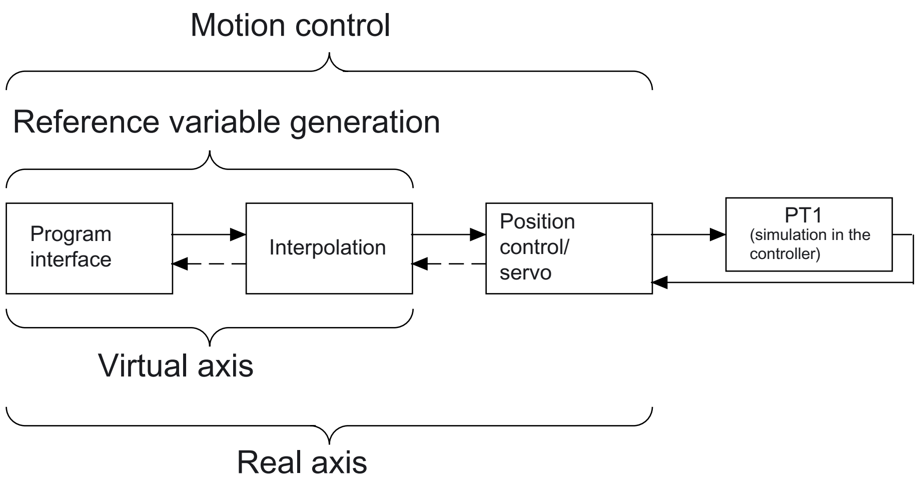


Figure 2-13 Axis simulation functional scheme

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.

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 2- 11 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 (V4.0 and higher)	Time constant T1
TypeOfAxis.NumberOfDataSet.DataSet_x.ProcessModel.T2 (V4.0 and higher)	Time constant T2

Note the following:

- 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).
- Output cam functions are executed.
- Homing to the encoder zero mark and the external zero mark is not possible in Axis simulation mode.
- 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 servo cycle in the controller/simulation driver, i.e., it may be necessary to cancel the gain and adjust the following error monitoring.

See also

Enabling/disabling program simulation (Page 298)

2.5 Encoders and encoder parameters

2.5.1 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

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 (data sets) can be generated on the axis. All generated encoders are internally active, and the measured values are updated cyclically.

The active encoders for closed-loop control and motion control can be switched via a data set. Encoder 1 is active by default.

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 changeover and encoder configuration, see Data set switchover / encoder switchover (Page 178).

See also

Data set overview (Page 178)

2.5.2 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 69)

2.5.3 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

2.5.4 Encoder assignment and terminology

The encoder type is specified with the encoder mode.

Table 2- 12 Definable encoder mode depending on the encoder type

	Encoder type		
Encoder mode	Absolute encoder	Cyclic absolute encoder	Incremental encoder
Endat (encoder data interface)	x	x	x
SSI (synchronous serial interface)	x	x	
Sinusoidal			x
Rectangle			x
Resolver		x1)	x
Analog encoder (value in the I/O area)	x		

1) Possible with single-pole-pair resolver only

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

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 message frame (for more information, see *Encoder interconnection via PROFIdrive message frame*).

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 2- 13 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 message frame (applies to SINAMICS, SIMODRIVE 611U and MASTERDRIVES)			
Incremental encoder	Rectangle	2,048	-
	Sinusoidal	2,048	-
	Resolver	2,048	-
	Endat	2,048	-
Absolute encoder	Endat	2,048	512
	SSI	1	1
...Cyclic absolute	Resolver (only pole pair number 1 possible)	2,048	512
	Endat	2,048	512
	SSI	1	1
PROFIBUS absolute encoder in PROFIdrive encoder message frame			
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 53)).

Default setting:

- Fine resolution (**~Encoder.~ResolutionMultiplierCyclic**) = 0
- Fine resolution of the absolute encoder in Gn_XIST2 (**AbsEncoder.absResolutionMultiplierAbsolute**) =0

See also

Encoder interface as a direct value in the I/O area (Page 58)

Encoder list (Page 53)

2.5.5 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 on the Utilities & Applications CD under "FAQs > Drives > Parameters of the connectable encoders" and in the online help (index search via Encoder parameter assignment).

2.5.6 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 53))

See also

Encoder list (Page 53)

2.5.7 Encoder interface using the PROFIdrive message frame

The encoder values are transmitted in the PROFIdrive message frame (see Table *Message frame 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 message frame.

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.

See also the Encoder list (Page 53).

Note

For SINAMICS drives, 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.

Encoder value via PROFIdrive axis message frames

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

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

Encoder value via PROFIdrive encoder message frame 8x

PROFIBUS/PROFINET encoders in accordance with the specification "Profile for DP-V2 Encoders Version 3.2" with message frame type 81 can be used. These encoders can be assigned freely. See also **PROFIBUS absolute encoder via PROFIdrive encoder message frame** in the next chapter.

Inconsistent configuration

If there are any differences between the configuration data in SIMOTION and the parameter settings for the encoder in the drive, the technological alarm

error 20005: Device type:2, log.address:1234 faulty. (Bit:0, reason: 0x80h)

is triggered as soon as an online connection is established between the controller and the drive/encoder.

For SINAMICS and SIMODRIVE, 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).

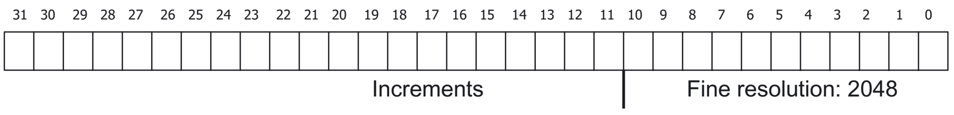


Figure 2-14 Example composition of the 32-bit encoder data of the cyclic actual value Gn_XIST1

Example of for an encoder with number of encoder pulses = 2,048 (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

2,048 (the default value depends on the encoder mode setting; see Table *Default settings for fine resolution in SIMOTION*).

SIMODRIVE 611U:

Table 2- 14 Settings

SIMOTION		611U	
Encoder pulses per revolution 1)	=2,048	P1007	=2,048
Fine resolution 2)	=0 (≡ 2,048)	P1042	=11

1) Inc/AbsEncoder.Inc/AbsResolution

2) Inc/AbsEncoder.Inc/AbsResolutionMultiplierCyclic

SINAMICS:

Table 2- 15 Settings

SIMOTION		SINAMICS	
Encoder pulses per revolution 1)	=2,048	P408	=2,048
Fine resolution 2)	=0 (≡ 2,048)	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 passed in Gn_XIST_2 (n = 1 or 2, number of encoder), they will be sent 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 = 2,048 (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 2- 16 Setting the encoder data

Encoder pulses per revolution [1)]	2,048
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 (= 2,048)

1) AbsEncoder.AbsResolution

2) AbsEncoder.absDataLength

3) AbsEncoder.absResolutionMultiplierAbsolute

4) AbsEncoder.AbsResolutionMultiplierCyclic

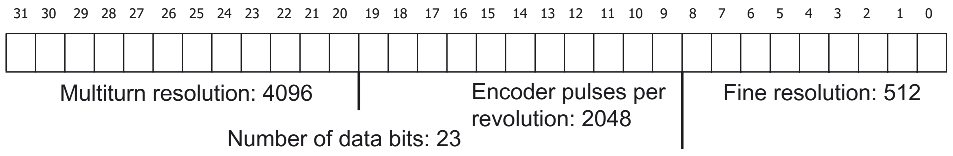


Figure 2-15 Example composition of the 32-bit encoder data of the absolute actual value Gn_XIST2

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 message frame

For SINAMICS and MASTERDRIVES, parameters are assigned for the number of pole pairs 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 = 2,048, 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 value encoder cyclical absolute, Endat, line count = 1, fine resolution = 2048, fine resolution absolute value = 2048, data width = 0)

See also the Encoder list (Page 53).

PROFIBUS absolute encoder via PROFIdrive encoder message frame

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

See also the Encoder list (Page 53).

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 4,096:
measuring steps per revolution = 4,096
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 in 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 message frame 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 53).

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*. You can find these documents on the *SIMOTION SCOUT Add-on* CD under *4_Additional_Documentation* in the document index.

See also

Setting as a real axis with digital drive coupling (Page 36)

Encoder list (Page 53)

2.5.8 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
  - 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)

  Note that the measured value in accordance with the justification specification will be mapped to an internal 32-bit wide signed data value of the DINT type and the mapped value then tested with the maximum values (see maximum limits below) 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.

  For the VALUE_RIGHT_MARGIN_WITHOUT_SIGN setting, the maximum permitted encoder resolution or measured value width is 31 bits.

  For the VALUE_LEFT_MARGIN setting, the measured value for the setting of the encoder resolution of measured value width is

  - Mapped < 16 bits left-justified to an internal 16-bit wide data value, namely, to the least-significant byte 1 and byte 2 of the internal data value of the DINT type. The missing bits 15 minus the measured value width are right-padded with zeros and the most-significant byte 3 and byte 4 of the internal data value padded appropriately in accordance with the sign.
  - Mapped ≥ 16 bits left-justified to the 32-bit wide data value and the missing 31 bits minus the measured value width right-padded with zeros.

For the VALUE_LEFT_MARGIN_WITHOUT_SIGN setting, the measured value for the setting of the encoder resolution of measured value width

– Mapped ≤ 16 bits left-justified to an internal 16-bit wide data value, namely, to the least-significant byte 1 and byte 2 of the internal data value of the DINT type. The missing 16 bits minus the measured value width are right-padded with zeros and the most-significant byte 3 and byte 4 filled with zeros.
– > Mapped > 16 bits left-justified to the 32-bit wide value and the missing 32 bits minus the measured value width right-padded with zeros.

  Because this mapped measured value is tested with the maximum limits of DINT data type, only the zero value is possible for the VALUE_LEFT_MARGIN_WITHOUT_SIGN setting and a measured value width > 16 bits as value of the most-significant bit in the measured value. This means the measuring range is limited to maximum 50% of the measured value width.

- 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.

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

The following bits can be configured for the direct value as absolute value in the I/O area (as of V4.1 SP1):

- **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 controller.

The controller supports the following options:

- The peripheral module displays the new measured value in an update bit/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)

2.5.9 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.

2.5.10 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 256).

See also

Setting a non-exclusive drive assignment (as of V4.1 SP1) (Page 44)

2.5.11 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.).

2.6 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.**

2.7 Setting for axis and encoder mechanics

2.7.1 Overview of setting options for axis and encoder mechanics

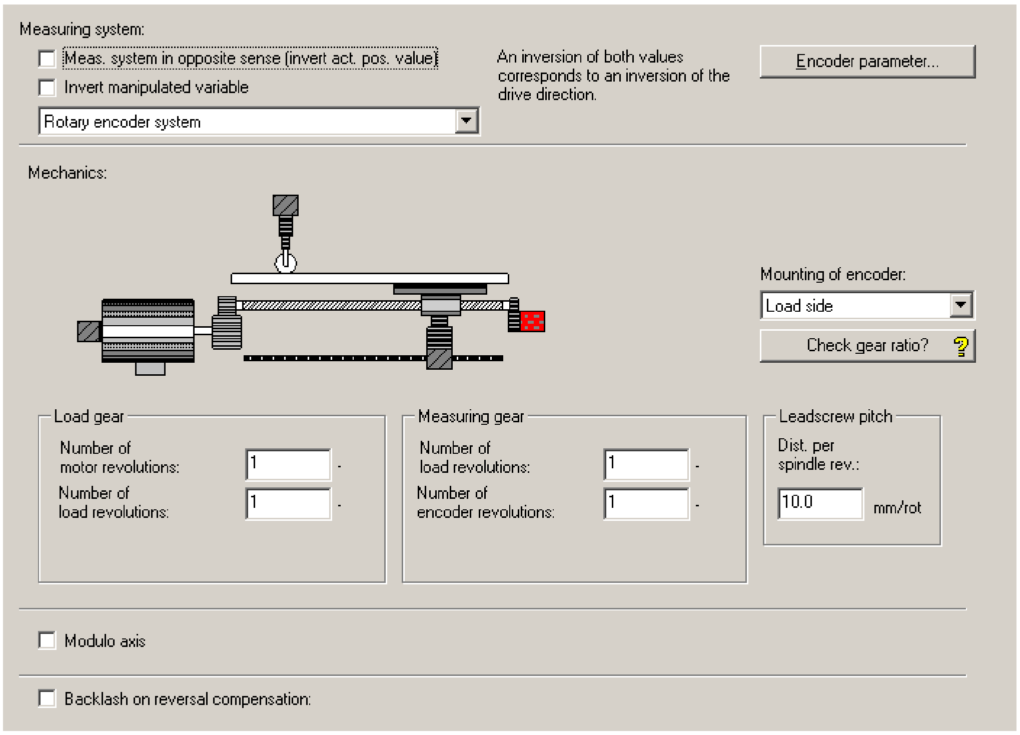
The following options are available on the drive side of a real axis:

- 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)



Measuring system:

☐ Meas. system in opposite sense (invert act. pos. value)

☐ Invert manipulated variable

An inversion of both values corresponds to an inversion of the drive direction.

Rotary encoder system

Mechanics:

Mounting of encoder: Load side

Load gear		Measuring gear		Leadscrew pitch	
Number of motor revolutions:	1 -	Number of load revolutions:	1 -	Dist. per spindle rev.:	10.0 mm/rot
Number of load revolutions:	1 -	Number of encoder revolutions:	1 -		

☐ Modulo axis

☐ Backlash on reversal compensation:

Figure 2-16 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 in the axis wizard or using the experts list. (see SIMOTION SCOUT online help)

2.7.2 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).

2.7.3 Boundary conditions for mechanics settings for modulo axes (long-term stability)

With modulo axes, a check is performed to determine whether the long-term stability is guaranteed. Long-term stability ensures that the positions measured by the encoder and the internal representation of the actual values are always synchronous. This enables positions to be approached exactly, even after any number of modulo overflows. If long-term stability cannot be guaranteed, 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 stability 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 inappropriate selection of values in the configuration data. These values must satisfy the following conditions:

Calculating f_1 for a modulo rotary axis

f_1 = measuring gear numerator x 360 x internal resolution x load revolutions 1)

Calculation of $f1$ for a modulo linear axis

f_0 = TRUNC(leadscrew pitch x internal resolution) 2)
The decimal places of f_0 are truncated. The leadscrew pitch must be specified in the base unit of mm.

f_1 = measuring gear numerator x f_0 x load revolutions 1)

Calculating f_2

f_2 = measuring gear denominator x encoder resolution x actual value factor x motor revolutions 1)

Calculating f_{11} and f_{22}

From f_1 and f_2, the greatest common divisor **k** must be determined and used in the following formula:

$f_{11} = (f_1 / k) < 2^{31}$

$f_{22} = (f_2 / k) < 2^{31}$

The results of f_{11} and f_{22} must be less than 2^{31}. If this is not the case, check whether an appropriate modification of the parameters in the f_1 and f_2 formulas will produce values for f_{11} and f_{22} that do not exceed the maximum permissible value.

Calculating f_{31}

If f_{11} and f_{22} meet the requirements described above, and the download operation is still aborted with a **Configuration error 20006 reason 3041** message, you can perform the following tests:

f_3 = TRUNC(modulo length · internal resolution)

The decimal places of f_3 are truncated. The modulo length must be specified in mm for linear axes and degrees for rotary axes.

Now you can calculate the greatest common divisor k_2 of f_3 and f_{11} and insert it in the following formula:

$f_{31} = (f_3 \times f_{22} \times f_{11}) / (k_2 \times k_2) < 2^{62}$

At this point you have to check whether f_{31} is less than 2^{62}. If not, check whether you can reduce modulo length. You can also make changes in the parameters in the f_1 and f_2 formulas, as long as you ensure that the requirements for f_{11} and f_{22} are still met.

1)	For load-side or external encoder mounting, insert 1 in the formula for motor revolutions and load revolutions.
2)	For external encoder mounting, insert the configured distance per revolution in place of leadscrew pitch.

Table 2- 17 Description of parameters

Parameter	Comment / configuration data element on the axis
Measuring gear numerator	Numerator of the measuring gear ratio TypeOfAxis.NumberOfEncoders.Encoder_1.AdaptDrive.numFactor TypeOfAxis.NumberOfEncoders.Encoder_1.AdaptExtern.numFactor TypeOfAxis.NumberOfEncoders.Encoder_1.AdaptLoad.numFactor
Measuring gear denominator	Denominator of the measuring gear ratio TypeOfAxis.NumberOfEncoders.Encoder_1.AdaptDrive.denFactor TypeOfAxis.NumberOfEncoders.Encoder_1.AdaptExtern.denFactor TypeOfAxis.NumberOfEncoders.Encoder_1.AdaptLoad.denFactor
Motor revolutions	Numerator for motor revolutions TypeOfAxis.NumberOfDataSets.DataSet_1.Gear.numFactor
Load revolutions	Numerator for load revolutions TypeOfAxis.NumberOfDataSets.DataSet_1.Gear.denFactor
Internal resolution	Internal increments / position unit Defined in the configuration using the axis wizard.
Encoder resolution	Encoder pulses per revolution (specified on the encoder) TypeOfAxis.NumberOfEncoders.Encoder_1.IncEncoder.incResolution TypeOfAxis.NumberOfEncoders.Encoder_1.AbsEncoder.absResolution
Multiplication factor, actual value (actual value factor)	= 1 for onboard SSI encoder = 4 for onboard incremental encoders = X [1] IncEncoder.incResolution MultiplierCyclic TypeOfAxis.NumberOfEncoders.Encoder_1.IncEncoder.incResolution MultiplierCyclic TypeOfAxis.NumberOfEncoders.Encoder_1.AbsEncoder.absResolution MultiplierCyclic
Modulo length	Modulo.length
Leadscrew pitch for each revolution of the axis	LeadScrew.pitchVal
Path per revolution	TypeOfAxis.NumberOfEncoders.Encoder_1.pathPerResolution.length

[1] X=0:For the fine resolution, take into account the default value according to the Default settings for fine resolution in SIMOTION table in the chapter titled Encoder interface using the PROFIdrive message frame (encoder in PROFIdrive axis message frame/fine resolution (Gn_XIST1)). (In this case, the test of long-term stability is performed only on the target device. It is not tested during the consistency check in the engineering system.)

X<>0: Use the configured value for the fine resolution.

Note

For more information, refer to the Utilities & Applications CD under FAQs.

See also

Encoder interface using the PROFIdrive message frame (Page 53)

2.8 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 2- 18 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 137)

2.9 Homing

2.9.1 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 81).

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 281)

2.9.2 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.

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.)

Homing 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.

2.9.3 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/setting the 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.

2.9.4 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 homing 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.

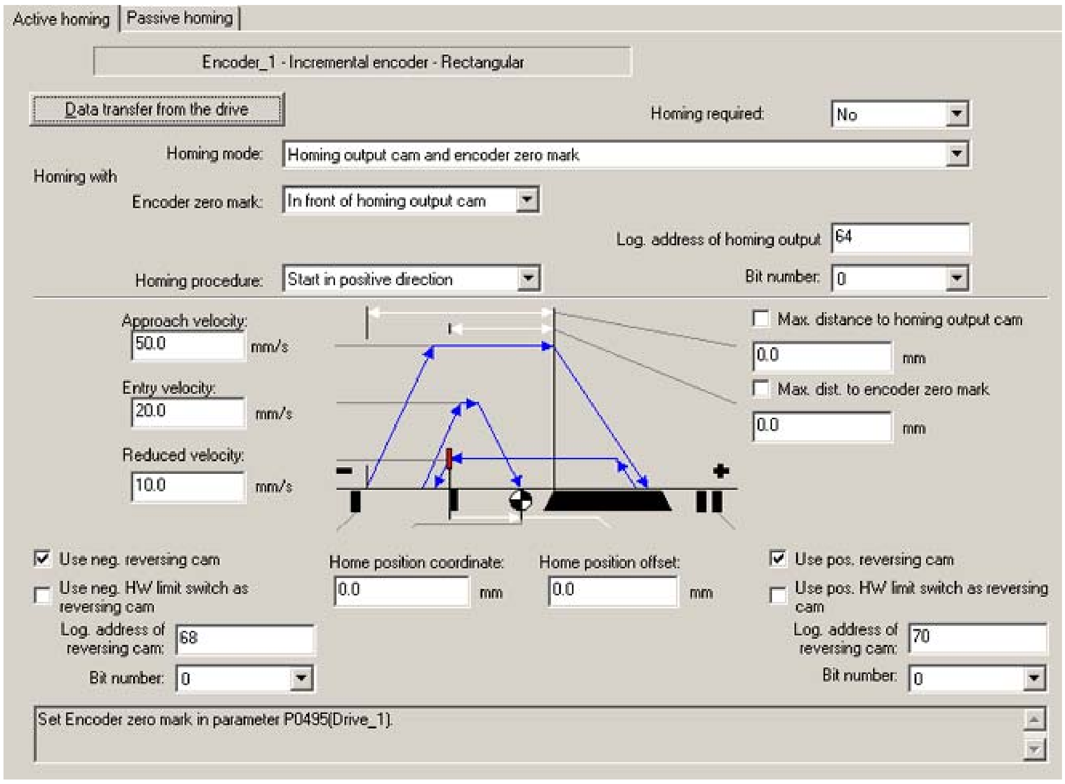
The axis has synchronized or homed status once the homing mark is detected.

Note

The direction of approach to the homing output cam or homing 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 homing output cam and encoder zero mark mode

The homing command causes the axis to move toward the homing output cam. After the axis has detected and left the homing output 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 homing output 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.



Encoder_1 - Incremental encoder - Rectangular

Homing required:	No
Homing mode:	Homing output cam and encoder zero mark
Encoder zero mark:	In front of homing output cam
Log. address of homing output	64
Homing procedure:	Start in positive direction
Bit number:	0
Approach velocity:	50.0 mm/s
Entry velocity:	20.0 mm/s
Reduced velocity:	10.0 mm/s
☐ Max. distance to homing output cam	0.0 mm
☐ Max. dist. to encoder zero mark	0.0 mm
☑ Use neg. reversing cam	
☐ Use neg. HW limit switch as reversing cam	
Log. address of reversing cam:	68
Bit number:	0
Home position coordinate:	0.0 mm
Home position offset:	0.0 mm
☑ Use pos. reversing cam	
☐ Use pos. HW limit switch as reversing cam	
Log. address of reversing cam:	70
Bit number:	0

Set Encoder zero mark in parameter P0495(Drive_1).

Figure 2-17 Parameters for active homing with homing output cam and encoder zero mark

The sequence can be divided into three phases:

- Phase 1

  Travel to the homing output cam

  The axis moves to the homing output 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 homing output 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**.

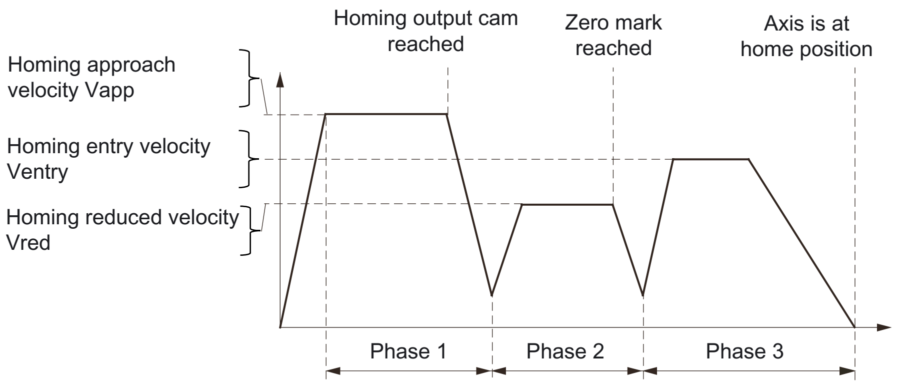


Figure 2-18 Homing with incremental measuring systems

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

The hardware limit switches can also be defined as a homing output 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 homing output cam and encoder zero mark, whereby the encoder zero mark lies in front of the homing output 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 message frame of the drive TO, 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 message frame 390 or 391 of the CU is transferred to the controller.

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 negative edges are permitted in the positive traversing direction, and only positive edges are permitted in the negative traversing direction, meaning that the same output cam side is always used.

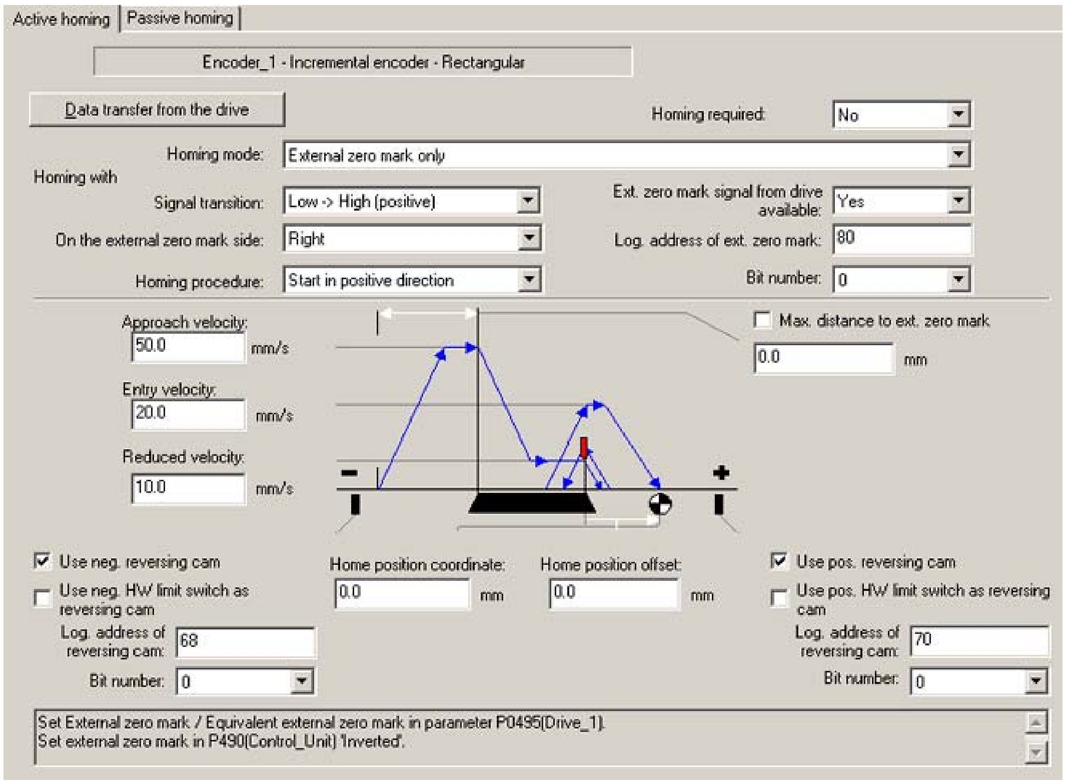
Inverting is not possible with SIMODRIVE 611U. The edges can be inverted by using initiators of the inverted type or via an application (inaccurate).

MASTERDRIVES

With MASTERDRIVES Motion Control, only positive edges are permitted.

OnBoard C2x0

With OnBoard, only positive edges are permitted.


Active homing | Passive homing

Encoder_1 - Incremental encoder - Rectangular

Data transfer from the drive

Homing required: No

Homing with

Homing mode: External zero mark only

Signal transition: Low -> High (positive)

Ext. zero mark signal from drive available: Yes

On the external zero mark side: Right

Log. address of ext. zero mark: 80

Homing procedure: Start in positive direction

Bit number: 0

Approach velocity: 50.0 mm/s

Entry velocity: 20.0 mm/s

Reduced velocity: 10.0 mm/s

☐ Max. distance to ext. zero mark: 0.0 mm

☑ Use neg. reversing cam

☐ Use neg. HW limit switch as reversing cam

Log. address of reversing cam: 68

Bit number: 0

Home position coordinate: 0.0 mm

Home position offset: 0.0 mm

☑ Use pos. reversing cam

☐ Use pos. HW limit switch as reversing cam

Log. address of reversing cam: 70

Bit number: 0

Set External zero mark / Equivalent external zero mark in parameter P0495(Drive_1).
Set external zero mark in P490(Control_Unit) 'Inverted'.

Figure 2-19 Parameters for active homing with external zero mark only

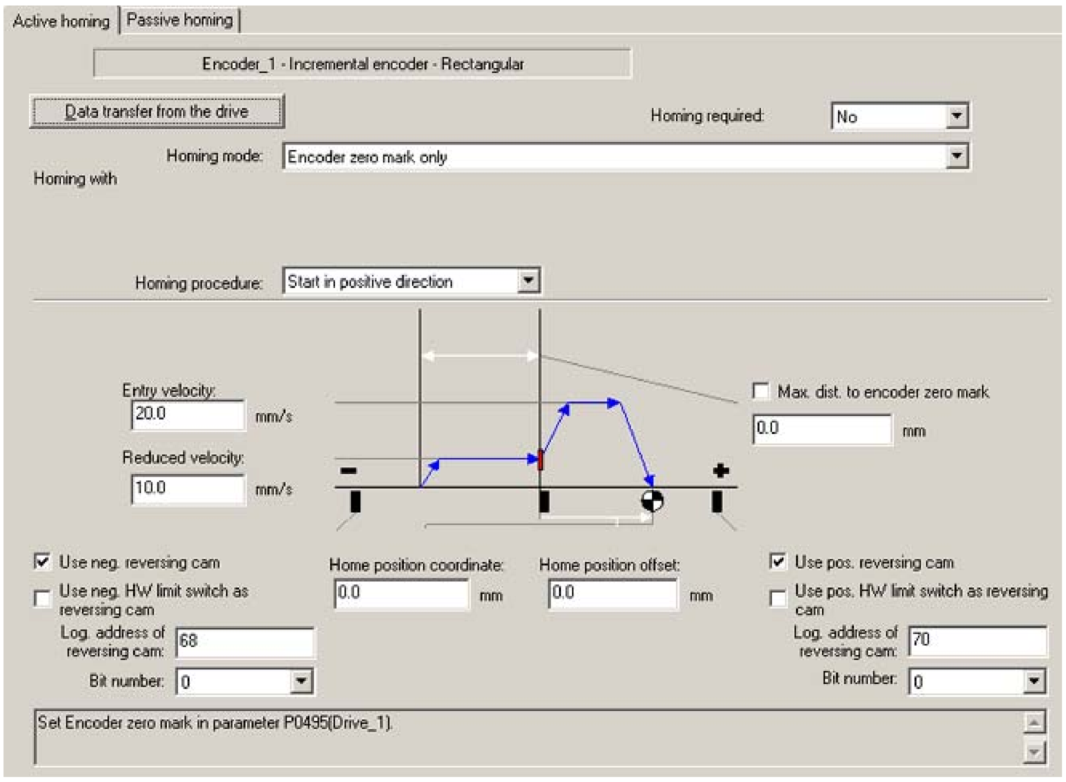
Note

- Only homing with a homing output 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 homing output cam)

Homing without a homing output 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.



Encoder_1 - Incremental encoder - Rectangular

Homing required: No

Homing mode: Encoder zero mark only

Homing procedure: Start in positive direction

Entry velocity: 20.0 mm/s

Reduced velocity: 10.0 mm/s

☐ Max. dist. to encoder zero mark: 0.0 mm

☑ Use neg. reversing cam

☐ Use neg. HW limit switch as reversing cam

Log. address of reversing cam: 68

Bit number: 0

Home position coordinate: 0.0 mm

Home position offset: 0.0 mm

☑ Use pos. reversing cam

☐ Use pos. HW limit switch as reversing cam

Log. address of reversing cam: 70

Bit number: 0

Set Encoder zero mark in parameter P0495(Drive_1).

Figure 2-20 Parameters for active homing with encoder zero mark only

---

Note

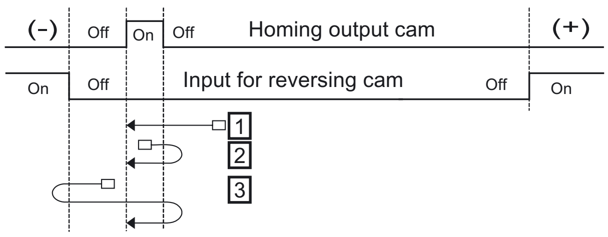
Only homing with a homing output cam and an encoder zero mark should be considered for a resolver with a pole pair number > 1.

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 homing output cam.



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

1 Starting point in front of homing output cam

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

3 Axis is behind the homing output cam on the left
A normal homing operation with a Left-hand approach direction is begun. The axis reverses direction at the reversing cam and travels against the approach direction until it is over the homing output cam. This is followed by normal homing.

Figure 2-21 Reversing cams for homing

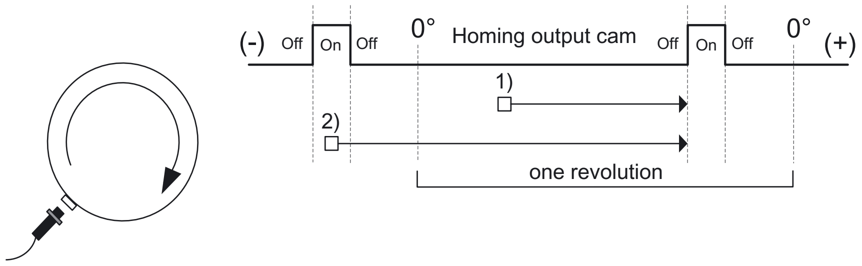
The hardware limit switches can also be defined as reversing cams. In this case, the hardware limit switch is deactivated during homing.

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 homing output cam

Homing output cam is found in the modulo area.

2) Axis is on homing output cam

This is detected by the system. The axis travels in the programmed approach direction to the next homing output cam, even if the modulo area is overrun.

Figure 2-22 Example of homing in one direction only

2.9.5 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.

2.9.6 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.

See also

Positioning (Page 284)

Synchronization/homing with incremental encoders (Page 332)

2.9.7 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.

2.9.8 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
- 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

2.9.9 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.

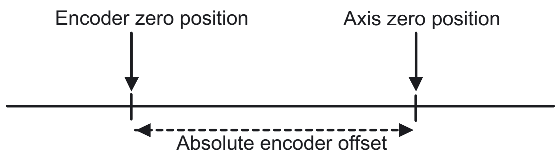


Figure 2-23 The axis zero position is the encoder zero position plus the absolute encoder offset

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 controller is commissioned.

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

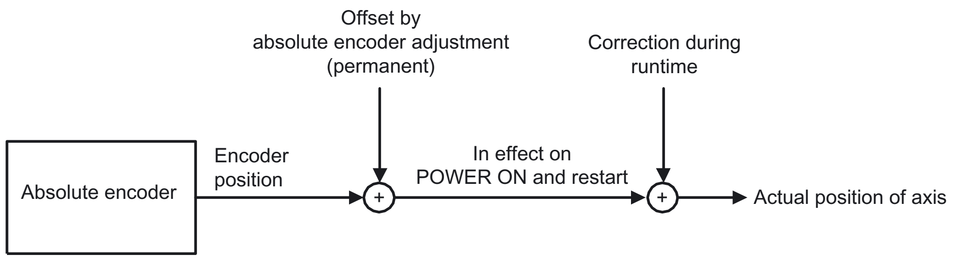


Figure 2-24 Incorporating the absolute encoder offset

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	→	Position = 105.000
_homing() command called for the second time	→	Position = 105.000
absshift = 7.000		
_homing() command called for the third time	→	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. (The **absHomingEncoder.absshift** configuration data element 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 controller, the stored offset is no longer available.

  If the controller 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 48)

2.9.10 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.

2.9.11 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.

2.9.12 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.

2.9.13 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.

2.9.14 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 296)

2.9.15 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 217)

2.10 Monitoring/limiting functions

2.10.1 Overview of monitoring/limiting functions (block diagram)

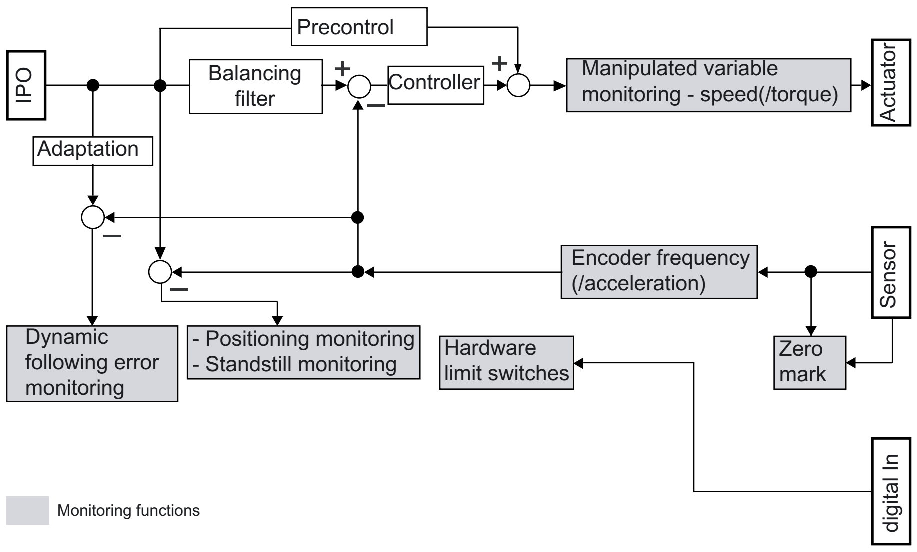


Figure 2-25 Block diagram of positioning axis monitoring functions

2.10.2 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.

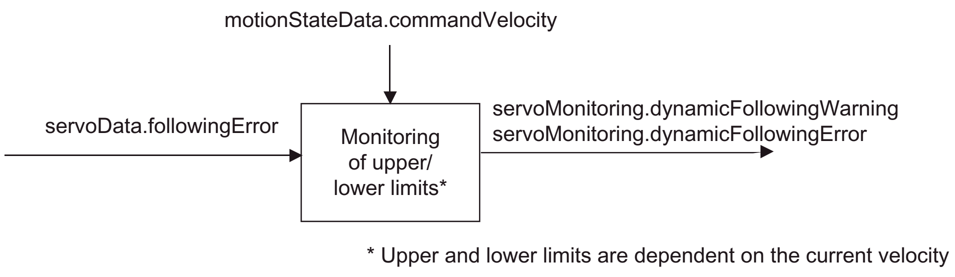
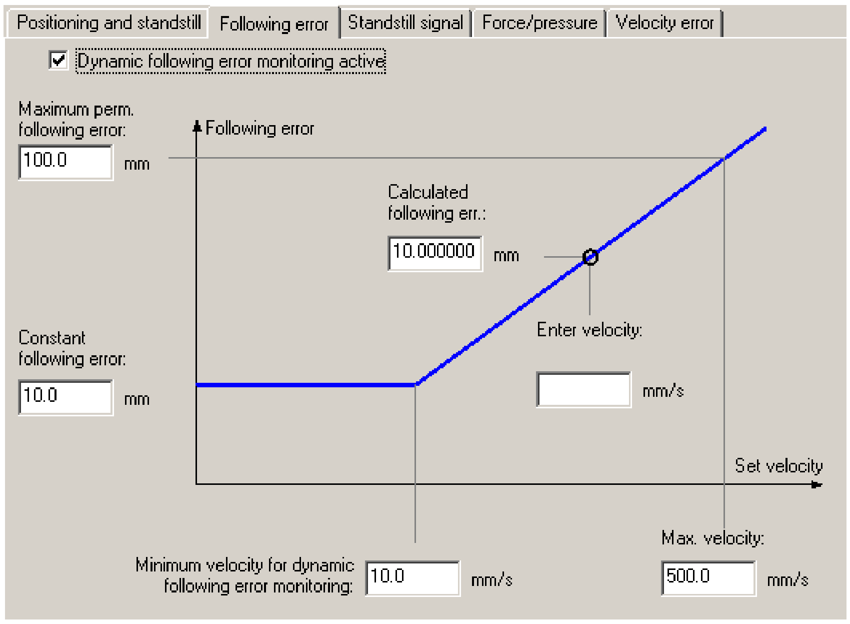

motionStateData.commandVelocity
servoData.followingError
Monitoring of upper/ lower limits*
servoMonitoring.dynamicFollowingWarning
servoMonitoring.dynamicFollowingError
* Upper and lower limits are dependent on the current velocity

Figure 2-26 Dynamic monitoring of following errors



☑ Dynamic following error monitoring active

Maximum perm. following error: 100.0 mm

Calculated following err.: 10.000000 mm

Enter velocity: mm/s

Constant following error: 10.0 mm

Minimum velocity for dynamic following error monitoring: 10.0 mm/s

Max. velocity: 500.0 mm/s

Figure 2-27 Function and parameters for following error monitoring

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 controller.

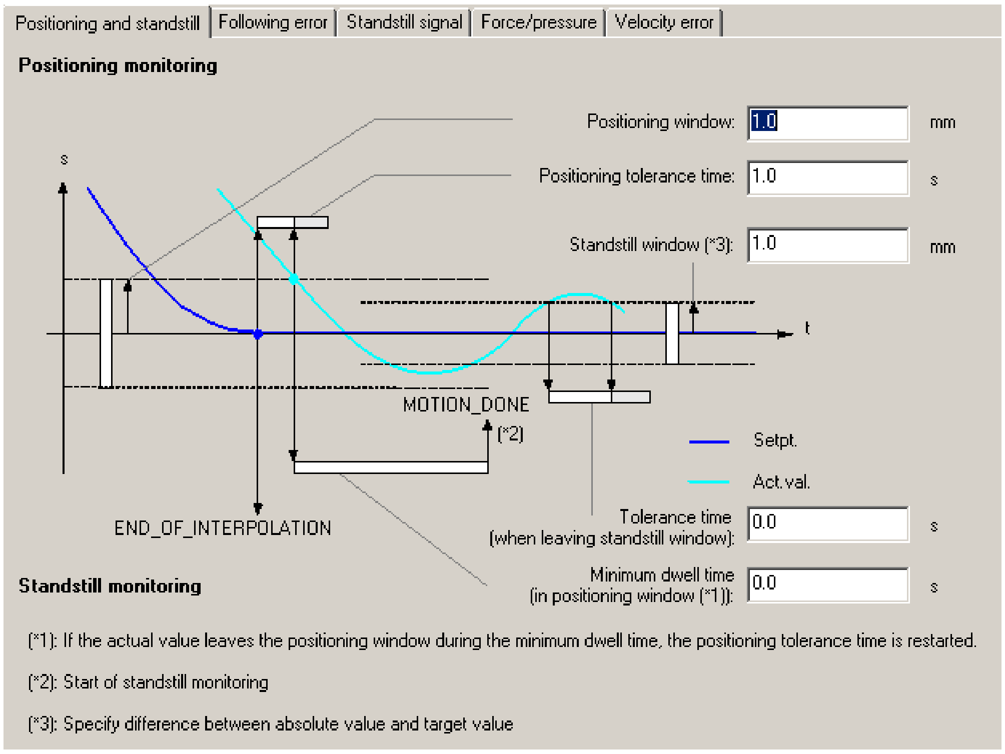
Therefore, the transfer times of the setpoint to the drive and the actual value to the controller are included in the following error.

Following error determination with DSC

The following error is determined from the difference between the non-symmetrical setpoint delayed by 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 controller.

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

2.10.3 Positioning and standstill monitoring



Positioning monitoring

Positioning window: 1.0 mm

Positioning tolerance time: 1.0 s

Standstill window (*3): 1.0 mm

Tolerance time (when leaving standstill window): 0.0 s

Standstill monitoring

Minimum dwell time (in positioning window (*1)): 0.0 s

(*1): If the actual value leaves the positioning window during the minimum dwell time, the positioning tolerance time is restarted.

(*2): Start of standstill monitoring

(*3): Specify difference between absolute value and target value

Figure 2-28 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.

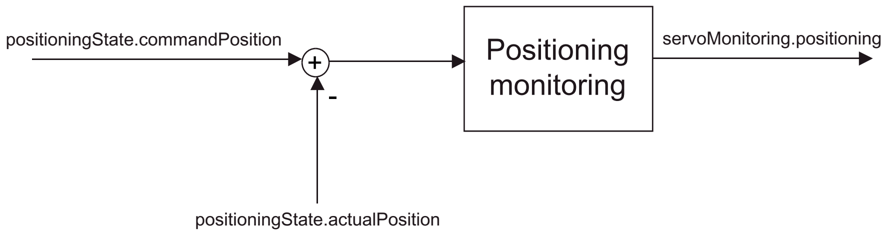

positioningState.commandPosition
+
-
Positioning monitoring
servoMonitoring.positioning
positioningState.actualPosition

Figure 2-29 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.

2.10.4 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.

If the WHEN_MOTION_DONE command step enabling condition is set in speed-controlled mode for the drive axis and positioning axis, the command is ended when the standstill signal changes from INACTIVE to ACTIVE. The completion of commands with this setting for position-controlled motion is described in **Positioning and standstill monitoring**.

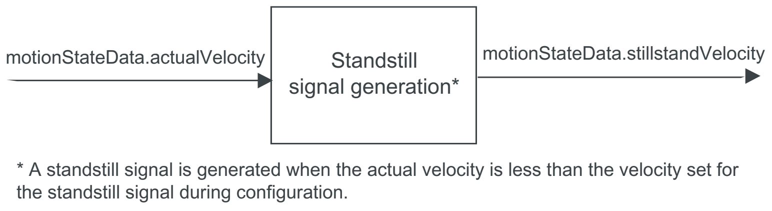
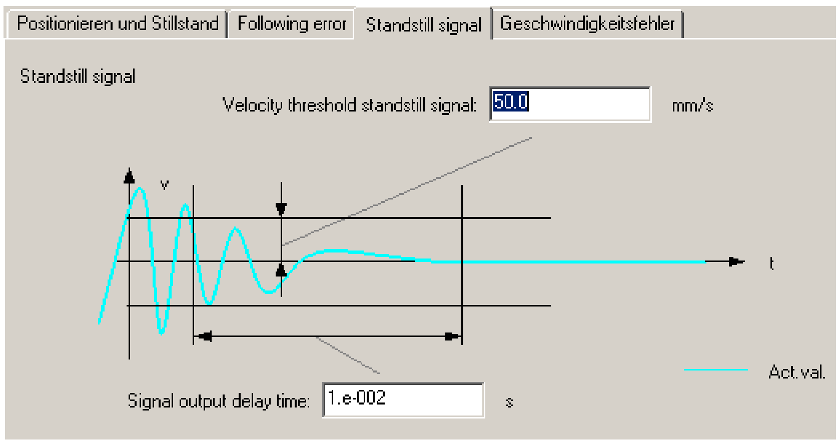

motionStateData.actualVelocity
Standstill signal generation*
motionStateData.stillstandVelocity
* A standstill signal is generated when the actual velocity is less than the velocity set for the standstill signal during configuration.

Figure 2-30 Standstill signal generation

The standstill signal is available on the positioning axis and the drive axis.



Velocity threshold standstill signal: 50.0 mm/s

Signal output delay time: 1.e-002 s

Figure 2-31 Function and adjustable parameters for the standstill signal

See also

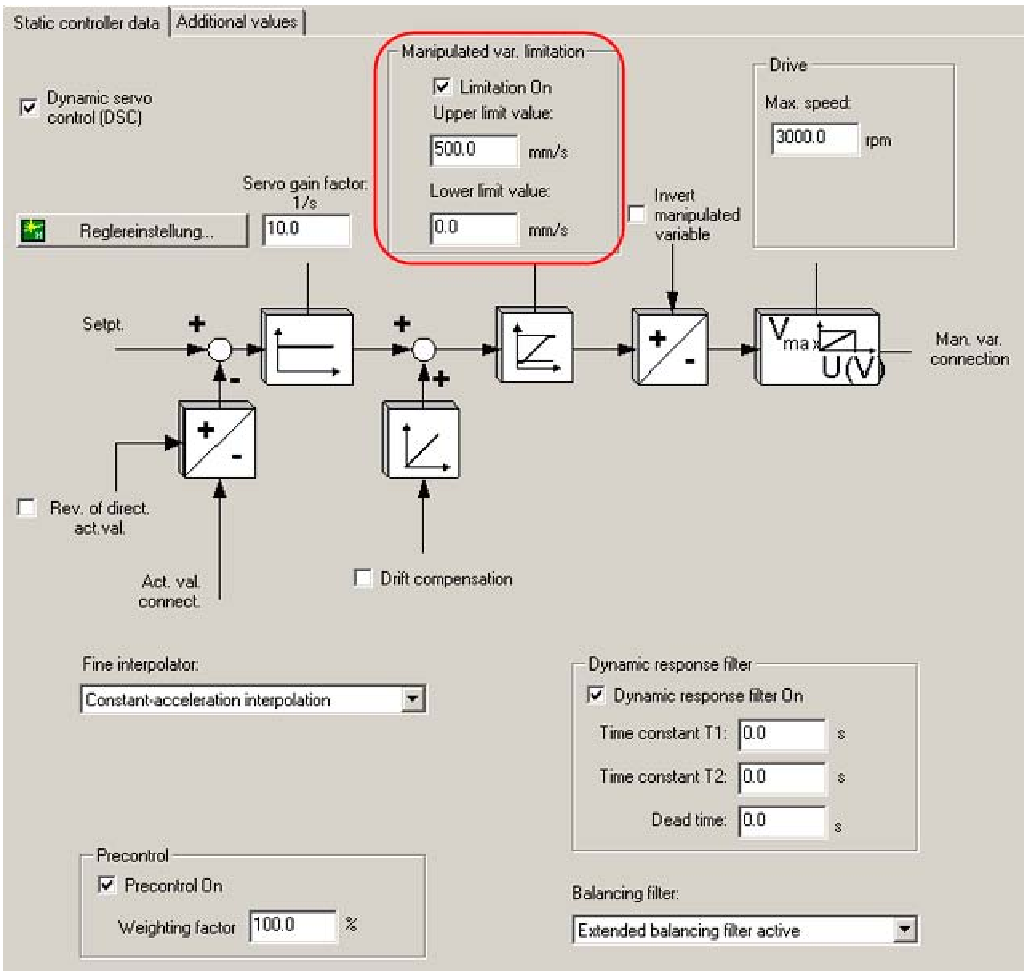
Positioning and standstill monitoring (Page 88)

2.10.5 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.

The maximum possible acceleration is monitored along with the maximum torque by monitoring the rise of the manipulated variable.

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



Manipulated var. limitation
[✓] Limitation On
Upper limit value: 500.0 mm/s
Lower limit value: 0.0 mm/s

Drive
Max. speed: 3000.0 rpm

[✓] Dynamic servo control (DSC)
Servo gain factor: 10.0 1/s
[ ] Invert manipulated variable
[ ] Rev. of direct. act.val.
[ ] Drift compensation

Fine interpolator: Constant-acceleration interpolation

Dynamic response filter
[✓] Dynamic response filter On
Time constant T1: 0.0 s
Time constant T2: 0.0 s
Dead time: 0.0 s

Precontrol
[✓] Precontrol On
Weighting factor 100.0 %

Balancing filter: Extended balancing filter active

Figure 2-32 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.

2.10.7 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.

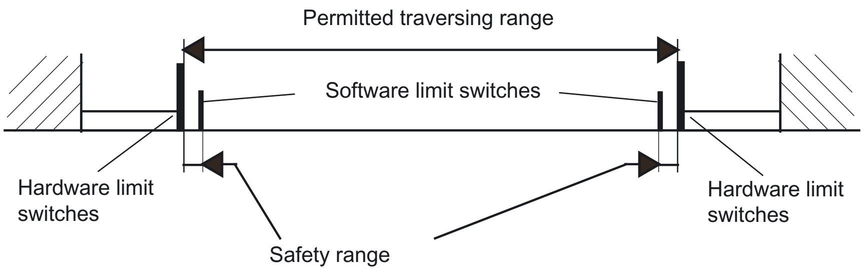


Figure 2-33 Monitoring of the permissible traversing range by limit switches

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 the 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

Once the limit switch has been overtraveled, the control 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 control is switched off, the information relating to the limit switch polarity is lost. The axis must then be moved into the permissible range by the user.

When the control is switched on, the axis must be positioned within the permissible traversing range.

If the hardware limit switches are configured as reversing cams or homing output 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.

2.10.8 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.

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.

Tolerance window for the retraction

If the axis overtravels the software limit switch while position control is inactive (alarm 40107, *Software limit switch overtraveled*), a tolerance window within which the axis can be retracted can be defined using the **relieveWindow** configuration data element. This means that jittering of the actual value will not cause the software limit switch to respond again during retraction.

2.10.9 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.

2.10.10 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.

2.10.11 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.

2.11 Positioning axis with position control

2.11.1 Overview of positioning axis with position control

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

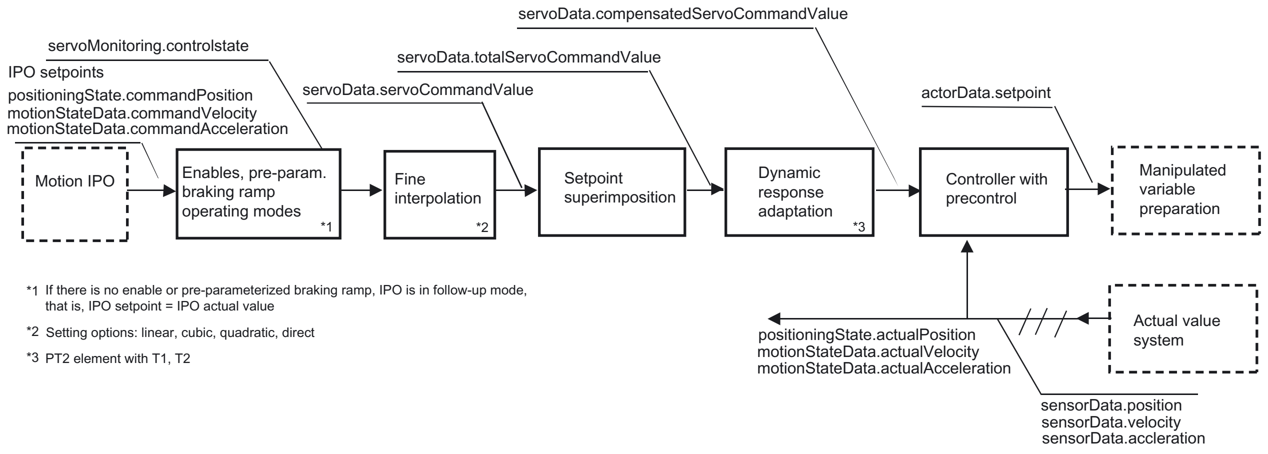

servoData.compensatedServoCommandValue
servoMonitoring.controlstate
servoData.totalServoCommandValue
IPO setpoints
positioningState.commandPosition
motionStateData.commandVelocity
motionStateData.commandAcceleration
servoData.servoCommandValue
actorData.setpoint
Motion IPO
Enables, pre-param. braking ramp operating modes
*1
Fine interpolation
*2
Setpoint superimposition
Dynamic response adaptation
*3
Controller with precontrol
Manipulated variable preparation
*1 If there is no enable or pre-parameterized braking ramp, IPO is in follow-up mode, that is, IPO setpoint = IPO actual value
*2 Setting options: linear, cubic, quadratic, direct
*3 PT2 element with T1, T2
positioningState.actualPosition
motionStateData.actualVelocity
motionStateData.actualAcceleration
Actual value system
sensorData.position
sensorData.velocity
sensorData.accleration

Figure 2-34 Overview of positioning axis with position control

---

Note

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

2.11.2 Position control

When position control is active, controllers, monitoring, and compensation are active. There are modes in which monitoring are deactivated, e.g., the position-related monitoring functions for torque or pressure limiting.

All compensation functions can be enabled/disabled.

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

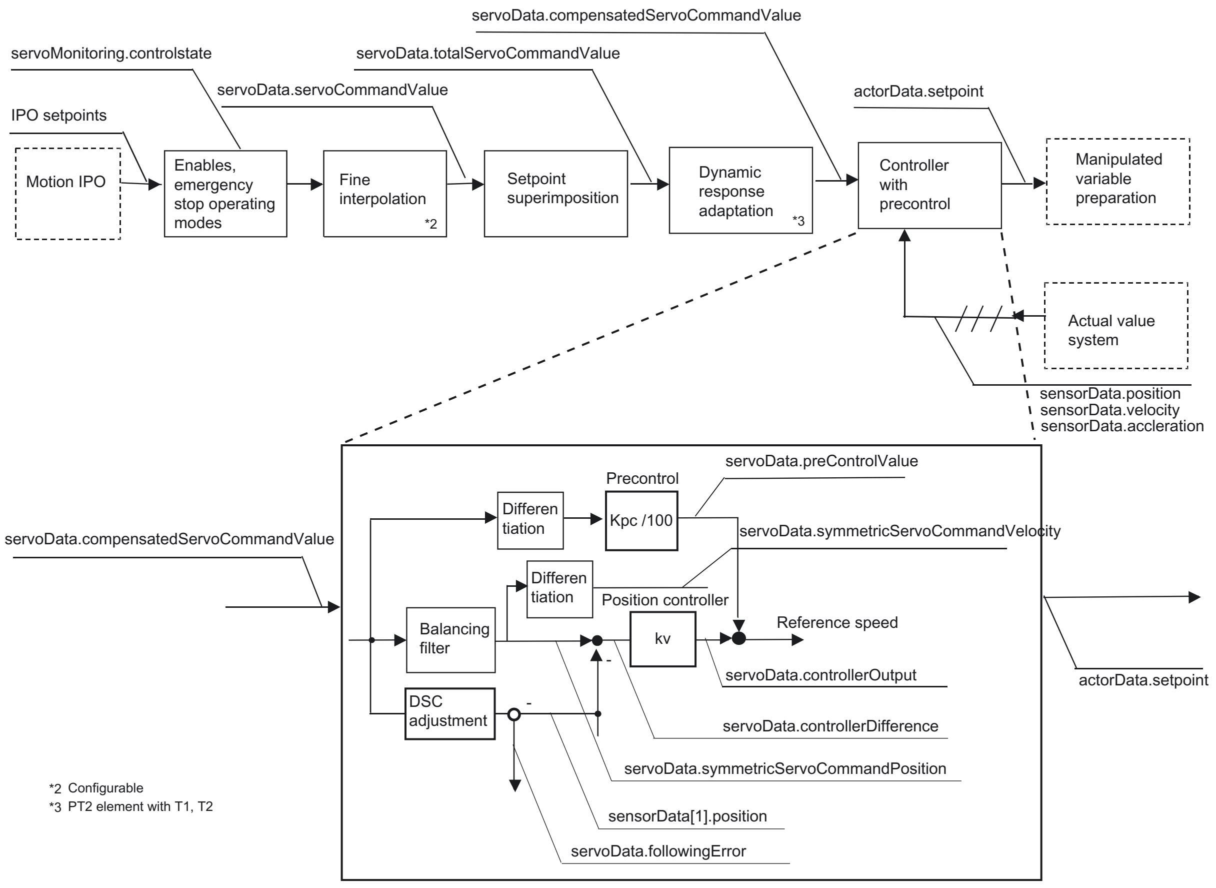
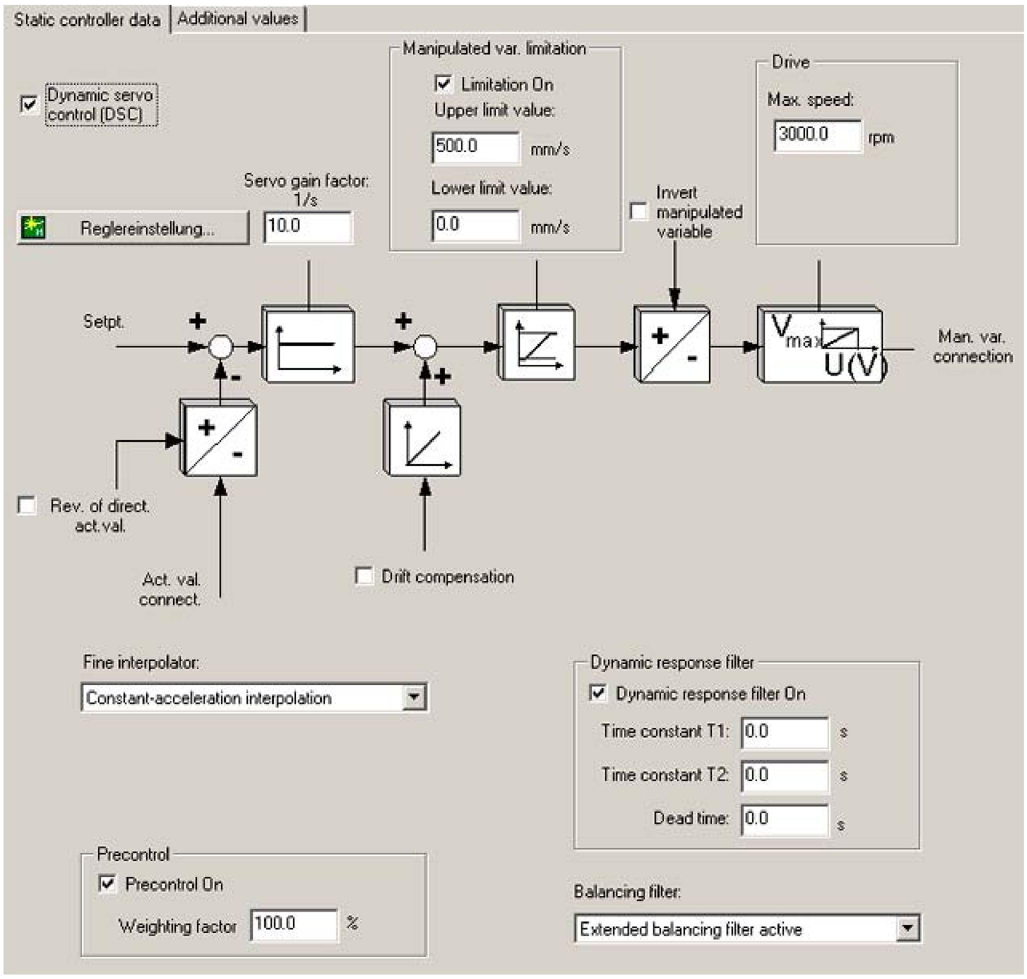

servoData.compensatedServoCommandValue
servoMonitoring.controlstate
servoData.totalServoCommandValue
servoData.servoCommandValue
actorData.setpoint
IPO setpoints
Motion IPO
Enables, emergency stop operating modes
Fine interpolation
*2
Setpoint superimposition
Dynamic response adaptation
*3
Controller with precontrol
Manipulated variable preparation
Actual value system
sensorData.position
sensorData.velocity
sensorData.accleration
servoData.preControlValue
Precontrol
Differen tiation
Kpc /100
servoData.compensatedServoCommandValue
servoData.symmetricServoCommandVelocity
Differen tiation
Position controller
Balancing filter
kv
Reference speed
servoData.controllerOutput
actorData.setpoint
DSC adjustment
servoData.controllerDifference
servoData.symmetricServoCommandPosition
sensorData[1].position
servoData.followingError
*2 Configurable
*3 PT2 element with T1, T2

Figure 2-35 Proportional-action controller with precontrol



☑ Dynamic servo control (DSC)

Servo gain factor: 1/s: 10.0

Manipulated var. limitation
☑ Limitation On
Upper limit value: 500.0 mm/s
Lower limit value: 0.0 mm/s

☐ Invert manipulated variable

Drive
Max. speed: 3000.0 rpm

☐ Rev. of direct. act.val.

☐ Drift compensation

Fine interpolator: Constant-acceleration interpolation

Dynamic response filter
☑ Dynamic response filter On
Time constant T1: 0.0 s
Time constant T2: 0.0 s
Dead time: 0.0 s

Precontrol
☑ Precontrol On
Weighting factor 100.0 %

Balancing filter: Extended balancing filter active

Figure 2-36 Static controller data

Note

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

Recommendation

Controllers featuring a precontrol function should be used as shown below:

- 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 246).

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

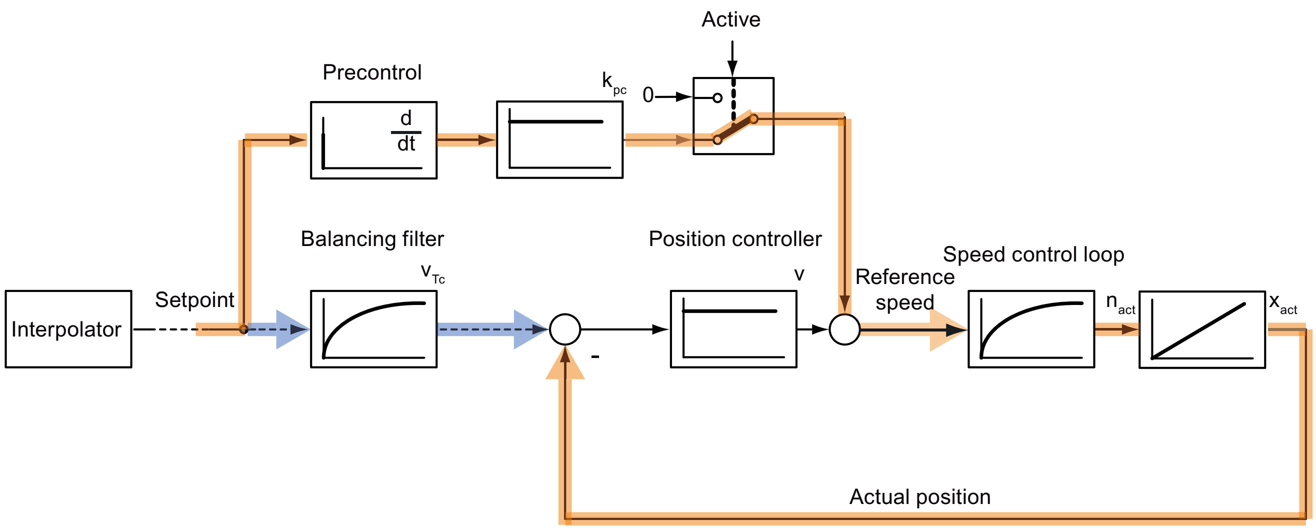


Figure 2-37 Balancing filter - Example of an electrical axis without DSC

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**: $T_{\text{Balancing filter}}:=T_{\text{Speed control loop equivalent time}}+T_i+T_o+T_{dp}+T_{servo}$

    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**: $T_{\text{Balancing filter}}:=T_{\text{Speed control loop equivalent time}}$, 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**: $T_{\text{Balancing filter}}:=T_{\text{qOutputEquivalent time}}$ + Communication times according to the setpoint output/actual value interface.

    ($T_{\text{qOutputEquivalent time}}$ 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.

System variable: **servoData.ControllerDifference**

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 controller 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 controller is displayed with DSC.

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 enabled for stepper motors.

See also

Overview of commissioning the position controller of positioning axes (Page 127)

Hydraulic axis with position control/velocity control (Page 246)

2.11.3 Dynamic Servo Control (DSC)

With the Dynamic Servo Control function, the dynamically active component of the position controller in the drive is executed at the frequency of the speed loop.

It is thus possible to set a substantially greater position controller gain factor Kv. This increases the dynamic response for the reference variable sequence and disturbance variable compensation for highly dynamic drives.

The position differential (XERR) and the gain factor for the position controller in the drive are transferred in the PROFIdrive message frame, 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 message frames 5 and 6 according to PROFIdrive), SIMODRIVE 611U, and SINAMICS S120 (additional SIEMENS message frames 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.

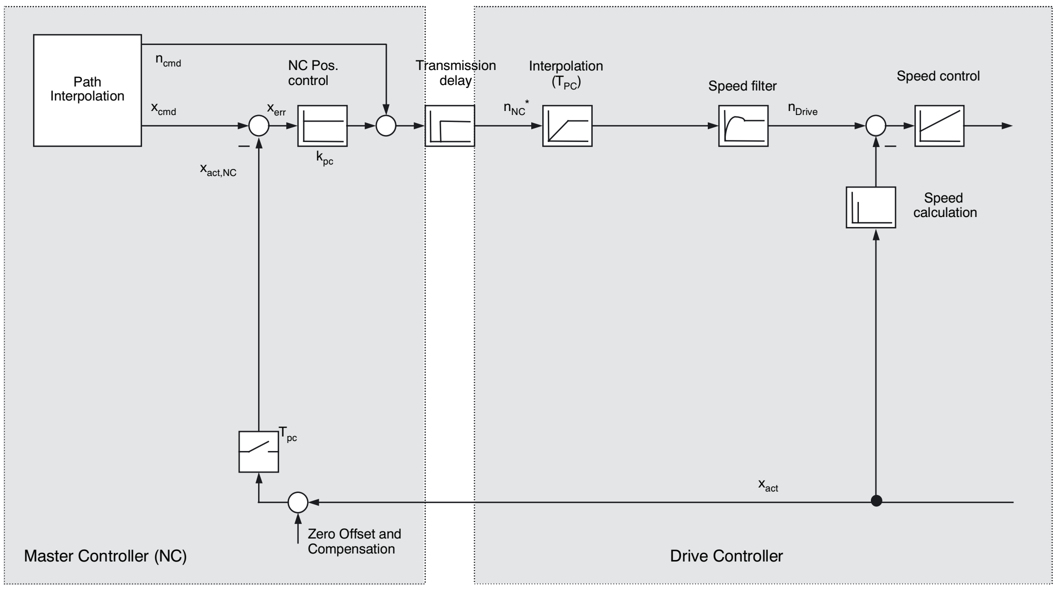
Advantages of DSC (compared to a position controller in the control unit):

- Higher servo gain factor Kv (position controller gain) possible
- Larger bandwidth -> higher dynamic response
- Shorter response times for disturbance characteristic

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 message frame 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 1 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.

Structure



Symbols: $n_{cmd:}$: speed command
x_{cmd} : position command
x_{err} : position error command
x_{act} : actual position
T_{pc} : position controller sampling time (= T_{MAPC})
k_{pc} : position controller gain

Figure 2-38 Structure of the position-control loop with the velocity setpoint interface to the drive without DSC

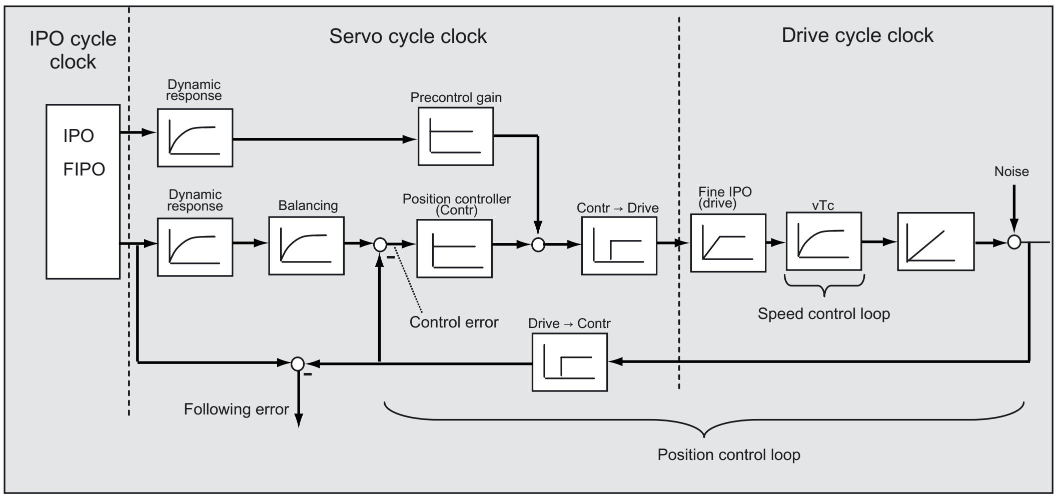


Figure 2-39 Control structure without DSC (simplified)

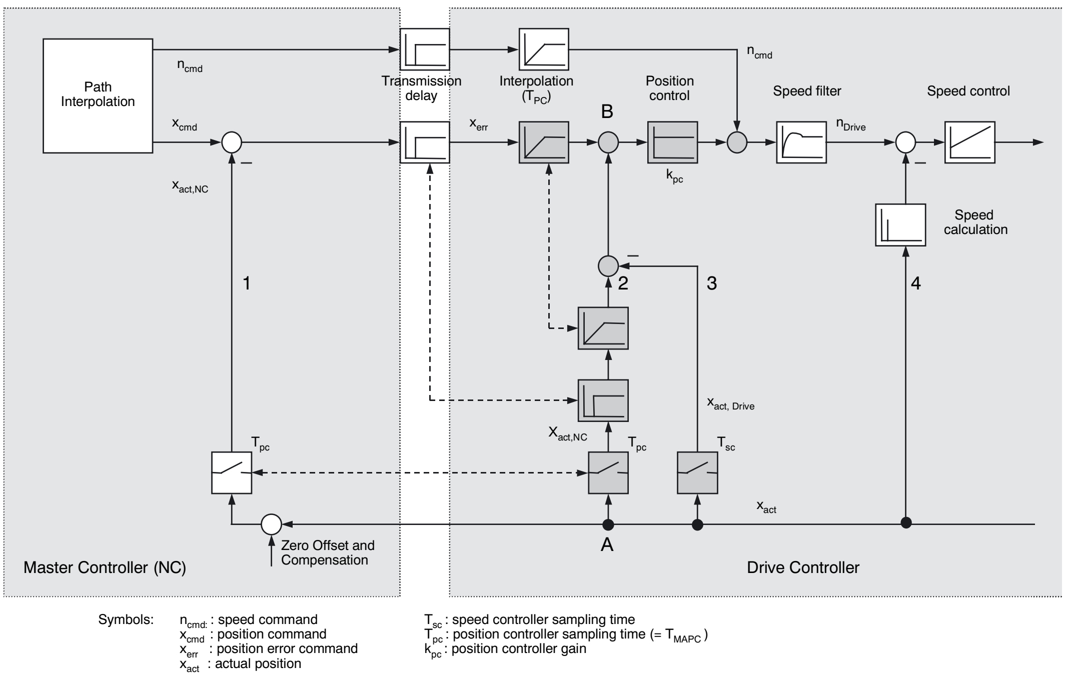


Figure 2-40 Structure of the position-control loop with DSC functionality in the drive

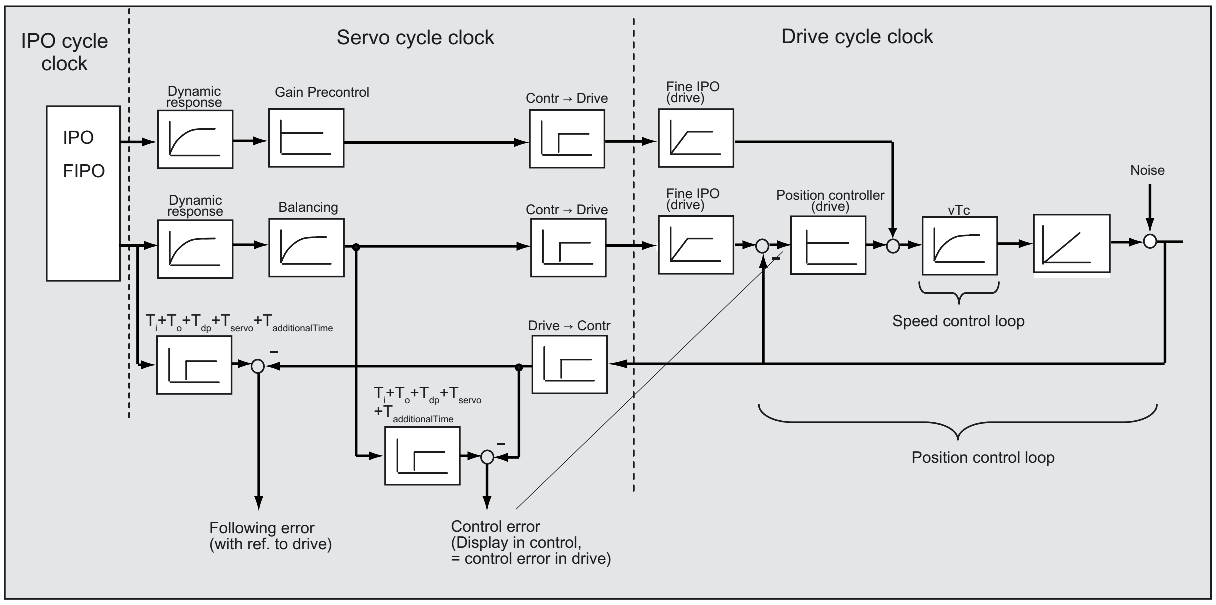


Figure 2-41 Control structure with DSC (simplified)

Further information on DSC can be found in the relevant drive documentation, e.g., in the SINAMICS S120 Commissioning Manual for SINAMICS.

See also

Overview of commissioning the position controller of positioning axes (Page 127)

2.11.4 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.

2.11.5 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 97).

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.

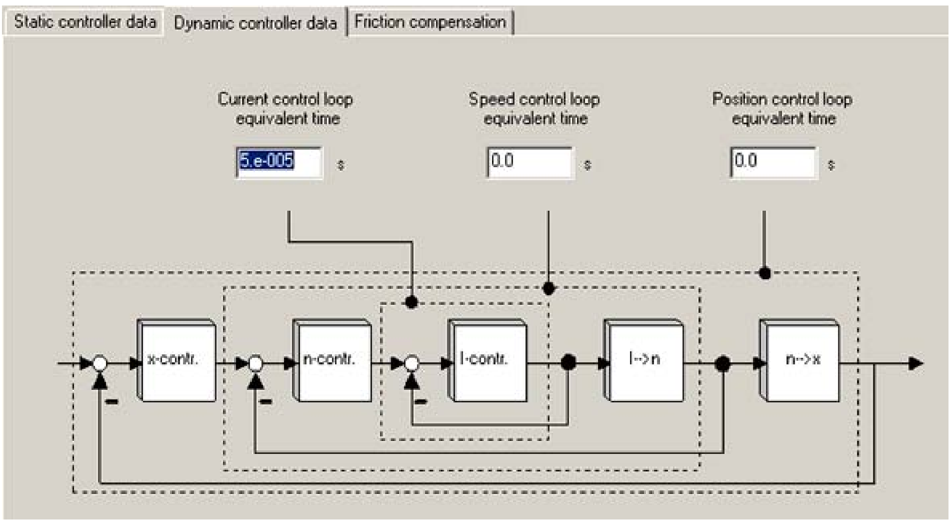
With DSC, the equivalent time of the position control loop can be set as follows:

- 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)



Current control loop equivalent time: 5.e-005 s
Speed control loop equivalent time: 0.0 s
Position control loop equivalent time: 0.0 s

Figure 2-42 Dynamic controller data

2.11.6 Setpoint superimposition

The setpoint specified by the interpolator can be superimposed in the servo via cyclically active system variables.

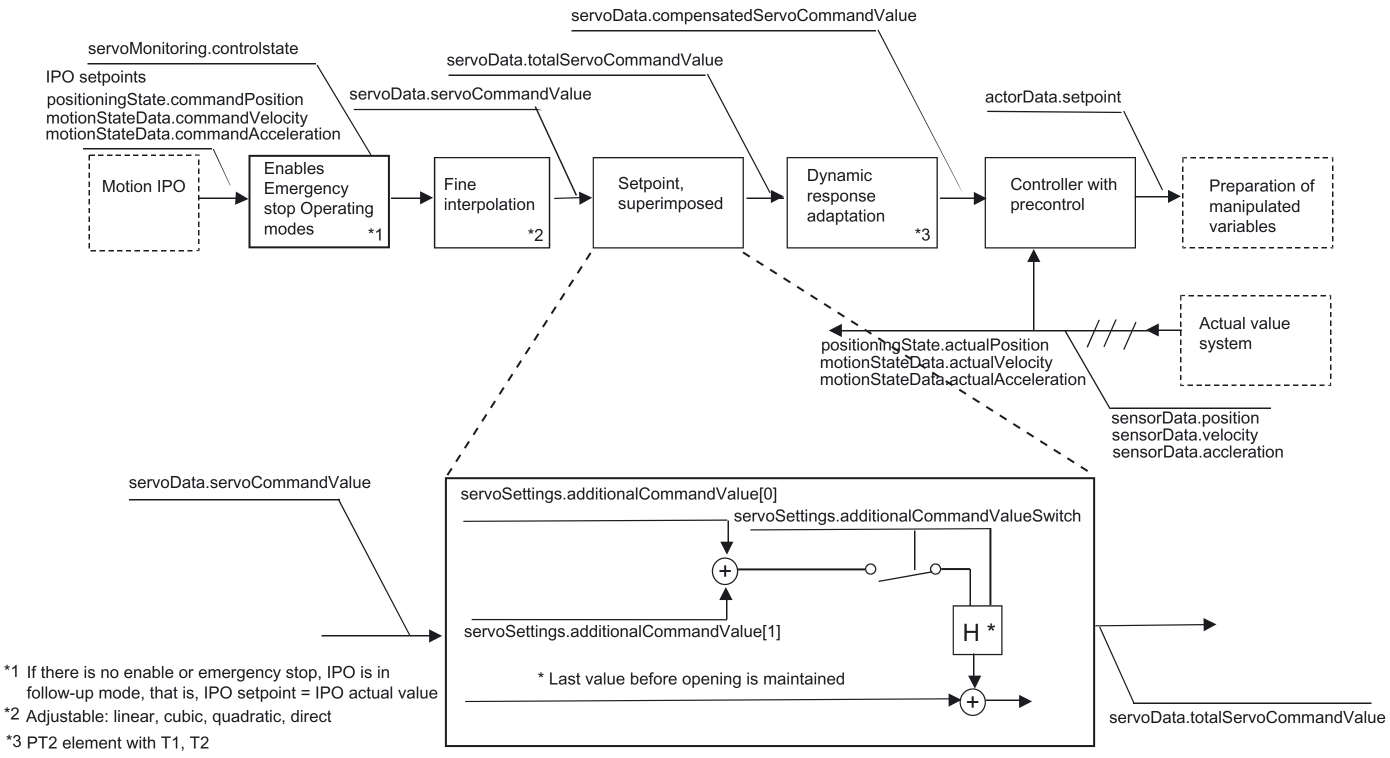

servoData.compensatedServoCommandValue
servoMonitoring.controlstate
servoData.totalServoCommandValue
IPO setpoints
positioningState.commandPosition
motionStateData.commandVelocity
motionStateData.commandAcceleration
servoData.servoCommandValue
actorData.setpoint
Motion IPO
Enables Emergency stop Operating modes *1
Fine interpolation *2
Setpoint, superimposed
Dynamic response adaptation *3
Controller with precontrol
Preparation of manipulated variables
Actual value system
positioningState.actualPosition
motionStateData.actualVelocity
motionStateData.actualAcceleration
sensorData.position
sensorData.velocity
sensorData.accleration
servoData.servoCommandValue
servoSettings.additionalCommandValue[0]
servoSettings.additionalCommandValueSwitch
servoSettings.additionalCommandValue[1]
H *
* Last value before opening is maintained
servoData.totalServoCommandValue
*1 If there is no enable or emergency stop, IPO is in follow-up mode, that is, IPO setpoint = IPO actual value
*2 Adjustable: linear, cubic, quadratic, direct
*3 PT2 element with T1, T2

Figure 2-43 Setpoint superimposition

Note

An axis operated with setpoint superimposition cannot be switched back directly to normal position-controlled mode. You must first reset the setpoint superimpositions to zero.

Superimposition has an effect on the position of the positioning 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.

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.

2.11.7 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.

2.11.8 Actual value measurement / actual value system

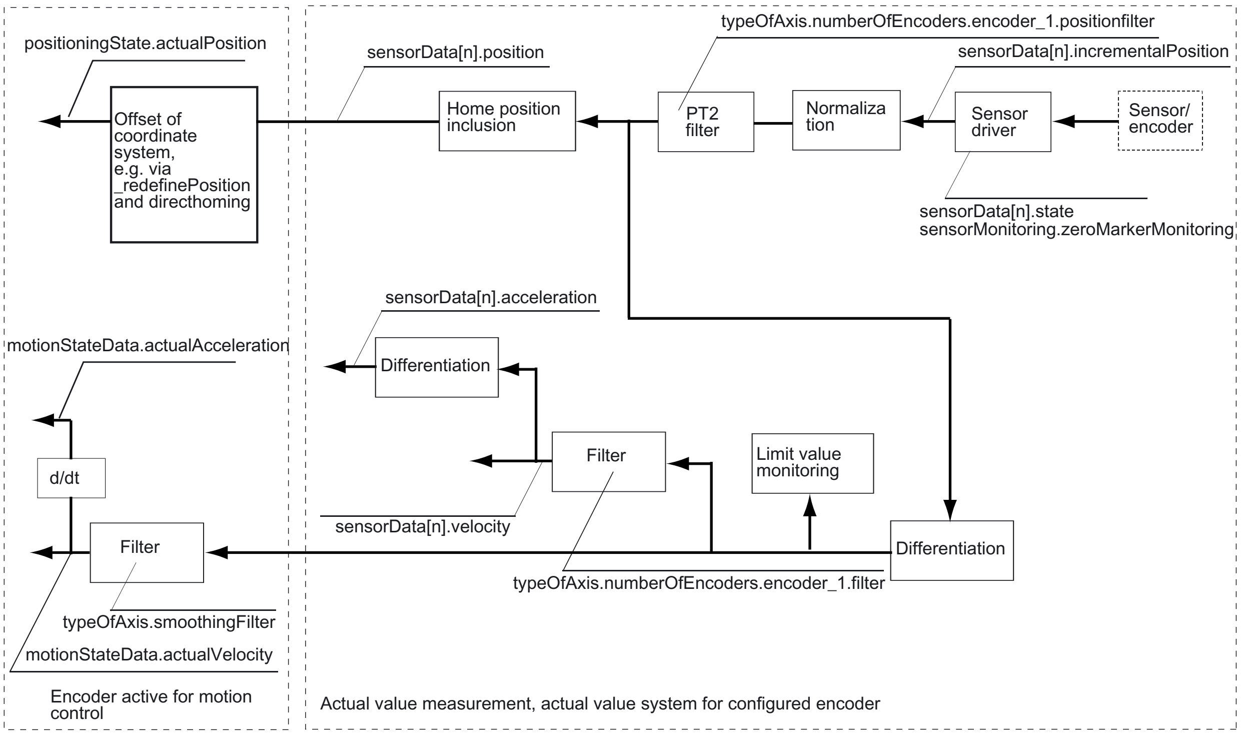

positioningState.actualPosition
Offset of coordinate system, e.g. via _redefinePosition and directhoming
sensorData[n].position
Home position inclusion
typeOfAxis.numberOfEncoders.encoder_1.positionfilter
PT2 filter
Normaliza tion
sensorData[n].incrementalPosition
Sensor driver
Sensor/ encoder
sensorData[n].state
sensorMonitoring.zeroMarkerMonitoring
sensorData[n].acceleration
Differentiation
motionStateData.actualAcceleration
d/dt
Filter
Limit value monitoring
sensorData[n].velocity
Filter
Differentiation
typeOfAxis.numberOfEncoders.encoder_1.filter
typeOfAxis.smoothingFilter
motionStateData.actualVelocity
Encoder active for motion control
Actual value measurement, actual value system for configured encoder

Figure 2-44 Actual value system

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.

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 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**).

Note

Extreme care must be taken when changing the extrapolation time to the runtime; otherwise knocking could result in the machine.

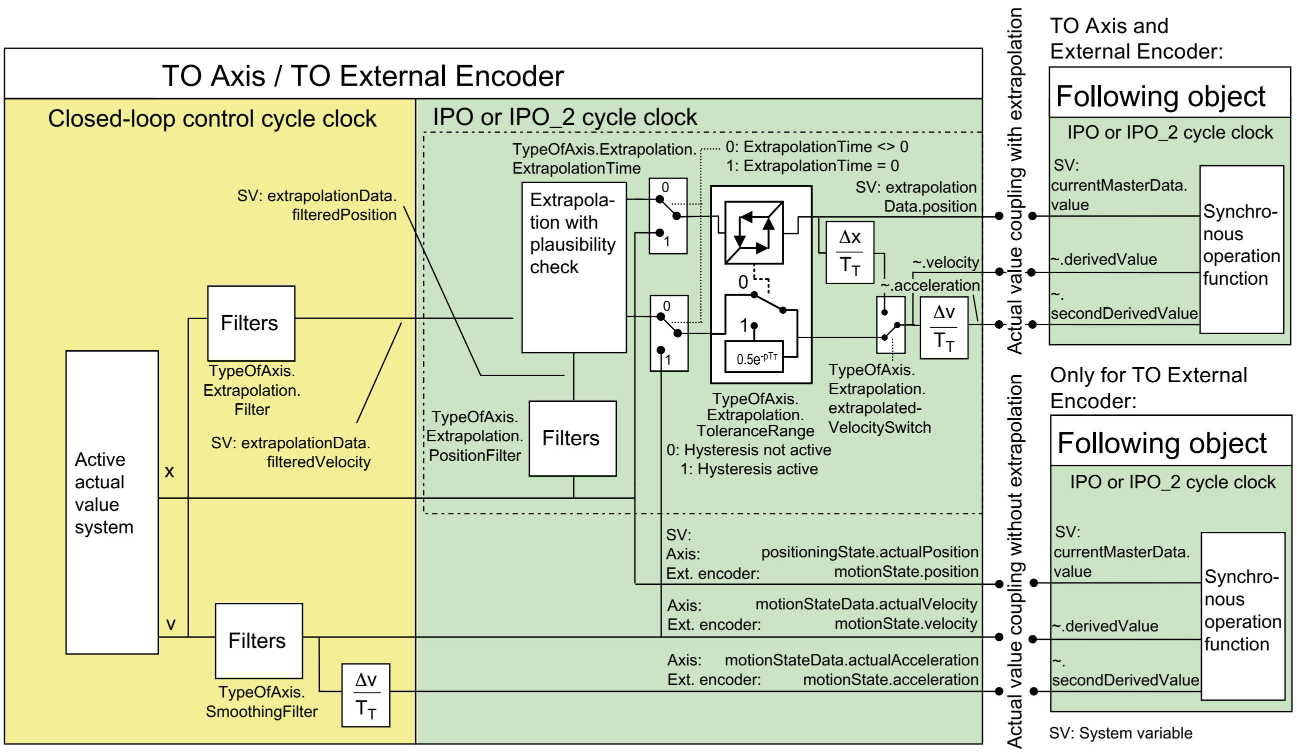


Figure 2-45 Actual value coupling with extrapolation for the Axis technology object or External Encoder technology object

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**).

The filter for the velocity during extrapolation is independent of this filter.

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.)

Transferring the actual velocity from the drive (as of V4.1 SP1)

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 message frame
- 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 counted exactly during operation, but it is reset when the system is switched on and in response to initialization tasks such as set actual value system (**_redefinePosition()**) or homing (**_homing()**).
- There is no special overflow handling for the count value.

  A counter overflow must be taken into account in the user program.

Table 2- 19 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

See also

Actual value coupling with extrapolation

2.11.9 Preparation of manipulated variables for electric axis

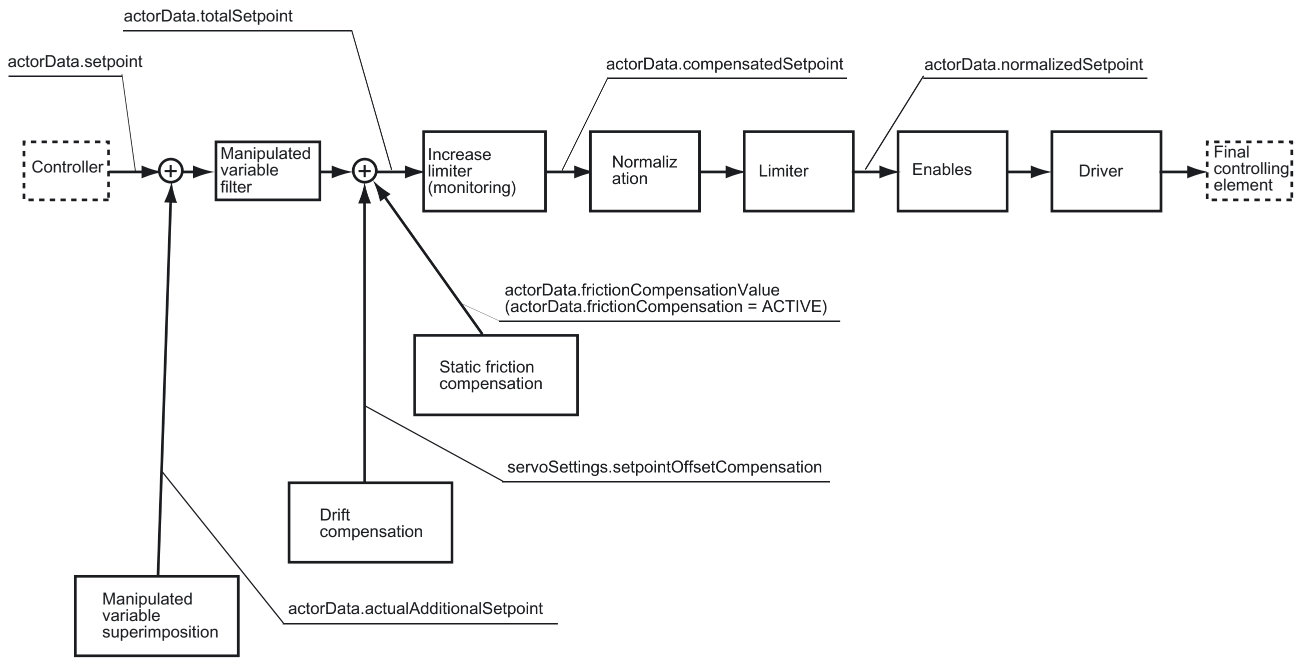

actorData.totalSetpoint
actorData.setpoint
actorData.compensatedSetpoint
actorData.normalizedSetpoint
Controller
Manipulated variable filter
Increase limiter (monitoring)
Normaliz ation
Limiter
Enables
Driver
Final controlling element
actorData.frictionCompensationValue (actorData.frictionCompensation = ACTIVE)
Static friction compensation
servoSettings.setpointOffsetCompensation
Drift compensation
Manipulated variable superimposition
actorData.actualAdditionalSetpoint

Figure 2-46 Manipulated variable preparation

2.11.10 Manipulated variable superimposition

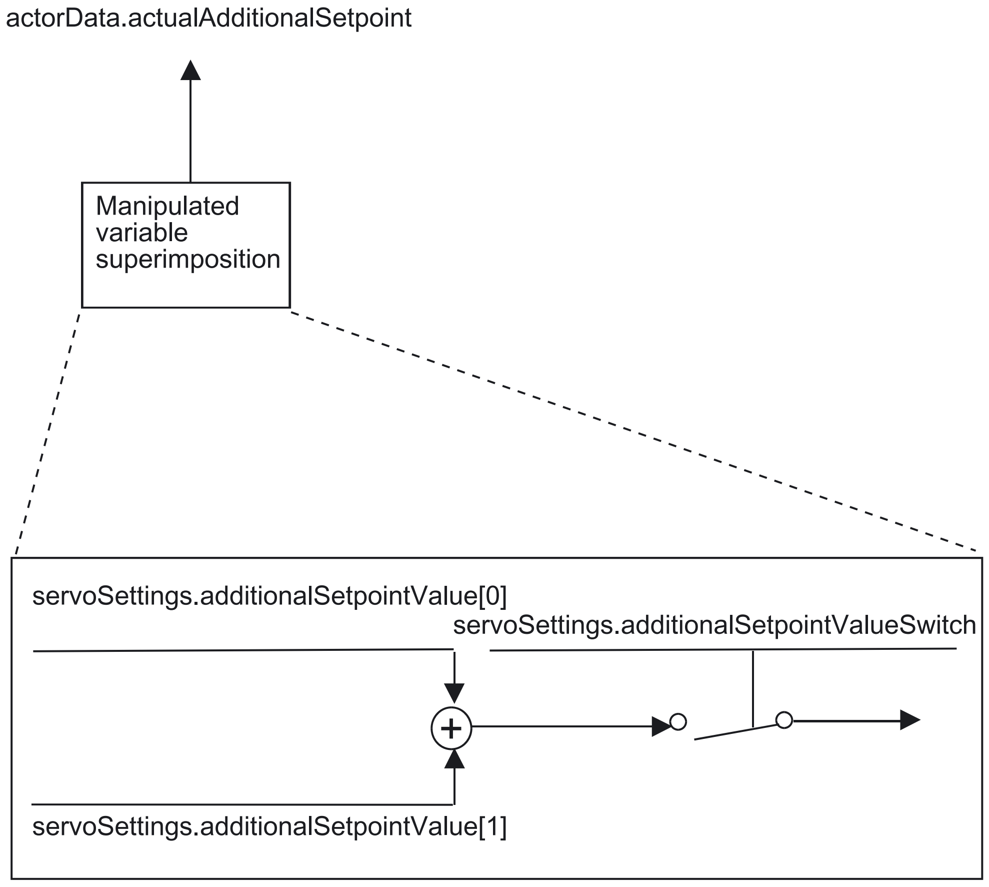

actorData.actualAdditionalSetpoint
Manipulated variable superimposition
servoSettings.additionalSetpointValue[0]
servoSettings.additionalSetpointValueSwitch
+
servoSettings.additionalSetpointValue[1]

Figure 2-47 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 does not act for the FEEDBACK_EMERGENCY_STOP alarm response and the **_stopEmergency()** command with stopMode:=STOP_WITH_COMMAND_VALUE_ZERO.

2.11.11 Manipulated variable filtering (as of V4.1 SP1)

A manipulated variable 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 are effective immediately.

With the DSC setting, the manipulated variable filter is only effective for the precontrol.

2.11.12 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 servoSettings.setpointOffsetCompensation system variable.

2.11.13 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.

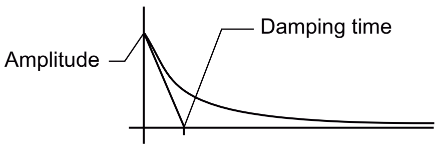
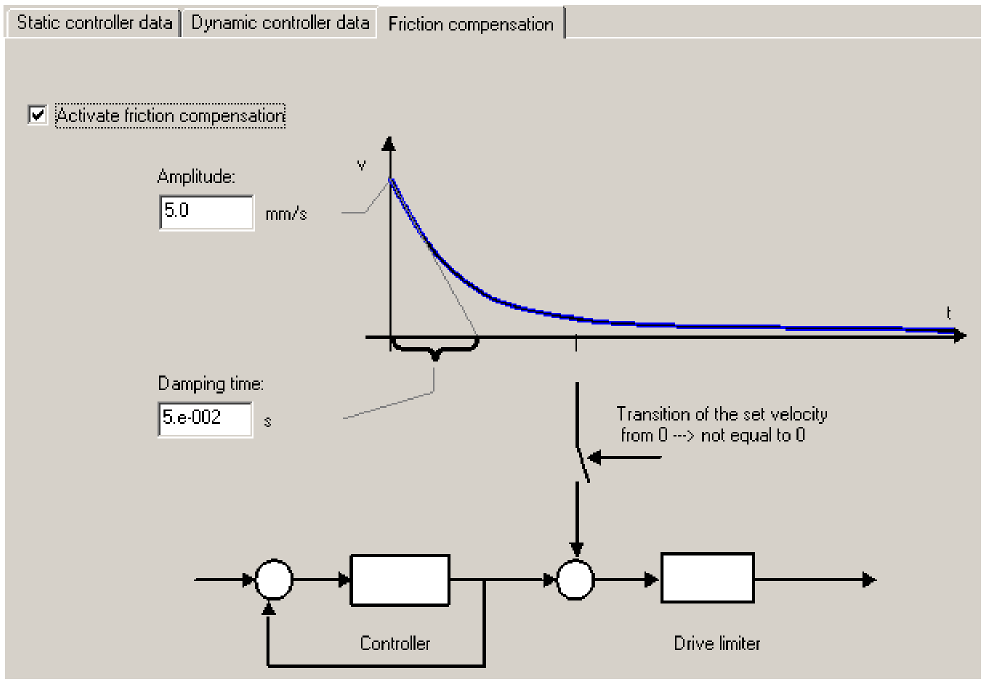


Figure 2-48 Static friction compensation

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.



☑ Activate friction compensation

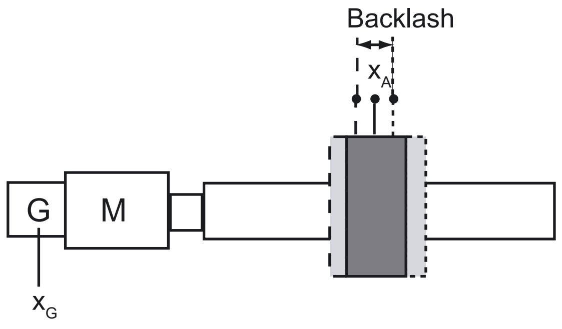
Amplitude: 5.0 mm/s

Damping time: 5.e-002 s

Figure 2-49 Static friction compensation

2.11.14 Backlash on reversal compensation

During the power transmission between a moving machine part and the corresponding drive, a backlash on reversal (play) usually occurs.



G: Encoder
M: Motor
X_G: Actual encoder value
X_A: Axis position

Figure 2-50 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 for 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

In a direction reversal and with 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.

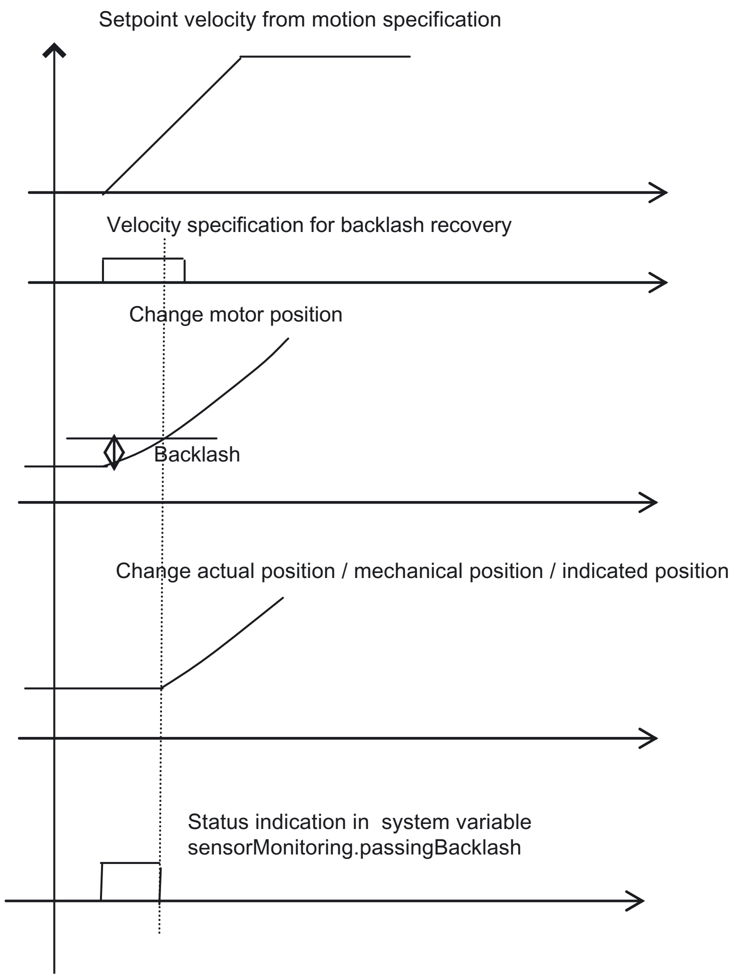

Setpoint velocity from motion specification
Velocity specification for backlash recovery
Change motor position
Backlash
Change actual position / mechanical position / indicated position
Status indication in system variable sensorMonitoring.passingBacklash

Figure 2-51 Backlash on reversal compensation sequence

2.11.15 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 2- 20 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 269)

Moving (Page 283)

Traversing the axis via velocity specifications (Page 142)

2.11.16 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.

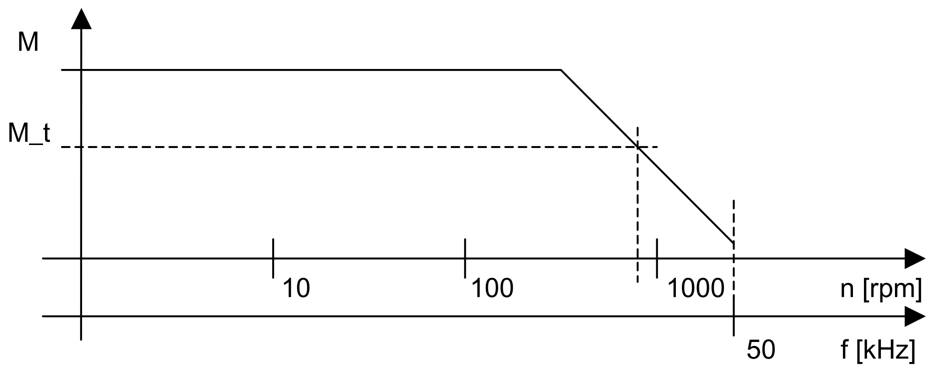


Figure 2-52 Example of a stepper motor torque characteristic

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.

2.11.17 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.

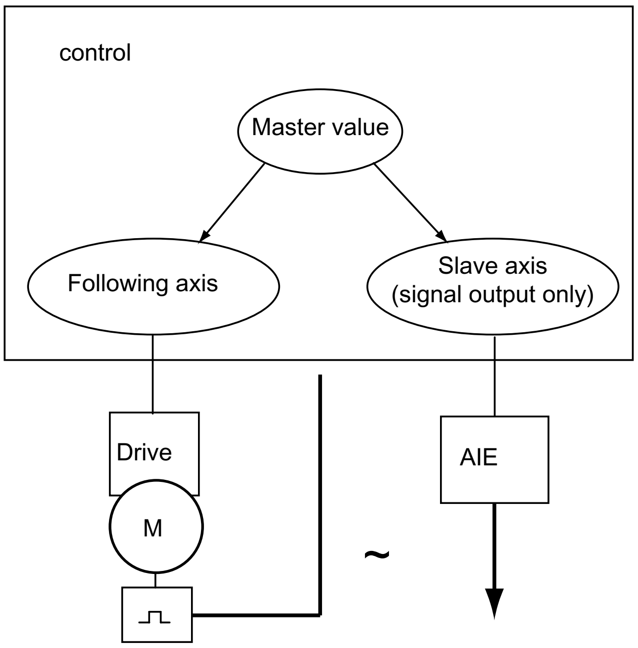


Figure 2-53 Using the axis for encoder signal output

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.

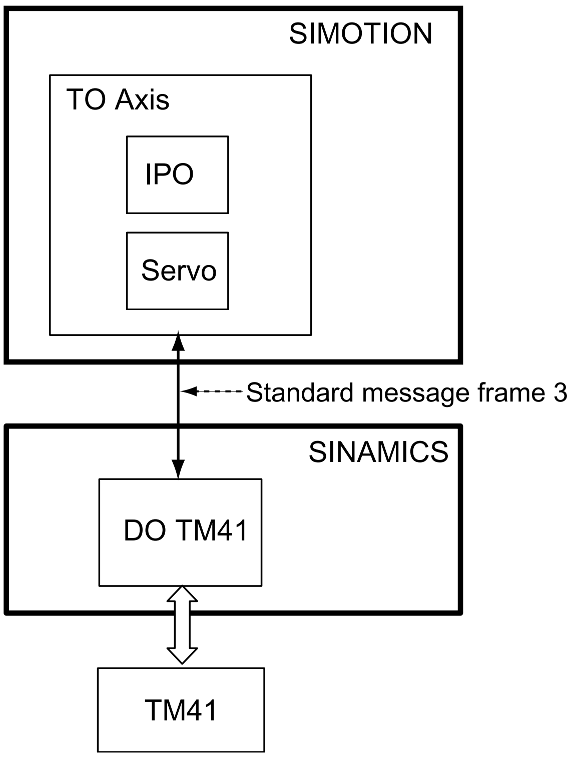


Figure 2-54 Interface of TM41 / DO41 to Axis technology object

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 enabled 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

For more information, refer to the Utilities & Applications CD under FAQs.

See also

Setting as a real axis with encoder signal simulation (V4.0 and higher) (Page 44)

2.12 Commissioning the position controller of positioning axes

2.12.1 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 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 226) for manual optimization and the automatic controller setting (Page 218) 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 will find a commissioning guide on the Utilities & Applications CD under FAQs.

See also

Overview of automatic controller setting (as of V4.1 SP1) (Page 218)

SIMOTION measuring functions (Page 226)

Axis control panel (Page 232)

2.12.2 Configuration data

Configuration data in the axis data set

The configuration data of a data set are contained in the structure **TypeOfAxis.NumberOfDataSets.DataSet_n** (n: data set number).

Table 2- 21 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
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
Mechanics	
Load gear	• Load revolutions Gear.denFactor • Motor revolutions: Gear.numFactor

Table 2- 22 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)

2.12.3 Example of commissioning a proportional-action controller with precontrol

The procedure for determining the control parameters of an electric axis is presented below using a proportional-action 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 controller 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 controller and the drive is faulty.

A common cause for the faulty behavior is a discrepancy in the normalization speed configuration between the SIMOTION controller 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 controller 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 transferred from the drive using the axis wizard, see Setting as a real axis with digital drive coupling (Page 36).

Table 2- 23 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 enabled 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 109).

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 enabled, 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 controller is powered up. The following online functions are available for this purpose under the target device in SCOUT.

Table 2- 24 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.

2.13 Command variable calculation

2.13.1 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.

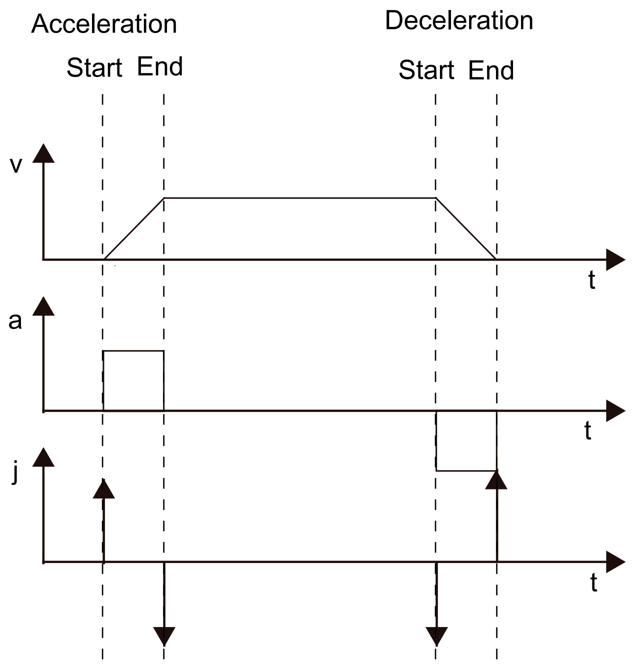
The parameters for profile, velocity, acceleration, and jerk can be assigned in the command, or the **userDefault** value settings are used.

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.

For single axis movements and for path movements, travel to zero acceleration is made for specification of a continuous acceleration curve for smoothing in the transition to the movement of the new command.

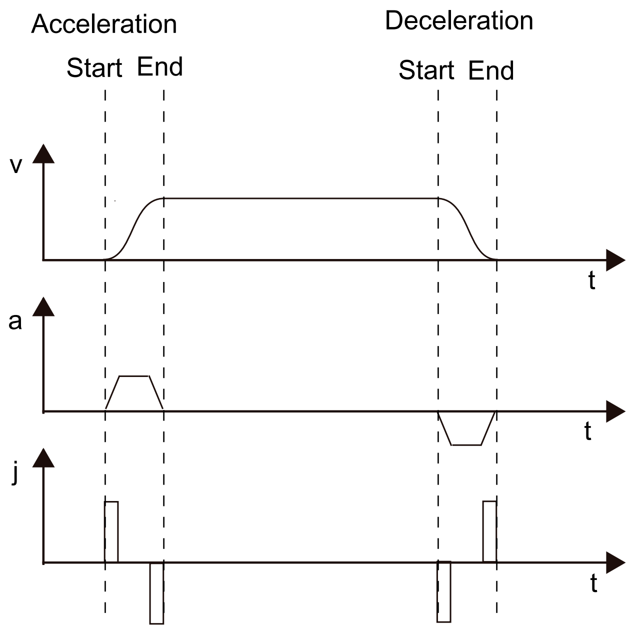
For single axis movements, for path movements, for the synchronization of synchronization movements and for the specification of a continuous acceleration curve, for any required reversing in the reversal point of the movement direction, the acceleration is made to zero. This means the acceleration/deceleration will be removed using the jerk and re-established after the reversal using the jerk.

This does not apply for motion specifications via user-defined profiles or via the MotionIn interface.



v - velocity
a - acceleration
j - jerk
t - time

Figure 2-55 Trapezoidal velocity profile



v - velocity
a - acceleration
j - jerk
t - time

Figure 2-56 Smooth velocity profile

2.13.2 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.

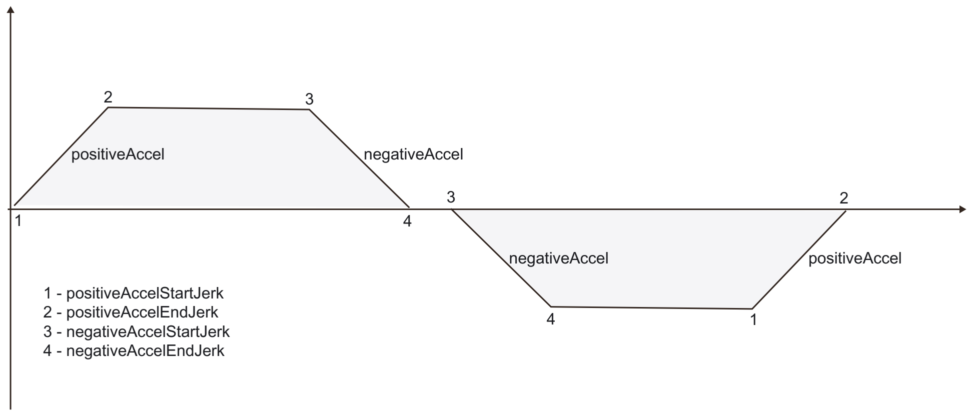


Figure 2-57 Operative points for acceleration and jerk parameters

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

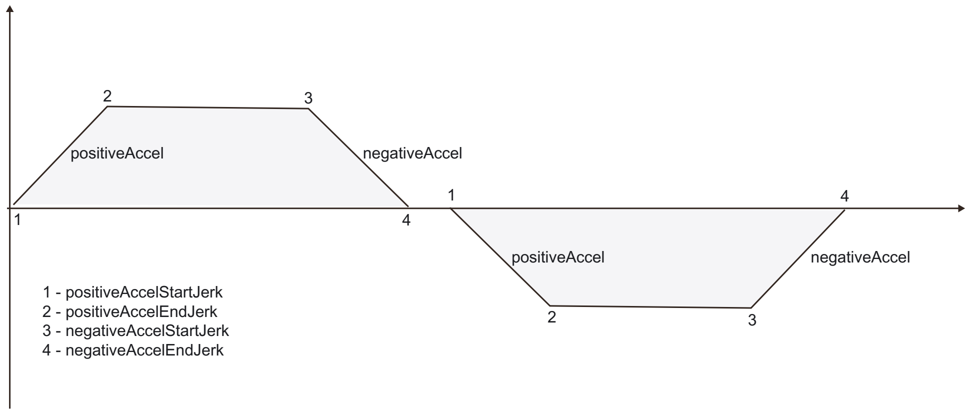


Figure 2-58 Operative points for acceleration and jerk parameters

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.

2.13.3 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**.

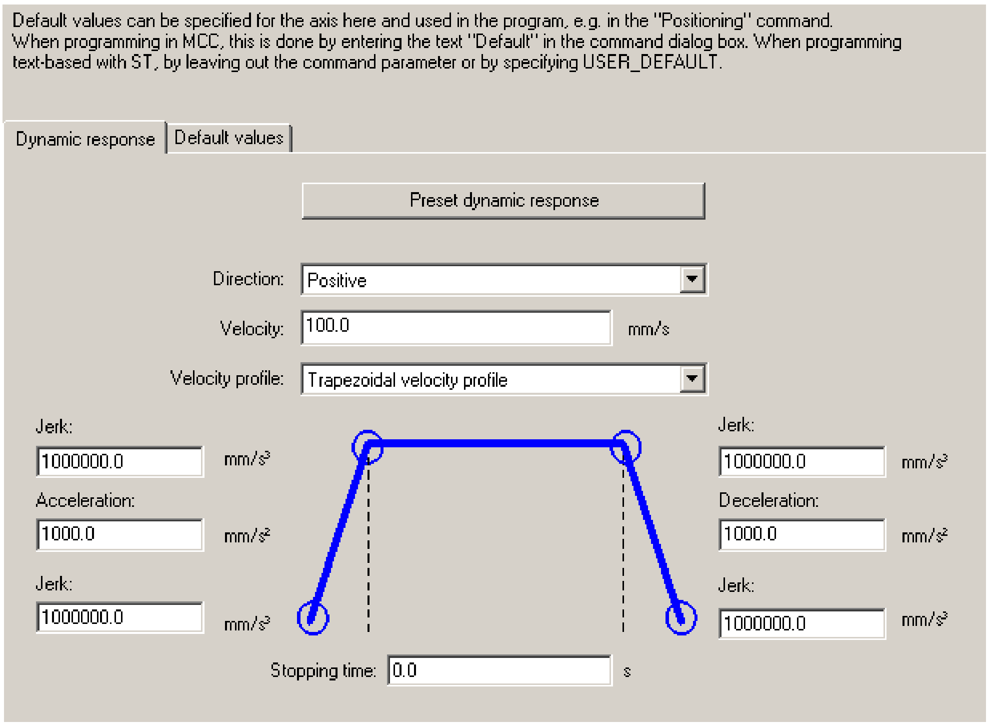
The override for the **velocity** can be set from **0** to **200%**.

The override for the **acceleration/deceleration** can be set from **1%** to **1000%**.

System variables

- override.velocity
- override.acceleration

2.13.4 Default settings for dynamic response parameters



Default values can be specified for the axis here and used in the program, e.g. in the "Positioning" command.
When programming in MCC, this is done by entering the text "Default" in the command dialog box. When programming text-based with ST, by leaving out the command parameter or by specifying USER_DEFAULT.

Direction: Positive

Velocity: 100.0 mm/s

Velocity profile: Trapezoidal velocity profile

Jerk: 1000000.0 mm/s³

Acceleration: 1000.0 mm/s²

Jerk: 1000000.0 mm/s³

Jerk: 1000000.0 mm/s³

Deceleration: 1000.0 mm/s²

Jerk: 1000000.0 mm/s³

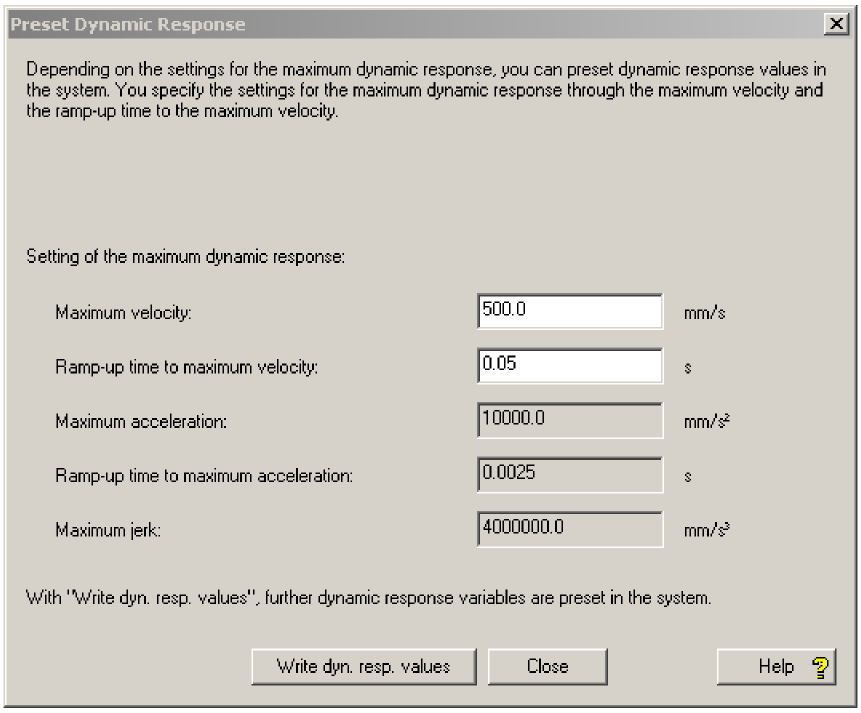
Stopping time: 0.0 s

Figure 2-59 Default settings for dynamic response parameters

Stop time

This time acts when for **_stopEmergency()** and the **stopDriveMode:=**STOP_IN_DEFINED_TIME setting, the default value is used for the time.

Dynamic response default setting



Depending on the settings for the maximum dynamic response, you can preset dynamic response values in the system. You specify the settings for the maximum dynamic response through the maximum velocity and the ramp-up time to the maximum velocity.

Setting of the maximum dynamic response:

Maximum velocity:	500.0	mm/s
Ramp-up time to maximum velocity:	0.05	s
Maximum acceleration:	10000.0	mm/s²
Ramp-up time to maximum acceleration:	0.0025	s
Maximum jerk:	4000000.0	mm/s³

With "Write dyn. resp. values", further dynamic response variables are preset in the system.

Figure 2-60 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.

See also

Velocity profiles (Page 133)

Defining accelerations and decelerations (Page 135)

Defaults (Page 68)

2.13.5 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.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

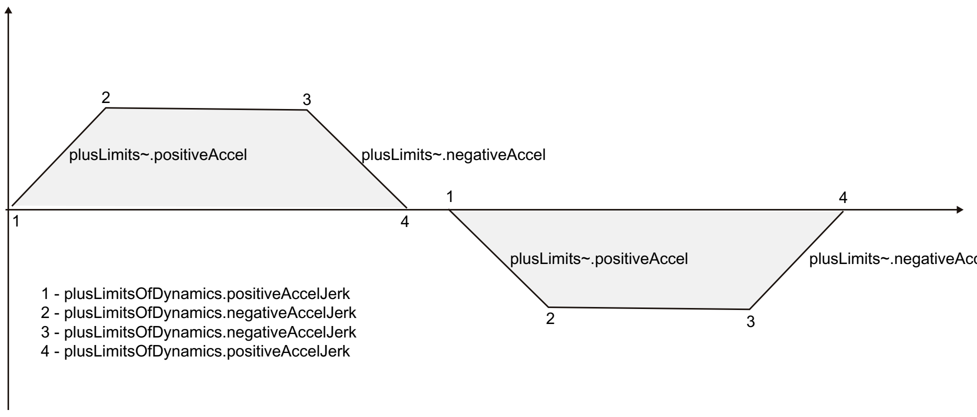

2
3
plusLimits~.positiveAccel
plusLimits~.negativeAccel
1
4
1
4
plusLimits~.positiveAccel
plusLimits~.negativeAccel
2
3
1 - plusLimitsOfDynamics.positiveAccelJerk
2 - plusLimitsOfDynamics.negativeAccelJerk
3 - plusLimitsOfDynamics.negativeAccelJerk
4 - plusLimitsOfDynamics.positiveAccelJerk

Figure 2-61 Technological maximum values for non-direction-dependent setting

Technological maximum values for direction-dependent setting:

- **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

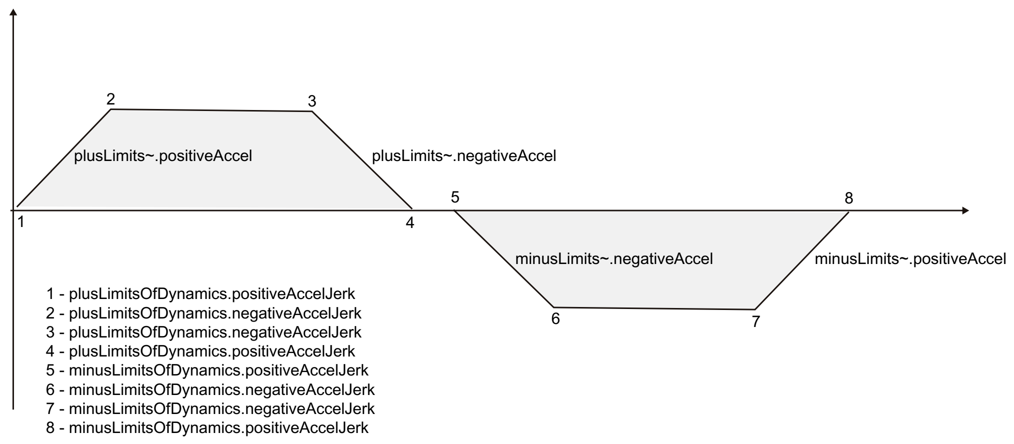

2
3
plusLimits~.positiveAccel
plusLimits~.negativeAccel
5
8
1
4
minusLimits~.negativeAccel
minusLimits~.positiveAccel
6
7
1 - plusLimitsOfDynamics.positiveAccelJerk
2 - plusLimitsOfDynamics.negativeAccelJerk
3 - plusLimitsOfDynamics.negativeAccelJerk
4 - plusLimitsOfDynamics.positiveAccelJerk
5 - minusLimitsOfDynamics.positiveAccelJerk
6 - minusLimitsOfDynamics.negativeAccelJerk
7 - minusLimitsOfDynamics.negativeAccelJerk
8 - minusLimitsOfDynamics.positiveAccelJerk

Figure 2-62 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.

Note

- 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.
- 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.

Note

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).

For the setpoint calculation of the axis, the minimum of the maximum velocity, the maximum technological velocity and the maximum master value velocity of the axis are used.

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.

Note

Please note that when linking with digital drives and with a ramp calculation for the speed active in the drive, additional dynamic limiting 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.

2.13.6 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

2.13.7 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

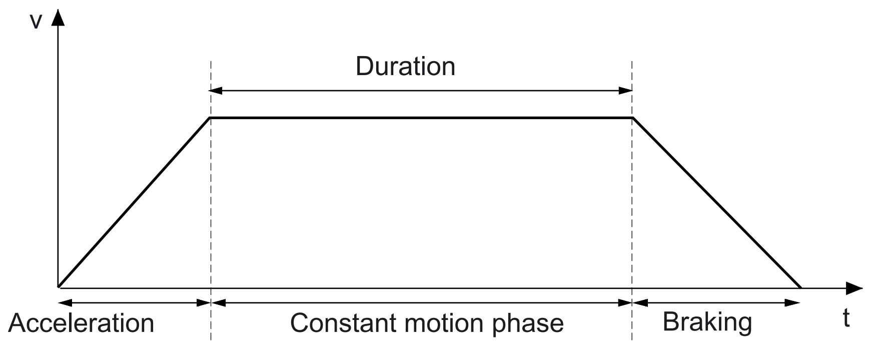


Figure 2-63 Phases for moving with _move()

An acceleration or deceleration phase can be ended with jerk control with the **velocityType:=RESULTING** parameter. The resulting velocity is then retained.

See also

Velocity profiles (Page 133)

Defining accelerations and decelerations (Page 135)

2.13.8 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 88)

2.13.9 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.

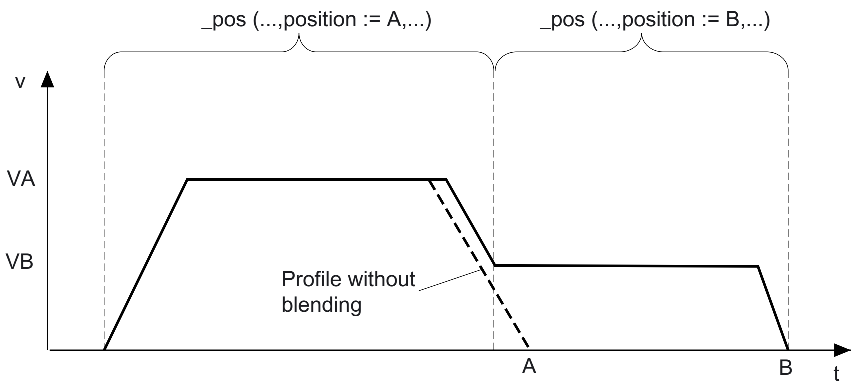


Figure 2-64 Blending when velocity of the subsequent motion is lower

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.

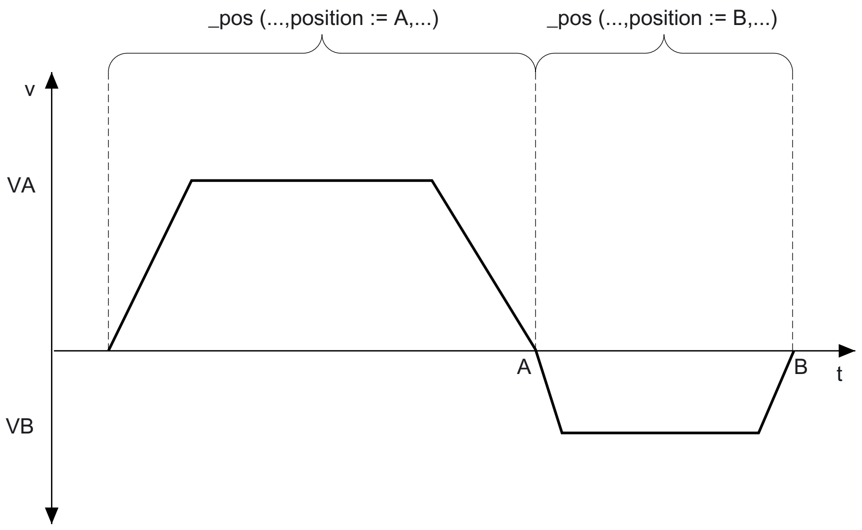


Figure 2-65 Blending when the subsequent motion reverses direction

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.

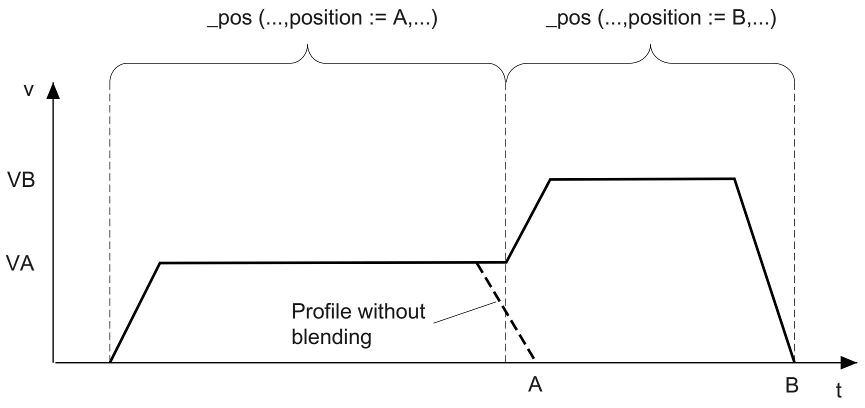


Figure 2-66 Blending when velocity of the subsequent motion is higher

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.

For blending with a programmed smooth acceleration characteristic (SMOOTH), the axis travels at acceleration = 0 during the transition to the new command.

The braking distance is always adhered to, even if the path length of the subsequent motion command is shorter than the required braking distance.

2.13.10 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.

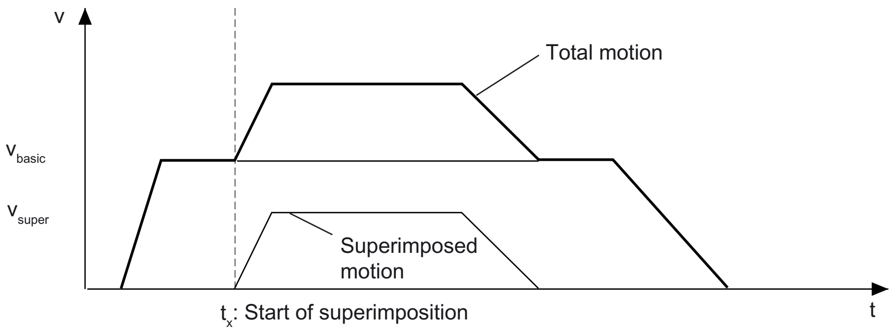


Figure 2-67 Superimposed positioning

See also

Superimposed motion (Page 148)

2.13.11 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 181)

2.13.12 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.

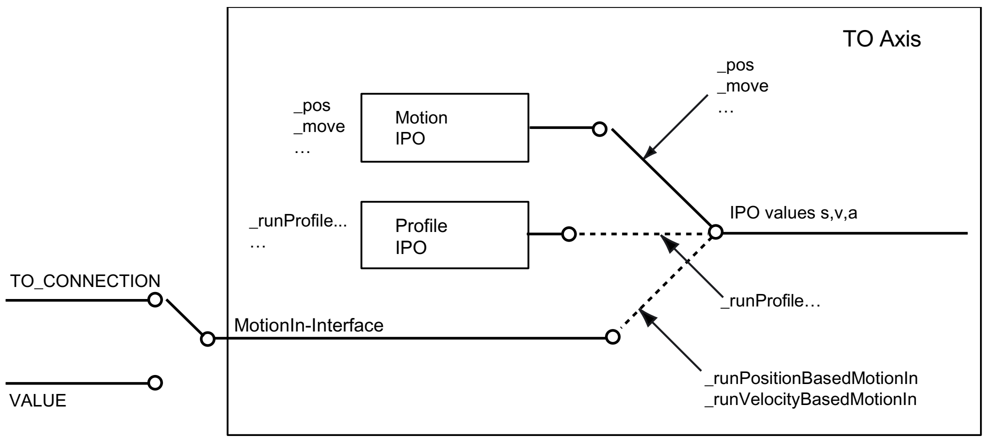
Options for linking speed-controlled and position-controlled axes (positioning axes, synchronous axes) to the motion vector:

- 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 (positioning axis) is linked to a speed-controlled axis.
- Linking speed-controlled axis to motion vector (based on: velocity)

  This links a speed-controlled axis to a speed-controlled 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 2-68 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.

2.13.13 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.

2.14 Superimposed motion

A 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 positioning axis are executed in the superimposed axis coordinate.

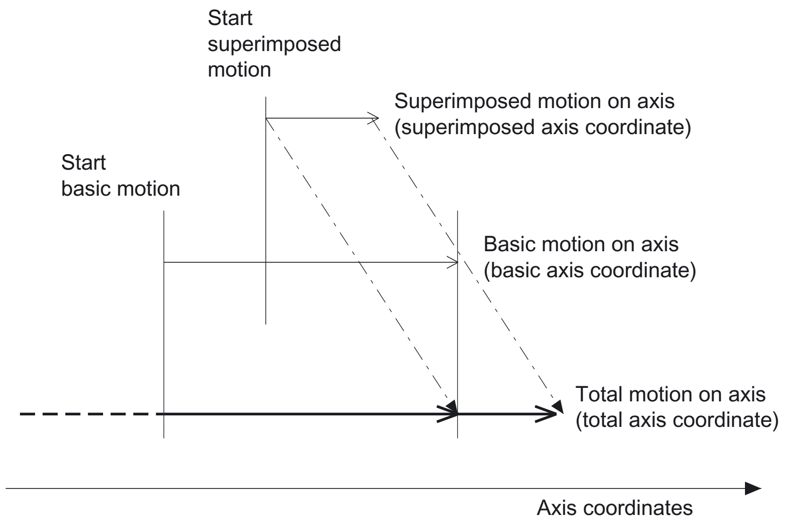

Start superimposed motion
Superimposed motion on axis (superimposed axis coordinate)
Start basic motion
Basic motion on axis (basic axis coordinate)
Total motion on axis (total axis coordinate)
Axis coordinates

Figure 2-69 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.

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

2.15 Torque limiting via torque reduction

2.15.1 Overview of torque limiting via torque reduction

On the electric axis, torque limiting is available via a torque reduction. The limiting value is specified in the command.

This function can be used with a speed-controlled axis, a positioning axis and a synchronous axis. The accuracy depends on the drive used.

To be able to use the function, the PROFIdrive message frame type 10x (apart from message frame 101) must be set. It also requires the use of a drive that supports the function.

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 message frame. 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 enabled before a motion or simultaneously with a motion and can be switched by reissuing the command.

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.

If, for example, torque limiting is not applied during the acceleration stage, the function cannot be activated until after the acceleration stage, or else the acceleration must be reduced.

Note

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.

Otherwise, incorrect limitations will be applied.

For SINAMICS, the parameters are present in P1520, P1521 and P2003:
~.DriveData.maxTorque = P1520 = |P1521|
If the technology data block (Page 159) is interconnected, then: **~.DriveData.maxTorque** = P2003

Enabling / disabling

The limiting function is disabled by default and must be specifically enabled.

The torque limiting is enabled 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 155).

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 enabled.

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 (in order to reduce the difference) even if the velocity calculated by the interpolator is now lower.

  If, for example, torque limiting is not wanted during the acceleration stage, the function cannot be activated until after the acceleration stage, or else the acceleration must be reduced.
- The torque limits B+ / B- and the torque reduction must not be active simultaneously.

Status displays

Table 2- 25 Status display variables

Variable	Meaning
TorqueLimitingCommand.State	
	Indicates the torque limiting status. 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 2- 26 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 assignment: 10.0

Setting the resolution in SIMOTION

The resolution of the torque reduction to the drive can be set in **driveData.torqueReductionGranularity** or **linearMotorDriveData.forceReductionGranularity** in V4.0 and higher.

Possible settings:

- BASIC (default, 1% resolution)

  64H in PZD (PROFIdrive message frame) 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 message frame) 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 288)

2.15.2 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**):

$F = M_{Motor} \times 2 \times \pi \times (\eta_{Leadscrew} / S)$ x (motor revolution/load revolution)

Linear axes with linear motor

There is always a force in N, kN or MN specified when programming.

2.16 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**.

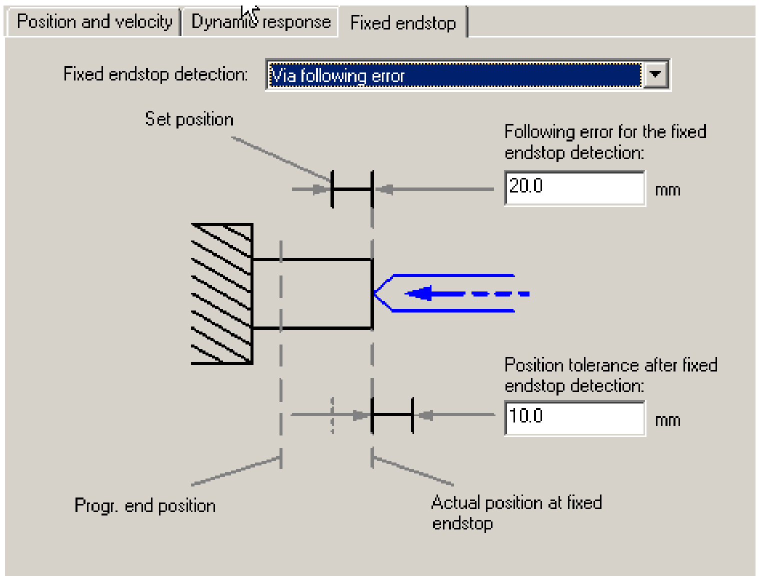
This function requires that torque reduction be supported by the digital coupled drive.

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 endstop 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 message frame/status word (signal word Bit1 for SIEMENS message frame 101, 102, 103, 104, 105, and 106))

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.


Fixed endstop detection: Via following error

Following error for the fixed endstop detection: 20.0 mm

Position tolerance after fixed endstop detection: 10.0 mm

Figure 2-70 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.

Moving to fixed end 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 **_disableAxis()** and **_stopEmergency()** commands cancel the function.

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.

The actual position of the axis is tracked to monitor for a possible breakaway of the fixed endstop, i.e. monitoring of the clamping tolerance window.

If the clamping tolerance window is exited, **alarm 50108: Clamping monitoring error** is triggered and the **travel to fixed endstop** 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 end 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 endstop" (**_enableMovingToEndStop()**) is permitted (and has an override effect), but the reverse is not permitted.

System variables

Table 2- 27 System variables for travel to fixed endstop

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 endstop with fixed endstop detection by means of the following error, the entry for the position tolerance applicable to the fixed end stop detection should be significantly less than the entry for the following error for the fixed endstop detection.

Note

If the "travel to fixed endstop" 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 message frame (i.e. message frame without torque reduction and without actual torque identification)

  The "travel to fixed endstop" functionality is only possible by means of the following error criterion.

In Version 2.0, fixed endstop detection can only be performed by means of the following error. Fixed endstop detection by means of axis torque evaluation is not possible in Version 2.0. As of version 2.1, endstop detection can also be performed using the **limit torque reached** information/identification from the drive.

See also

Overview of torque limiting via torque reduction (Page 150)

2.17 Technology data

Technology data can be specified cyclically from the control 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 (rotary 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
    (indicate the value in the **additiveTorque.value** system variable)
  - Positive torque limit (B+): 16 bits
    (indicate the value in the **torqueLimitPositive.value** system variable)
  - Negative torque limit (B-): 16 bits
    (indicate the value in the **torqueLimitNegative.value** system variable)
- Drive → controller:
  - Actual torque value: 16 bits
    (indicate the value in the **actualTorque.value** system variable)

The technology data are appended as INT to the In/Out drive protocol. This is enabled 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 message frame is specified in the axis wizard. (When you set the axis message frame 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 transmitted 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.

Creating technology data block 1 (e.g. with SINAMICS drive):

The drive has already been configured.

Message frame extension in HW Config (up to V4.1 SP1)

- Set standard message frame for the drive, then add to the PZD via Details:
  - Input: +1 (PZD)
  - Output: +3 (PZD)
- Save HW Config

Message frame 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"
- Press the "Insert line" button and select "Message frame extension"
- Set one word for the input data and three words for the output data
- Press the "Transfer to HW Config" button and confirm the comparison with HW Config
- Press the "Configure message frame" button
- For "PROFIBUS receive direction" and "PROFIBUS send direction", enter the message frame extension as shown in the table below at:
  Project - CPU (e.g. D435) - drive device (e.g. SINAMICS_Integrated) - Drives - Drive_x - Communication - PROFIBUS

Table 2- 28 BICo interconnections

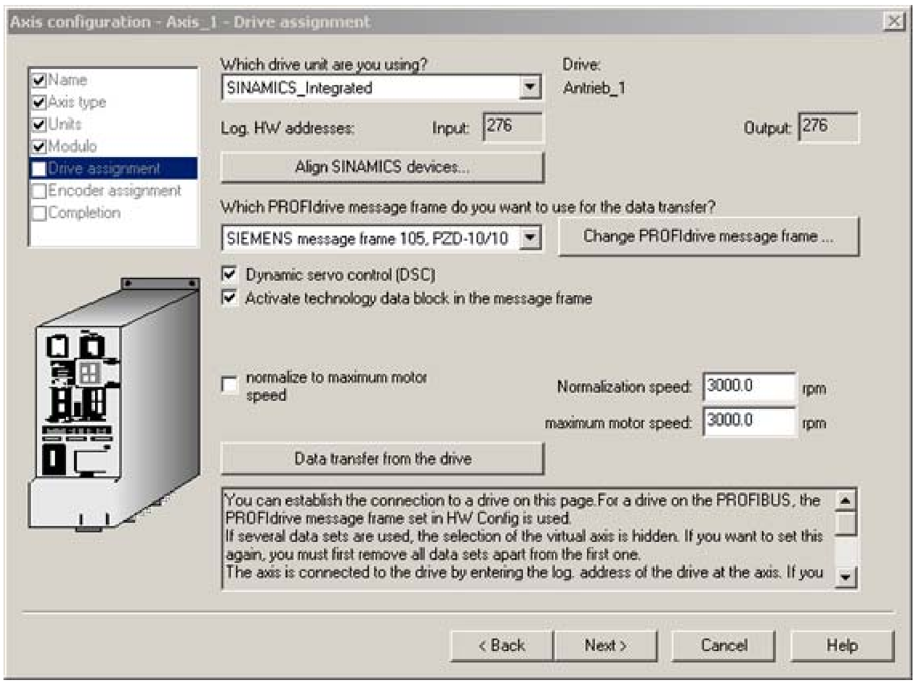
PROFIdrive message frame - Receive direction		
	1. additional PZD (M_Add)	P1511
	2. additional PZD (B+)	P1522
	3. additional PZD (B-)	P1523
PROFIdrive message frame - send direction		
	1. additional PZD (M_act)	E.g. R0080 (actual torque value)

The values are normalized in the drive via P2003.

NOTICE
The P1512 parameter ("Additional torque 1 scaling") must be interconnected with 100% so that M_Add is transferred correctly.

SIMOTION

The technology data block is created when the axis is configured.


Axis configuration - Axis_1 - Drive assignment

☑ Name
☑ Axis type
☑ Units
☑ Modulo
☐ Drive assignment
☐ Encoder assignment
☐ Completion

Which drive unit are you using? SINAMICS_Integrated
Drive: Antrieb_1
Log. HW addresses: Input: 276 Output: 276
Which PROFIdrive message frame do you want to use for the data transfer? SIEMENS message frame 105, PZD-10/10
☑ Dynamic servo control (DSC)
☑ Activate technology data block in the message frame
☐ normalize to maximum motor speed
Normalization speed: 3000.0 rpm
maximum motor speed: 3000.0 rpm

You can establish the connection to a drive on this page.For a drive on the PROFIBUS, the PROFIdrive message frame set in HW Config is used.
If several data sets are used, the selection of the virtual axis is hidden. If you want to set this again, you must first remove all data sets apart from the first one.
The axis is connected to the drive by entering the log. address of the drive at the axis. If you

Align SINAMICS devices...
Change PROFIdrive message frame ...
Data transfer from the drive
< Back
Next >
Cancel
Help

Figure 2-71 Selection of drive and message frame setting in SIMOTION SCOUT

2.18 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 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 limiting values can be predefined on the axis by means of interconnections or directly by means of cyclically effective system variables.

The replacement values can be preset.

The B+ value, B- value and the additive torque can be interconnected by means of a LREAL interface or a LREAL connector, e.g., using a formula object or a controller object.

For more information, refer to SIMOTION Supplementary Technology Objects Description of Functions and the *'LREAL' interconnection interface type* in the **Motion Control Basic Functions** manual.

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
- For linear motors: Force

Commands

The limiting is deactivated by default and has to be explicitly activated.

The torque limits are enabled using the **_enableAxisTorqueLimitPositive()/_enableAxisTorqueLimitNegative()** commands.

The command specifies whether the limits relate to the interconnected value or the replacement value (defaultValue). If they are related to the replacement value, then this is applied cyclically, i.e., a change in the system variables takes effect immediately. If the function is not enabled, then the 100% value is active for B+ and the -100% value for B-.

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()**; instead they must be explicitly disabled via a command.

The interconnection values are disabled using the _disableAxisTorqueLimitPositive()/_disableAxisTorqueLimitNegative() commands.

It is disabled/canceled

- with _disableAxisTorqueLimitPositive() / _disableAxisTorqueLimitNegative()
- In the case of _resetAxis(), _restartAxis(), etc.
- In the case of a transition to STOP

System variables

Table 2- 29 System variables for torque limiting

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	Displays the value that is transmitted to the drive (in user-defined unit)
torqueLimitNegative	
torqueLimitPositiveCommand	Command status / function status
torqueLimitNegativeCommand	

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 should be suppressed, the _setAxisStw() function can be used once to set bit 8 in STW2 for the 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 enabled.

The p2175 (0.0) parameter to suppress the fault message on the drive.

Alarms

20005 "Logical address of technology data block driver not found"

The limiting values and additive values are initialized/transferred regardless of the enable and error status of the axis.

No specific control mechanism is used for the drive.

See also

Setting as a real axis with digital drive coupling (Page 36)

Technology data (Page 159)

2.19 Additive set torque (V3.2 and higher)

An additive set torque can be transmitted to the drive on the axis.

Prerequisite is that an axis with digital drive link is used and that the technology data block is activated.

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 an interconnected value or a replacement value/system variable.

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
- For linear motors: Force

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 2- 30 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 162).

See also

Setting as a real axis with digital drive coupling (Page 36)

2.20 Force/pressure control

2.20.1 Overview of force/pressure control

Force/pressure control requires at least the **positioning axis technology**. If a hydraulic axis is used as a hydraulic speed-controlled axis, force/pressure control is also possible here.

Pressure control and force control are available on the positioning 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 limiting) and anti-windup (I-component) with manipulated variable limiting.

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**.

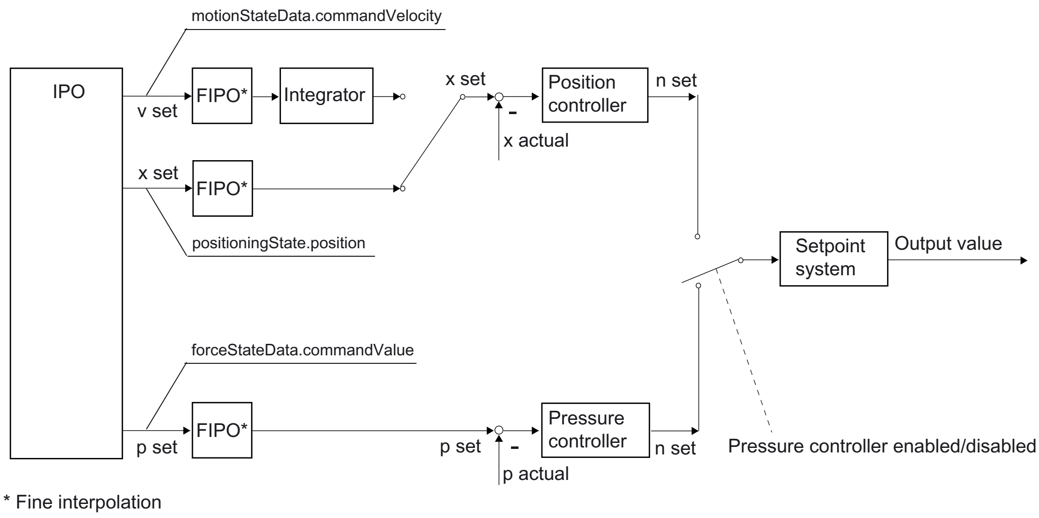


Figure 2-72 Overview of control structure for force/pressure control

---

Note

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

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 259)

2.20.2 Configuration of the actual force/pressure value sensors

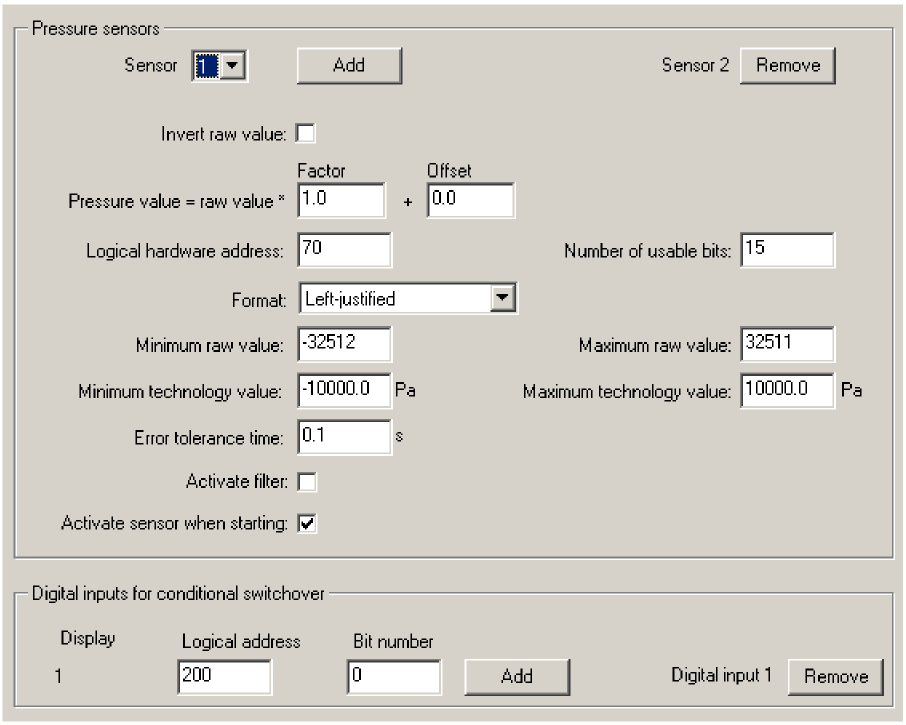
More than one pressure or force sensor can be configured.

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. Prerequisite is the setting **additionalSensorType**:= SET_ACTUAL_VALUE.



Pressure sensors

Sensor 1 — Sensor 2

Invert raw value: ☐

Pressure value = raw value * Factor 1.0 + Offset 0.0

Logical hardware address: 70

Number of usable bits: 15

Format: Left-justified

Minimum raw value: -32512

Maximum raw value: 32511

Minimum technology value: -10000.0 Pa

Maximum technology value: 10000.0 Pa

Error tolerance time: 0.1 s

Activate filter: ☐

Activate sensor when starting: ✓

Digital inputs for conditional switchover

Display	Logical address	Bit number
1	200	0

Digital input 1

Figure 2-73 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.

2.20.3 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.

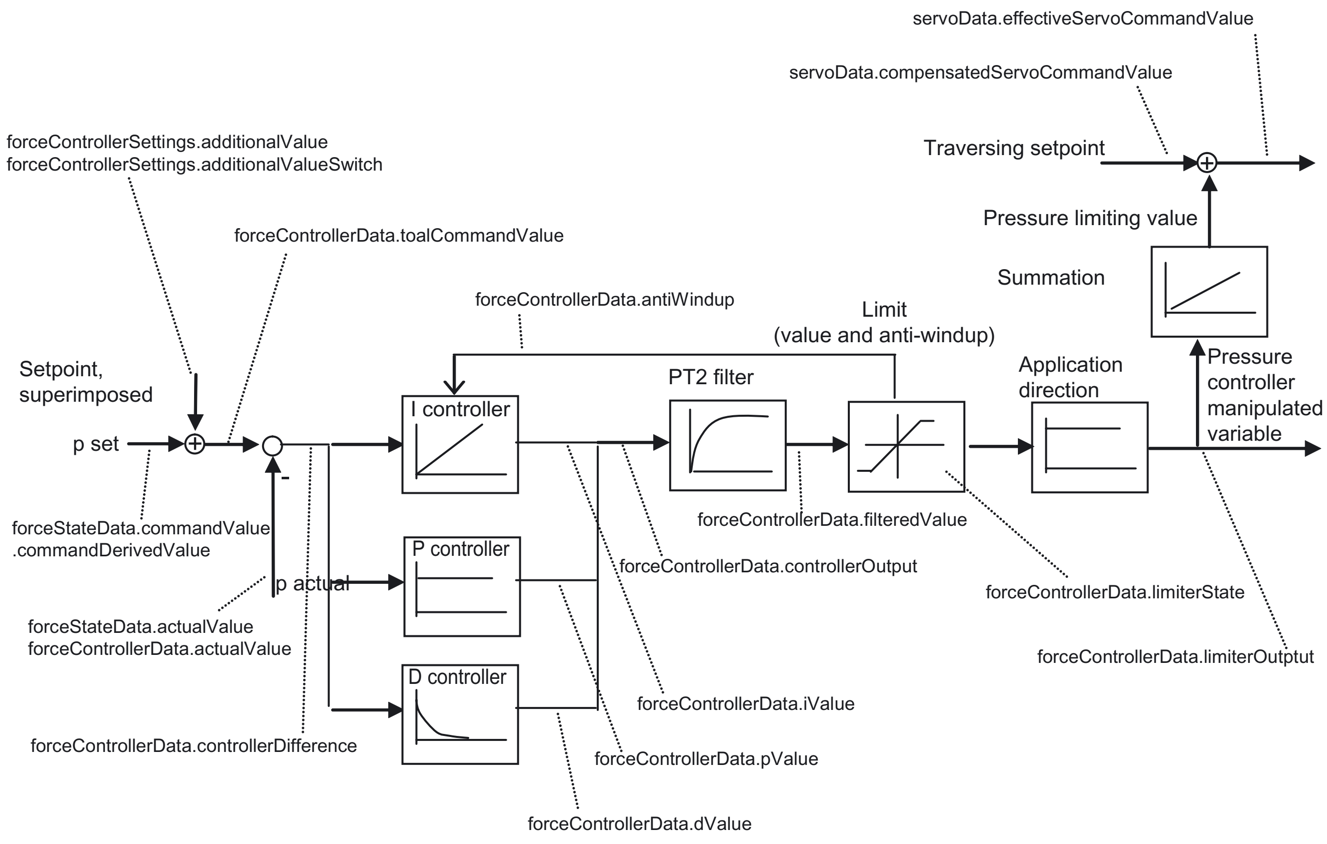
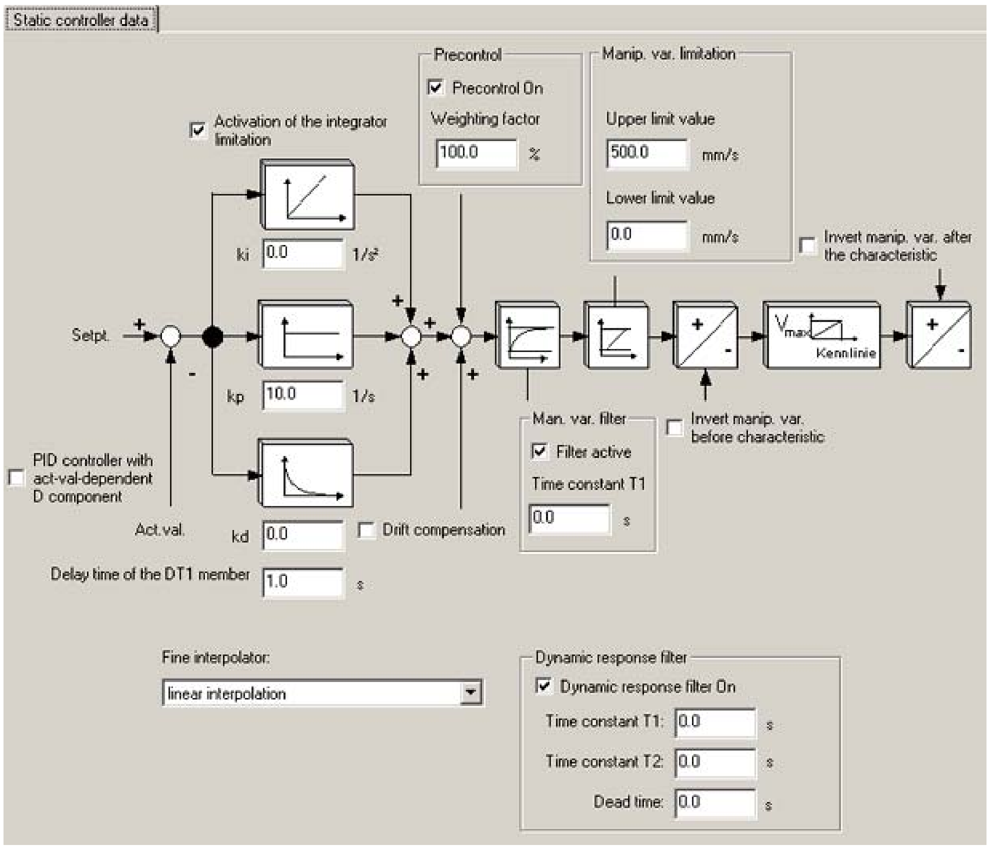

servoData.effectiveServoCommandValue
servoData.compensatedServoCommandValue
Traversing setpoint
forceControllerSettings.additionalValue
forceControllerSettings.additionalValueSwitch
Pressure limiting value
forceControllerData.toalCommandValue
Summation
forceControllerData.antiWindup
Limit
(value and anti-windup)
Setpoint,
superimposed
PT2 filter
Application
direction
Pressure
controller
manipulated
variable
I controller
p set
-
forceStateData.commandValue
.commandDerivedValue
forceControllerData.filteredValue
P controller
forceControllerData.controllerOutput
p actual
forceControllerData.limiterState
forceStateData.actualValue
forceControllerData.actualValue
forceControllerData.limiterOutptut
D controller
forceControllerData.iValue
forceControllerData.controllerDifference
forceControllerData.pValue
forceControllerData.dValue

Figure 2-74 Overview of pressure controller/pressure limiting controller



Precontrol
[✓] Precontrol On
Weighting factor 100.0 %

Manip. var. limitation
Upper limit value 500.0 mm/s
Lower limit value 0.0 mm/s

[ ] Invert manip. var. after the characteristic

[✓] Activation of the integrator limitation

ki 0.0 1/s²

kp 10.0 1/s

kd 0.0

Delay time of the DT1 member 1.0 s

[ ] PID controller with act-val-dependent D component

[ ] Drift compensation

Man. var. filter
[✓] Filter active
Time constant T1 0.0 s

[ ] Invert manip. var. before characteristic

Fine interpolator: linear interpolation

Dynamic response filter
[✓] Dynamic response filter On
Time constant T1: 0.0 s
Time constant T2: 0.0 s
Dead time: 0.0 s

Figure 2-75 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 252)

2.20.4 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 limiting is deactivated.

You must ensure that the maximum deceleration values are set correctly so that the setpoints of the software limit switches are not exceeded in the automatic mode of the system.

Limiting functions are available for:

- Force/pressure setpoint
- Force/pressure controlled variable
- Actual force/pressure value

Note

To stop immediately for active force/pressure control, e.g. via the **_stopEmergency()** command (stopDriveMode:=WITH_COMMAND_VALUE_ZERO), the controller switches to pressure limiting and, with pressure limiting active for the force/pressure setpoint, moves to velocity 0 using the specified velocity profile for active position control.

Note

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 limitation value, the actual pressure value will be removed appropriately.

See also

Dynamic limiting functions (Page 139)

2.20.5 Activating the force/pressure control

Force/pressure control is activated by commands from the application.

SIMOTION supports different activation modes.

- Direct activation

  Direct activation takes place by means of the **forceControlMode** parameter of the axis-enable command (**_enableAxis()** or **_enableQFAxis()**). 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 switched over by reissuing the command before the conditions occur. 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 estimated following error is calculated from the current manipulated variable multiplied with the equivalent time of the position control loop (PTC or configuration data **typeOfAxis.NumberOfDataSets.DataSet_1.DynamicData.positionTimeConstant**).

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.

2.20.6 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.

Command execution takes a certain time even with direct setpoint specification because of the possible transition. For this reason, 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 **derivedValue** refers to derivation of the setpoint over time.

An error message is output for starting at a position outside the profile.

2.20.7 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.

2.20.8 Force/pressure control with velocity limiting

On the positioning axis, velocity limiting (flow rate limiting for hydraulic functionality) can be activated in parallel with the force/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 enabled 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 enabled by means of the **velocityLimitingDirection** parameter (as of V4.1 SP1).

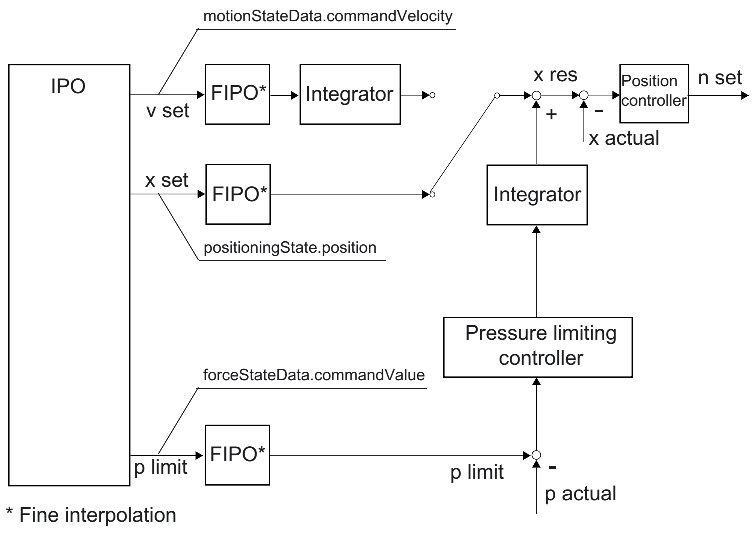
The set direction is indicated in **velocityLimitingCommand.velocityLimitingDirection**.

2.21 Force/pressure limiting

2.21.1 Overview of force/pressure limiting

The force/pressure limiting functionality requires **positioning axis technology**. If a hydraulic axis is used as a speed-controlled 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 enabled until the force/pressure limit is exceeded ($p_{act} > p_{limit}$). 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 enabled 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.

See also

Force/pressure limiting with hydraulic speed-controlled axes with Q valve only (V4.0 and higher) (Page 260)

2.21.2 Positioning with active force/pressure limiting (V3.2 and higher)

With position-related motions and active pressure limiting, cyclic continuation of the interpolator position and velocity can be set.

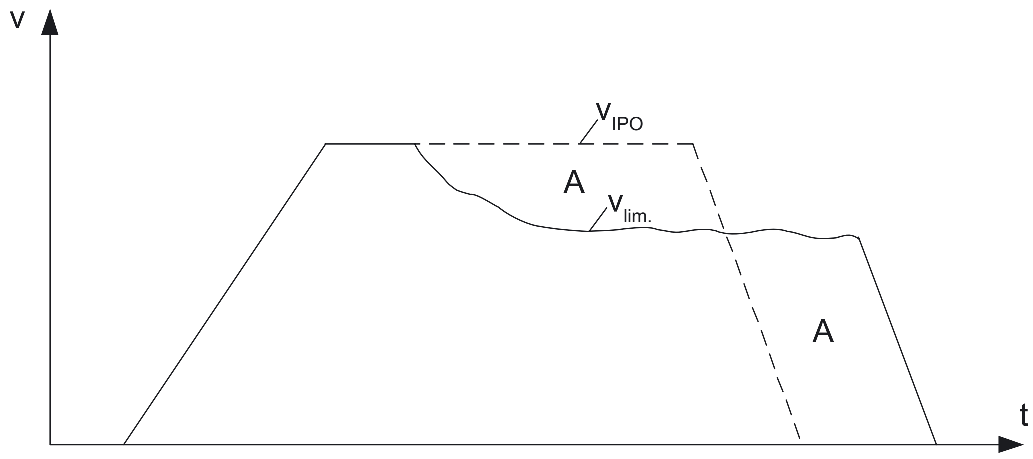


Figure 2-76 Position-related motion with pressure limiting and cyclic continuation

This example shows a v/t profile in which the velocity is reduced by the pressure limiting. 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 limiting remains active.

At standstill, the axis switches from position-controlled to velocity-controlled, after which the axis must be continued again.

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 limiting
  - 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 limiting, 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 limiting. In this case, the torque limiting takes place in the drive, and the setpoints limited by the drive are not known to the controller.

2.21.3 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.

2.22 Data set

2.22.1 Data set overview

The configuration data belonging to the data set of the axis is shown in the dataSetMain configuration data element, depending on the technology set on the axis.

The data set contains the following parameters:

- Parameter of the controller
- Dynamic characteristic values of the cascaded control system
- Parameter of the dynamic compensation of the control loop
- Parameter of the process model
- Parameter of the dynamic following error monitoring
- Parameter of the reference model monitoring
- Measuring system number
- Ratio of the load gear
- Parameter for setting the torque monitoring

Details of the parameters can be found in the TP Cam Configuration Data Reference List.

Data sets contain axis configuration data, especially that pertaining to the servo, for example, controller data, encoder data, process data, data for axis gearing, and data for force/pressure control.

Data sets must be defined and activated as a group because some data, for example, controller data, can only be activated simultaneously in groups to ensure consistency of the controller and function.

2.22.2 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.

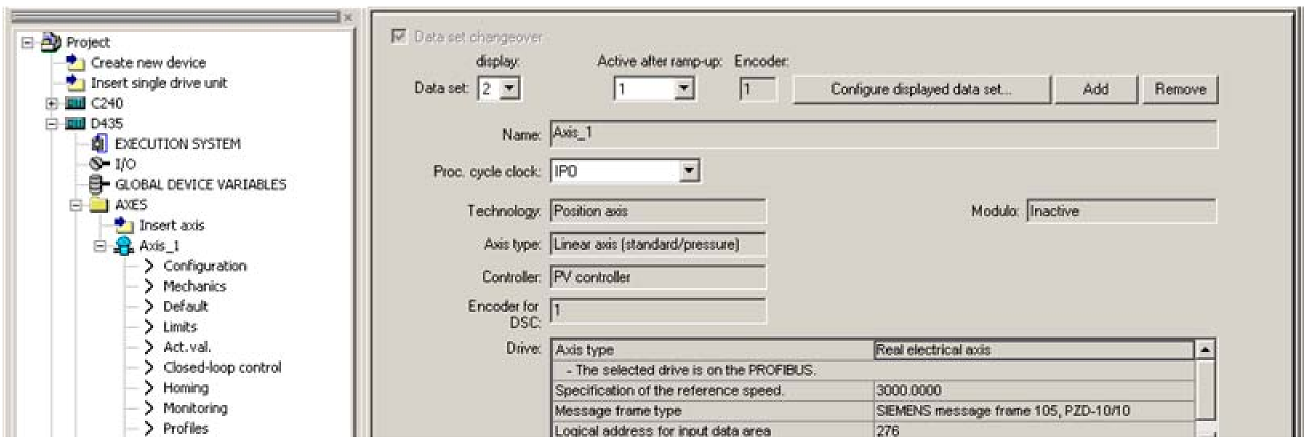
In V4.0 and higher, the smoothingTimeByChangeDifference configuration data element can be used to configure a switchover smoothing filter. This switchover smoothing filter is active for all status transitions/switchovers in which an offset in the manipulated variable can occur due to the switchover. Gear switchover in the data set is not smoothed.

Switchover to another encoder via the data set switchover

The data set shows which encoder is being used. If you activate encoder switchover (in the axis wizard when assigning the encoder), you can specify up to eight different encoders. All configured measuring systems are internally active, and the measured values are updated cyclically.

Data sets and encoders must be specified and selected using their numbers.

The data sets must have the same structure, e.g. 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. An operation with data sets having different structures is not possible (checked during download).



☑ Data set changeover

Data set display: 2	Active after ramp-up: 1	Encoder: 1

Name: Axis_1

Proc. cycle clock: IPO

Technology: Position axis   Modulo: Inactive

Axis type: Linear axis (standard/pressure)

Controller: PV controller

Encoder for DSC: 1

Drive:

Axis type	Real electrical axis
- The selected drive is on the PROFIBUS.	
Specification of the reference speed.	3000.0000
Message frame type	SIEMENS message frame 105, PZD-10/10
Logical address for input data area	276

Figure 2-77 Configuration of data set changeover in SIMOTION SCOUT

Table 2- 31 Data set (data set changeover activated)

display	Here, the number of the data set whose parameters are displayed is shown. If several data sets are available, you can select the number of the data set whose parameters you want to display.
Active	Here, the **active** data set is displayed.
Active after ramp-up	Here, the data set that is **active after ramp-up** is displayed. If several data sets are available, you can select the data set that is active after ramp-up.
Encoder	Here, the **encoder** that you have assigned to the axis is displayed. **Note** You select the encoder in the axis configuration wizard. You can start the wizard in the technology object under **Configuration**.

Note

The data for an encoder can be found in various places in the parameter screen forms. The initial setting is made in the wizard, where the logical addresses are determined from the connection location and the basic encoder data is queried. However, other data, such as the home position offset, is also encoder-dependent.

If you want to make use of the encoder switchover, it is therefore recommended that you use the expert list, since this allows you to see clearly which data is encoder-dependent. Alternatively, you can take the information from the tool tip in the screen forms: This shows the name of the variable, and it is also possible to tell from the tool tip whether the variable is assigned to a specific encoder.

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 2- 32 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.

Activating data sets

Data set commands can be used to read, write, and activate data sets on the axis.

The configuration data belonging to the data set of the axis is shown in the dataSetMain configuration data element, depending on the technology set on the axis.

Data sets must be defined and activated as a group because some data, for example, 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.

2.23 Traversing with user-defined motion and force/pressure profiles

2.23.1 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.

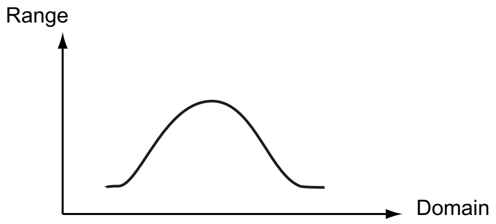


Figure 2-78 Application of cam/technological profiles

A defined start is also possible within a profile.

Application:

Specific adaptation and optimization of traversing specifications.

Additional information can be found in the SIMOTION CamTool Manual and in the SIMOTION SCOUT online help.

Table 2- 33 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 limiting	_runTimeLockedVelocityLimitingProfile()

Position-related velocity limiting profile	Position 1) 2)	Velocity limiting	_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.

2.23.2 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 2- 34 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

2.23.3 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, e.g., for entering the profile and exiting the profile, 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 MotionIn vector using the lowest 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 MotionIn vector using the lowest 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.

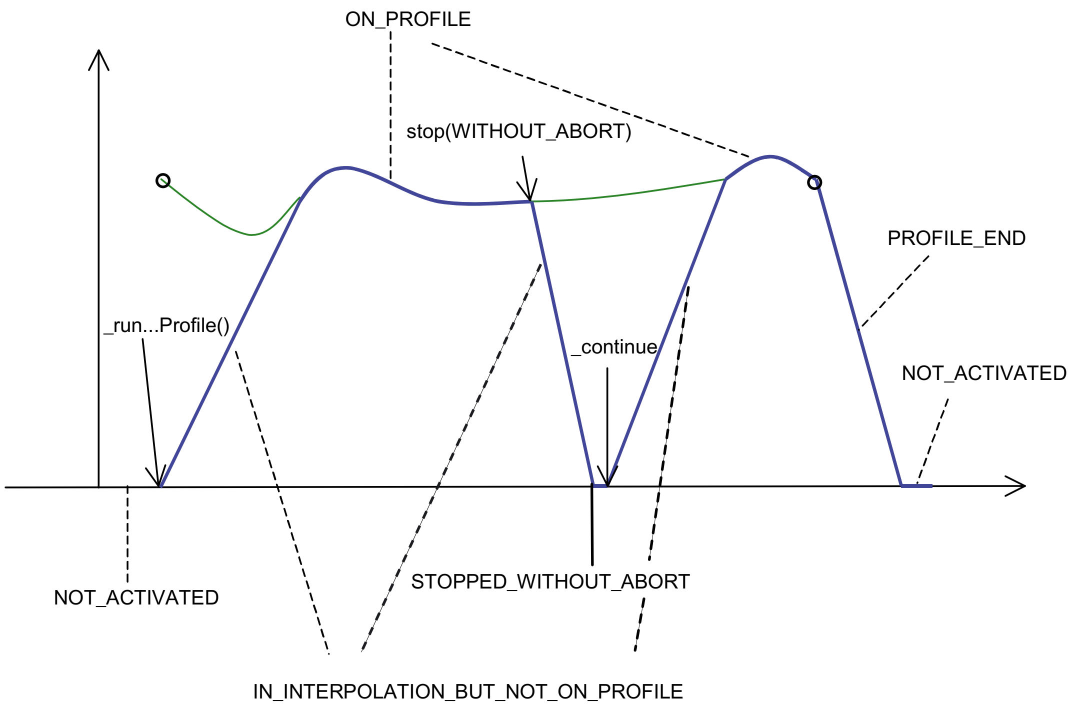


Figure 2-79 Status displays when traveling through a profile (example)

See also

Positioning with user-definable position profile (Page 287)

Enable force/pressure limiting with position-related force/pressure limiting profile (Page 289)

Enabling force/pressure limiting with time-related motion profile (Page 290)

Starting the time-related force/pressure profile (Page 291)

Starting the position-related force/pressure profile (Page 292)

Starting a time-related velocity profile (Page 285)

Starting a position-related velocity profile (Page 286)

Enabling velocity limiting with position-related velocity limiting profile (Page 295)

Enabling velocity limiting with time-related velocity limiting profile (Page 295)

2.23.4 Behavior at the end of the profile (V3.2 and higher)

The decodingConfig.behaviourAtTheEndOfProfile configuration data element is used to configure various behaviors.

Table 2- 35 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.

2.24 Motion commands

2.24.1 Motion execution/interpolator

Axis motion is executed in the interpolator and the servo.

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.

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 controller 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.

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 with the **execution.executionLevel:=SERVO** setting, this enables a very rapid response with motion influence at the user level, as well.

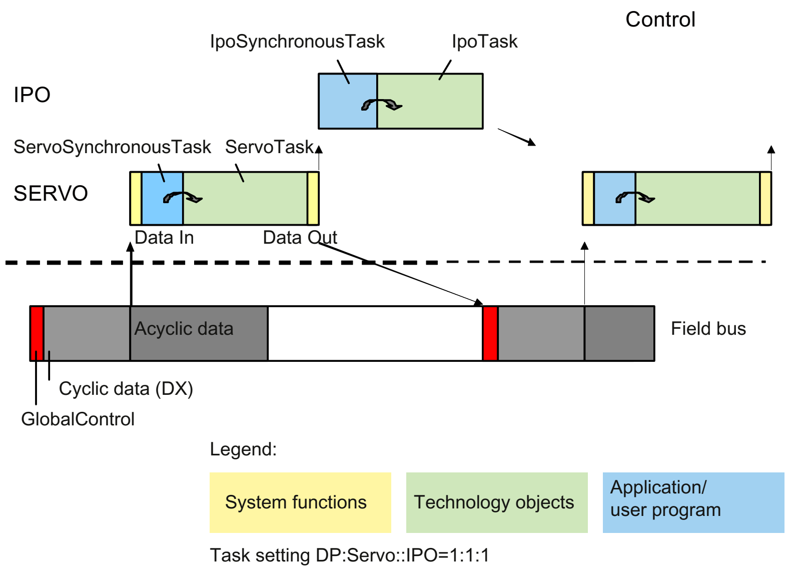


Figure 2-80 Example of cycle clock settings

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.

2.24.2 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 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 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 for enable and disable commands

  This buffer contains the enable and disable commands. The commands displace each other from the command buffer.
- Buffer 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

  These commands overwrite each other if they are issued within one IPO cycle clock.
- Buffer 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.

  For sequential motions with immediate command stepping (nextStep = IMMEDIATELY and mergeMode = SEQUENTIAL), newly entered commands can return with error when the buffer is full.

  For sequential motions with immediate command stepping, when the motion command can be accepted by the system (nextStep:=WHEN_BUFFER_READY and mergeMode:=

SEQUENTIAL), the command stepping waits until the motion command has been accepted by the system.

For mergeMode = NEXT_MOTION or mergeMode = IMMEDIATELY, the command can always be accepted by the system, because the current motion command will be overwritten in the interpolator or the next motion command in the command buffer.

The following table shows the assignment of axis commands to command buffers.

Table 2- 36 Allocation of commands to the command buffers

Command	Position
_stopEmergency() Stop the motion with cancellation of the motion commands without further motion commands having any affect	1
_stop() Stop a motion without traversing command abort	1
_stop() Stop a motion with traversing command abort	4
_continue() Resume motion	1
_enableAxis() Set axis enable	2
_enableQFAxis() Set enable for axis with hydraulic functionality	2
_disableAxis() Remove axis enable	2
_disableQFAxis() Remove enable for axis with hydraulic functionality	2
_enableAxisAdditiveTorque( ) Activate additive set torque	3
_enableAxisSimulation() Enable simulation mode	3
_enableAxisTorqueLimitNegative( ) Activate negative torque limiting	3
_enableAxisTorqueLimitPositive( ) Activate positive torque limiting	3
_disableAxisAdditiveTorque( ) Deactivate additive set torque	3
_disableAxisSimulation() Disable simulation mode	3
_disableAxisTorqueLimitNegative( ) Activate negative torque limiting	3
_disableAxisTorqueLimitPositive( ) Activate positive torque limiting	3
_setAxisDataSetActive() Switch over data set	3

Command	Position
_disableTorqueLimiting() Deactivate torque limiting	3
_enableTorqueLimiting() Activate torque limiting	3
_redefinePosition( ) Reset actual and set position	3
_setAndGetEncoderValue( ) Synchronize measuring systems	3
_enableMonitoringOfEncoder() Activate differential encoder monitoring	3
_disableMonitoringOfEncoder() Deactivate differential encoder monitoring	3
_enableForceControlByCondition() Switch over to p(t) profile with switchover criterion	3
_enablePositionLockedForceLimitingProfile() Activate force/pressure limiting with p(s) profile	3
_enablePositionLockedVelocityLimitingProfile() Activate velocity limiting with position-related limiting profile	3
_enableTimeLockedVelocityLimitingProfile( ) Activate velocity limiting with time-related limiting profile	3
_enableTimeLockedForceLimitingProfile() Activate force/pressure limiting with time-related limiting profile	3
_disableMovingToEndStop() Disabling command for travel to fixed endstop	3
_enableMovingToEndStop() Enabling command for travel to fixed endstop	3
_enableForceLimitingValue() Activate force/pressure limiting with constant force/pressure limiting value	3
_disableForceLimiting() Deactivate force/pressure limiting	3
_enableVelocityLimitingValue() Activate velocity limiting	3
_disableVelocityLimiting() Deactivate velocity limiting	3
_runTimeLockedVelocity() Apply a v(t) profile	4 3 for mergeMode := SUPERIMPOSED
_move() Rotate	4 3 for mergeMode := SUPERIMPOSED
_pos() Position	4 3 for mergeMode := SUPERIMPOSED

Command	Position
_runPositionLockedVelocityProfile() Apply a v(s) profile	4 3 for mergeMode := SUPERIMPOSED
_runTimeLockedPositionProfile() Apply an s(t) profile	4 3 for mergeMode := SUPERIMPOSED
_homing() Homing	3 4 homingMode := ACTIVE_HOMING
_runTimeLockedForceProfile() Run a p(t) profile	4
_runPositionLockedForceProfile() Run a p(s) profile	4
_setForceCommandValue( ) Set force/pressure setpoint	4
_resetAxisError() Reset error on the axis	5
_getStateOfAxisCommand() Read out command status	5
_resetMotionBuffer() Reset command queue	5
_getStateOfMotionBuffer() Shows whether the command queue can be filled	5
_setAxisDataSetParameter() Overwrite data set	5
_getAxisDataSetParameter( ) Read a data set	5
_getMotionStateOfAxisCommand() Read out motion phase of a command	5
_bufferAxisCommandId() Store command status permanently	5
_removeBufferedAxisCommandId() End permanent storage of the command status	5
_resetAxis() Reset axis	5
_getAxisUserPosition( ) Convert from encoder positions to axis coordinate system	5
_getAxisInternalPosition( ) Convert from axis coordinates to encoder position values	5
_setForceControlDataSetParameter() Overwrite data set	5
_getForceControlDataSetParameter() Read a data set	5

Command	Position
_getProgrammedTargetPosition( ) Display programmed absolute end position including superimposition	5
_setQFAxisQCharacteristics() Set characteristics for Q value	5
_setQFAxisPCharacteristics() Set characteristics for P 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

2.24.3 Changing motion commands into the interpolator

With a decodeSequentialMotionCommand configuration data element, you can set when the next SEQUENTIAL or NEXT_MOTION command will change into the interpolator or be executed, immediately in the same cycle clock, or in the next IPO cycle clock.

- If the setting is IMMEDIATELY, the next command is switched to immediately and started in the same cycle clock when interpolation/execution of the current command in the IPO cycle clock has ended.
- 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.

2.24.4 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

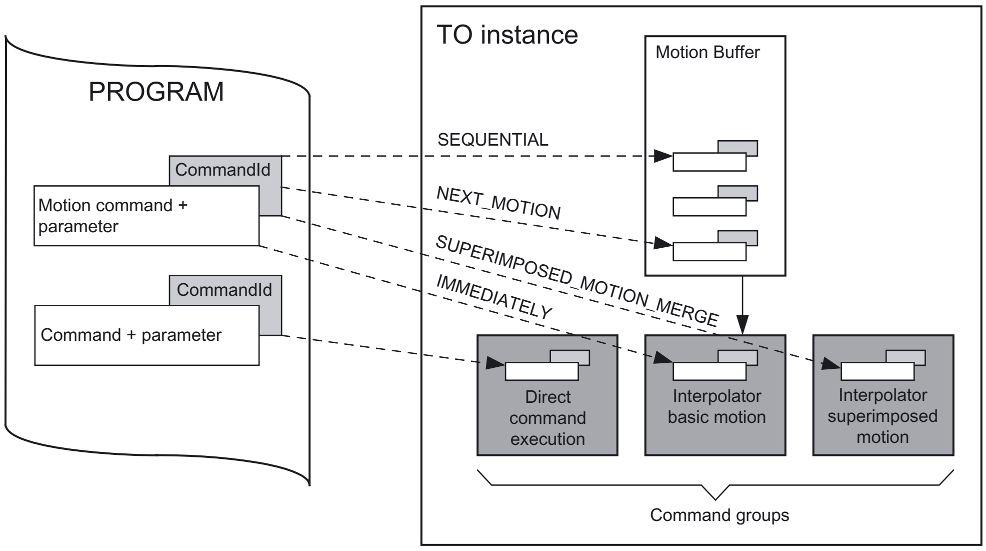


Figure 2-81 Command reactions in the Axis technology object

2.24.5 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)

2.24.6 State model/axis status

Table 2- 37 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

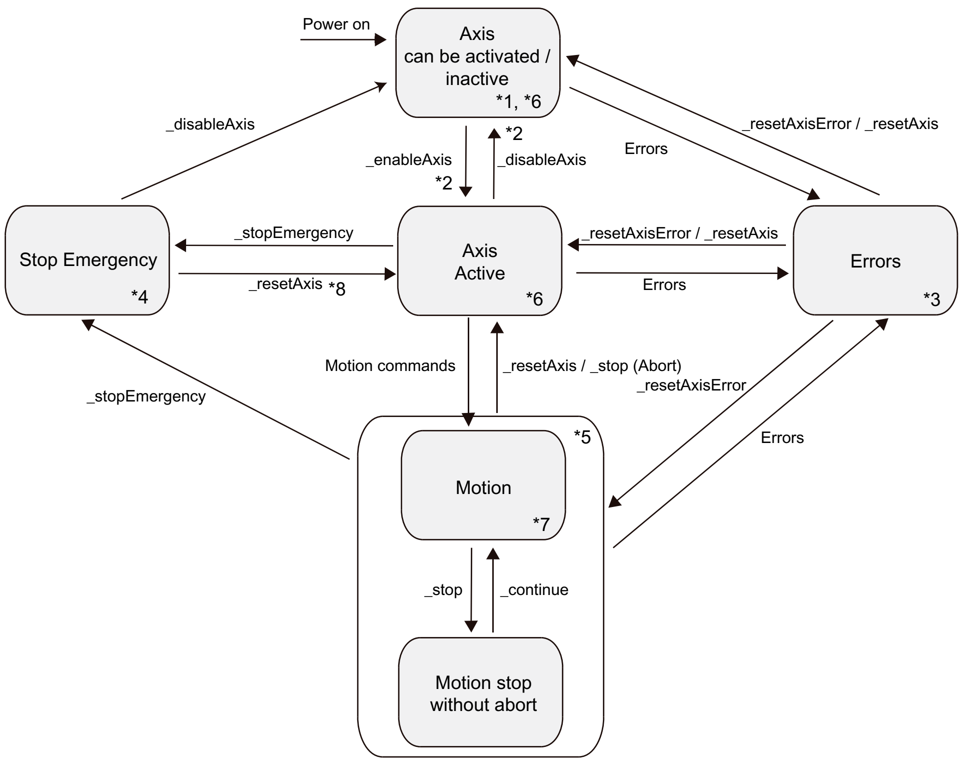


Figure 2-82 State model/axis status

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 2- 38 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 synchronous 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 controller 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 / RUN and vice versa has no effect on the Axis technology object
- During the transition from STOP / STOP U, all alarms that can be reset are acknowledged, and the motion buffer is cleared
- Encoders/actual value system are active in the STOP 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 means of the _stop() command, stopMode :=STOP_WITHOUT_ABORT and resumed by means of 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* active when a following error has built up during active torque limiting
- Is *not* active if a pressure axis switches from closed-loop pressure control to pressure limiting.

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. Commands in the motion buffer and commands pending at the motion buffer are not aborted.

2.25 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 in the *Motion Control Basic Functions* Manual.

2.26 SINAMICS Safety Integrated Extended Functions

No special support of the SINAMICS Safety Integrated BASIC Functions on the axis TO is provided. The standard axis functionality ensures that the drive can independently perform its safety-related stop responses (STO, SS1).

2.26.1 Support of SINAMICS Safety Integrated Extended Functions on the Axis technology object (as of V4.1 SP1)

V4.1 SP1 in SIMOTION supports the *SINAMICS Safety Integrated Extended Functions* for the axis TO in the following form:

- Indication of the status of the safety function in the drive in system variables
- 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)

To use the support of the *SINAMICS Safety Integrated Extended Functions* for the axis TO, proceed as follows:

- Configure the safety functions in the drive (see *SINAMICS Safety Integrated* function manual)
- If the activation of the safety functions is to be made using PROFIsafe, configure the PROFIsafe communication to the higher-level SIMATIC F CPU (see *SINAMICS Safety Integrated* function manual).
- Configure the safety data block as extension in the message frame (for details, refer to the message frame structure)
- Configure the axis

The individual items are described in detail below.

The activation of the *SINAMICS Safety Integrated Extended Functions* data is displayed in the configuration dialog of the axis.

Note

Although SIMOTION does not contain any safety-related functionality, it provides support for SINAMICS drives that can perform safety-related functions.

The purpose of this support from SIMOTION is to prevent stop reactions on the drive side by ensuring for the safety-related monitoring functions that the drive does not exit the monitored operating state.

Table 2- 39 Summary of the supported safety-related functions and stop responses in the drive

	Safety-related function in the drive	Stop response in the drive
STO	Safe Torque Off (safe stop)	STOP A
SS1	Safe Stop 1	STOP B
SS2	Safe Stop 2	STOP C
SOS	Safe Operational Stop	STOP D
SLS	Safely Limited Speed	-
SSM	Safe Speed Monitoring	-

Message frame structure

The SIMOTION support for *SINAMICS Safety Integrated Extended Functions* requires an extension of the actual value message frame with the safety data block (three words). The message frame extension of the input data of the axis message frame is set using the SINAMICS device > Configuration > PROFIBUS message frame dialog.

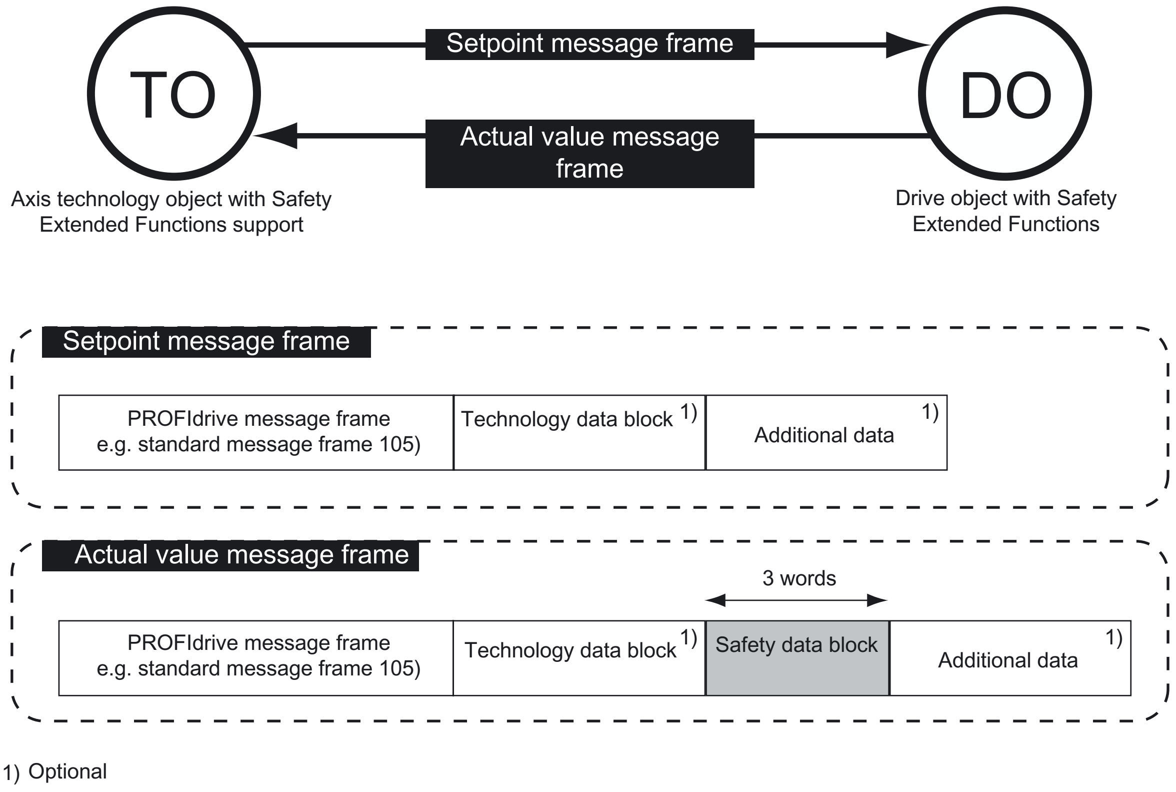

Setpoint message frame
TO
DO
Actual value message frame
Axis technology object with Safety Extended Functions support
Drive object with Safety Extended Functions
Setpoint message frame
PROFIdrive message frame e.g. standard message frame 105)
Technology data block 1)
Additional data 1)
Actual value message frame
3 words
PROFIdrive message frame e.g. standard message frame 105)
Technology data block 1)
Safety data block
Additional data 1)
1) Optional

Figure 2-83 Layout of a drive message frame with safety data block

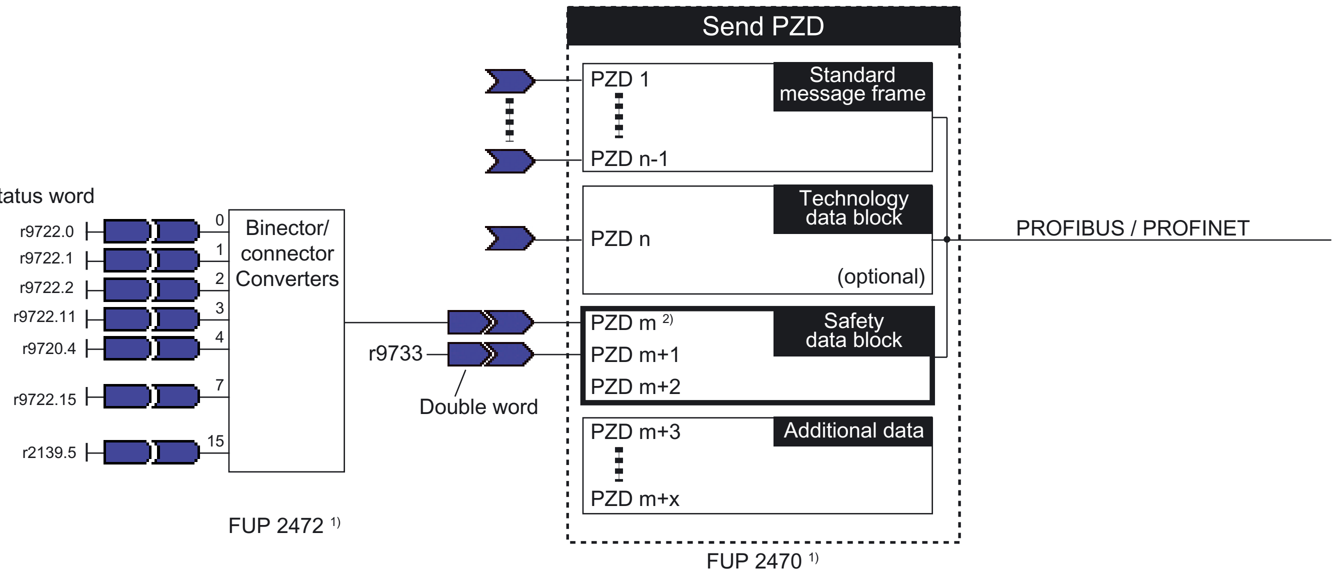
Table 2- 40 Structure of the safety data block in the actual value message frame

Word	Meaning
1	Safety status word
2	Effective absolute set velocity limitation in the drive
3	**Note:** This double word must be interconnected with r9733[0]. (r9733[0] = p9531[x] * p9533 / p9520 x is the selected SLS level) For *Interconnection of the safety data block to the actual value message frame in the drive*, see figure below.

Table 2- 41 Structure of the safety status word in the safety data block

Bit	Status	Name		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)		1-active	r9722.15
8	Reserved			
9	Reserved			
10	Reserved			
11	Reserved			
12	Reserved			
13	Reserved			
14	Reserved			
15	An internal event has occurred and at least one new event is present in the Safety Fault Buffer (see below *Safety message buffer*)	INTERNAL_EVENT	1-active	r2139.5

Configuring the extension of the actual value message frame in SIMOTION SCOUT



1) The "PDF documents" button in the online help can be used to fetch the function charts

2) When technology data block present: m = n + 1
When no technology data block present: m = n

Figure 2-84 Interconnection of the safety data block to the actual value message frame in the drive

The message frame extension is configured using the SINAMICS device > Configuration > PROFIBUS message frame dialog by extending the input data of the axis message frame with three words.
Configure message frame can be used to interconnect the user-defined PZDs using a binector-connector converter.

The logical base address for the safety data block **technologicalData.driveSafetyExtendedFunctionsInfoDataIn.logAddress** results as follows: standard message frame base address + (m-1) * 2
(m is the number of the PZD where the safety data block begins. (See also figure above).

Also refer to the information for creating the technology data block in Section Setting as a real axis with digital drive coupling (Page 36).

Axis configuration

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 data block in **technologicalData.driveSafetyExtendedFunctionsInfoDataIn.logAddress**.
- To support SS2 and STOP C, the status of the pulse enable of the drive on the axis TO must be processed. Transfer this status information appropriate for the message frame type to various positions in the message frame.
  This means the **driveControlConfig.pulsesEnabled.pzd** and **driveControlConfig.pulsesEnabled.bitNumber** configuration data as appropriate for the message frame configuration must be set as follows:
  – Standard message frame 2-6

    pzd = 4; bit = 10 (ZSW2; "enable pulses")

    (possible as of SINAMICS V2.6)
  – Ten SIEMENS message frames

    pzd = 5; bit = 13 (message word; "enable pulses")

Note

- Standard message frame 1 cannot be used.
- Ensure for standard message frames and SIEMENS message frames that p2038 = 0 or p2038 = 1 has been set in the drive.

If a safety commissioning of the *SINAMICS Safety Integrated Extended Functions* has already been performed in the drive 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, the *SINAMICS Safety Integrated Extended Functions* support can be deactivated using the **technologicalData.driveSafetyExtendedFunctionsEnabled**=NO configuration data.

Display

If **technologicalData.driveSafetyExtendedFunctionsEnabled** equals YES, the following information is displayed via system variables:

- **driveData.driveSafetyExtendedFunctionsInfoData.state**

  Display of the safety status word, for the structure, refer to the table above.
  The status word can be used to process the status of the safety functions in the drive in the controller. The status is required for SOS, SLS and SS2 in order to bring the axis from the user program to the permissible operating range. See also Behavior and user responses (Page 207).

- **driveData.driveSafetyExtendedFunctionsInfoData.safeSpeedLimit**

  Effective absolute set velocity limitation in the drive from the safety data block, also refer to the above table for the structure of the safety data block.
  If SLS is selected, **safeSpeedLimit** contains the maximum set velocity at which the axis TO may run.

  Note

  The value in **safeSpeedLimit** is already weighted with the 9533 parameter and does not represent the maximum possible set velocity for p9533 < 100%.

  For activated Safely Limited Speed, the axis must be run below the limit value (and not at the limit value). To reduce the velocity to **safeSpeedLimit**, ensure that the application observes the distance to the actual monitored limit value in the drive. Reduction either on the drive side using the p9533 parameter or using a programming reduction factor.

If **technologicalData.driveSafetyExtendedFunctionsEnabled** equals NO, the *SINAMICS Safety Integrated Extended Functions* are not activated on the axis TO. In the system variable, zero is displayed or the contents are irrelevant.

Note

In order for this information to be updated, the user must interconnect the relevant SINAMICS parameters with the safety data block using BiCo. The exact description of this interconnection using *Configuring the extension of the actual value message frame in SIMOTION SCOUT* is described above.

Messages to the user

These messages are displayed as technological alarms of the *SINAMICS Safety Integrated Extended Functions* on the SIMOTION Axis technology object.

Technological alarms:

- **Alarm 50201: Safety Alarm in the drive (see safety message buffer)**

  A new event is present in the safety message buffer of the drive.

- **Alarm 50202: SINAMICS Safety Integrated Extended Function is selected**

  If safety functions SS2/SOS/SLS are selected, alarm 50202 is signaled. When an automatic transition to SOS is carried out as part of SS2 selection, no alarm is activated.

  The alarm is used only for display in the SCOUT or HMI.

- **Alarm 50203: SINAMICS Safety Integrated Extended Function is deselected**

  If safety functions SS2/SOS/SLS are deselected, alarm 50203 is signaled. When SS2 is deselected, alarm 50203 is signaled only once, although this process also deselects SOS.

  The alarm is used only for display in the SCOUT or HMI.

Note

The alarms are displayed on the Axis technology object only when the *SINAMICS Safety Integrated Extended Functions* support has been configured.

When STO and SS1 are selected, only alarm 20005, reason 4 is signaled, as is also the case with the basic functions. For SS1, the alarm is not issued until the braking ramp has been completed. There is no alarm to signal that the functions have been deselected.

Safety message buffer

Errors in the safety function in the drive itself or stop responses initiated by monitoring functions will be displayed in the drive using a safety fault. The safety faults are stored in the safety message buffer.

Table 2- 42 Structure of the safety message buffer

Parameter	Meaning
r9744	Message buffer change count
r9747 [0…63]	Message code
r9748 [0…63]	Message time arrived in ms
r9749 [0…63]	Additional information
r9752	Messages counter
r9754 [0…63]	Message time arrived in days
r9755 [0…63]	Message time sent in ms
r9756 [0…63]	Message time sent in days

Safety faults may only acknowledge on a fail-safe basis, i.e.
from the Safety PLC using the PROFIsafe Internal_Event_Acknowledge signal - bit 7 of the PROFIsafe control word or using a terminal on the TM54F.

The following restriction applies to SINAMICS V2.5:
Fail-safe acknowledgment can be performed if there is an SLS violation followed by STOPC - SS2. In other cases, the acknowledgment must be issued by means of power off/power on.

For further information, see *SINAMICS Safety Integrated* function manual.

An entry in the safety message buffer is indicated with INTERNAL_EVENT (bit 15) in the **driveData.driveSafetyExtendedFunctionsInfoData.state** system variable. If this bit is set, the **50201 Safety Alarm in the drive** alarm will be initiated in SIMOTION.

The user program can fetch the fault cause from the p9744-p9756 drive parameters and, for example, display on an HMI.

See also

Technology data (Page 159)

2.26.2 Behavior and user responses

The safety functions in the drive are either selected or deselected using a Safety PLC (e.g. SIMATIC S7 CPU317F) using a safe PROFIsafe communication or using a safety-related SINAMICS TM54F terminal module. The selection and deselection of the safety functions is made independent of the Motion Control user program.

The SOS, SLS, SS1, and SS2 safety functions contain monitoring functions that must be supported by the SIMOTION user program in order to place or keep the drive in the monitored operating mode. If the monitored limits are exceeded, this causes a safety-related stop response in the drive.

The user can use the **driveData.driveSafetyExtendedFunctionsInfoData.state** system variable to evaluate the safety status word.

Safe Torque Off (STO) and Safe Stop 1 (SS1)

When the **STO** is selected, pulse suppression will be activated directly in the drive: A moving motor coasts to standstill.

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-independent response for STO or SS1 will be indicated with the 20005 reason 4 TO alarm on the Axis technology object.

If support for the Extended Functions has been configured on the axis (see Support of SINAMICS Safety Integrated Extended Functions on the Axis technology object (as of V4.1 SP1) (Page 201)), the status for STO or SS1 can be read in the **~.state** system variable in bit 0 = TRUE or bit 1 = TRUE.

Also, if support for the Extended Functions has been configured on the axis, SS1 is selected, and **technologicalData.driveSafetyExtendedFunctionsEnabled**=YES, the current motion on the axis must be aborted, as for SS2, and the position setpoint switched to follow-up mode, using the **_stop()** command 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.

When the pulse suppression is achieved, the drive will be disabled and the status of the power enable set in the **actorMonitoring.power**=INACTIVE system variable.

Additional response in the user program:

If the drive has been disabled using STO or SS1 (**actorMonitoring.power**=INACTIVE), the SIMOTION user program can use the **_enableAxis()** command to re-enable the drive. An enable, however, is possible only when STO or SS1 has been deselected again (safe switch-on disable).

Selection or deselection of the SS2, SOS and SLS Safety Extended Function

For the SS1, SS2, SOS, and SLS status transitions, an appropriate response for the technology object must be programmed in the user program so the monitored safety-related conditions (e.g. maximum velocity, standstill) are satisfied in the drive. The user program can detect status transitions by cyclically processing the **~.state** system variable.

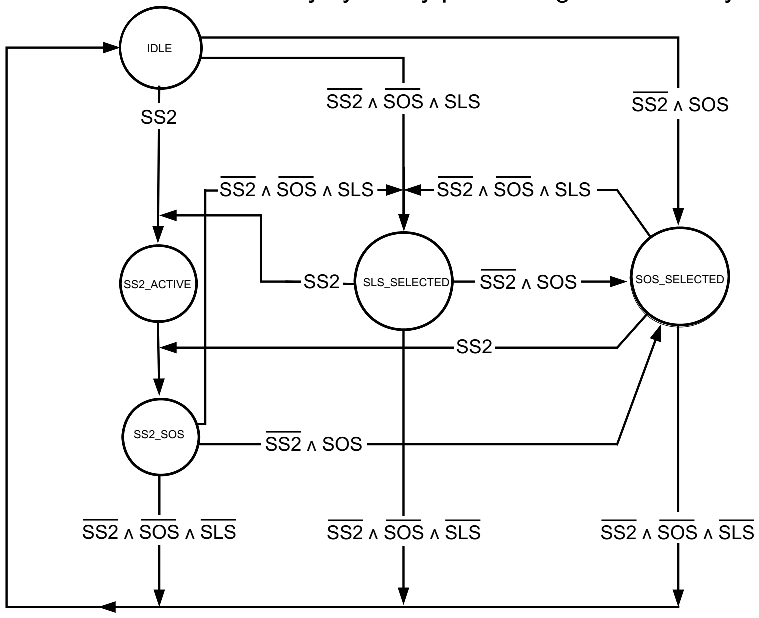

IDLE
SS2
$\overline{SS2} \wedge \overline{SOS} \wedge SLS$
$\overline{SS2} \wedge SOS$
$\overline{SS2} \wedge \overline{SOS} \wedge SLS$
$\overline{SS2} \wedge \overline{SOS} \wedge SLS$
SS2_ACTIVE
SS2
SLS_SELECTED
$\overline{SS2} \wedge SOS$
SOS_SELECTED
SS2
SS2_SOS
$\overline{SS2} \wedge SOS$
$\overline{SS2} \wedge \overline{SOS} \wedge \overline{SLS}$
$\overline{SS2} \wedge \overline{SOS} \wedge \overline{SLS}$
$\overline{SS2} \wedge \overline{SOS} \wedge \overline{SLS}$

Figure 2-85 Status diagram

Table 2- 43 States of the SS2, SOS and SLS safety-related functions

Status word	Safety status word	Drive response	Response in the user program
SLS_SELECTED	Bit 2 = 0 Bit 3 = 0 Bit 4 = 0	Drive monitored for the maximum permissible velocity after expiration of the delay time	Limit velocity setpoint
SS2_ACTIVE	Bit 2 = 1 Bit 3 = 0 Bit 4 = x	Drive brakes independently and after the delay time switches to SS2_SOS	Terminate motion and follow-up operation
SS2_SOS	Bit 2 = 1 Bit 3 = 1 Bit 4 = x	Monitoring of the operational stop	Follow-up mode
SOS_SELECTED	Bit 2 = 0 Bit 3 = 1 Bit 4 = x	Drive monitors the operational stop after expiration of the delay time	Bring to a standstill and remain in this state
IDLE	Bit 2 = 0 Bit 3 = 0 Bit 4 = 1	The SLS, SS2 and SOS safety-related functions are not active	No restrictions on the axis because of a safety-related function

]

Safe Stop 2 (SS2)

When Safe Stop 2 is selected, the drive independently decelerates on the OFF3 ramp speed-controlled (n = 0) and enters the monitored standstill (SOS). The drive then decouples itself from the SIMOTION setpoint interface.

Response in the user program:

When SS2 (bit 2 of **~.state**=TRUE) is selected, the current axis motion must be canceled and the position setpoint switched to follow-up operation, for example, with the **_stop()** command and movingMode=SPEED_CONTROLLED.

When SS2 is deselected (bit 2 = FALSE), the standstill monitoring in the drive will be ended. If no other safety function is active, the axis can be traversed again.

Safe Operating Stop (SOS)

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 (bit 3 of **~.state**=TRUE), bring the drive to a standstill within the time parameterized at p955/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.

Safely Limited Speed (SLS)

When SLS is selected, the drive continues to follow the setpoint interface and activates the velocity monitoring after the delay time configured in the drive.

Response in the user program:

When SLS is selected (system variable **~.state** bit 4 = **FALSE**), reduce the velocity to the maximum velocity defined by the **safeSpeedLimit** system variable.

When SLS is deselected, the drive deactivates the velocity monitoring. The drive can traverse again without restriction provided no other safety function is active.

- **For selected SLS, the drive is in the monitored standstill (SOS or SS2)**

  The drive is in the monitored standstill (by a previous selection of SS2 or SOS). This means a motion can be performed only when SS2 and SOS are not active (bit 2 and bit 3 for **~.state**=FALSE).

  Response in the user program:

  When SS2 and SOS are deselected, a motion is permitted immediately with the displayed maximum velocity **safeSpeedLimit**.

- **Axis performs motion**

  When SLS is selected, the axis traverses with a higher velocity than preselected with SLS.

  Response in the user program:

  When SLS is selected (with stop function also active), the axis must be reduced to the maximum velocity displayed as **safeSpeedLimit**.

- **Modification of velocity limitation**

  The velocity limit value monitored in the drive can be changed dynamically. A reduction of the limit value is critical because the deceleration must be performed in the user program to prevent violation of the new limit value.

  Response in the user program:

  If SLS is active (and all stop functions are inactive), the **safeSpeedLimit** system variable will be monitored for change:

  – **new safeSpeedLimit > old safeSpeedLimit:** Higher velocity permitted immediately

  – **new safeSpeedLimit > old safeSpeedLimit:** Deceleration and limitation to the new limit value

Configuring an axis 3

3.1 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 box for the purpose of data set changeover. If data set changeover is activated, you can:

- Create new data records
- Delete data records
- Define which data record to load for the technology object during CPU power up
- Reconfigure the selected data record

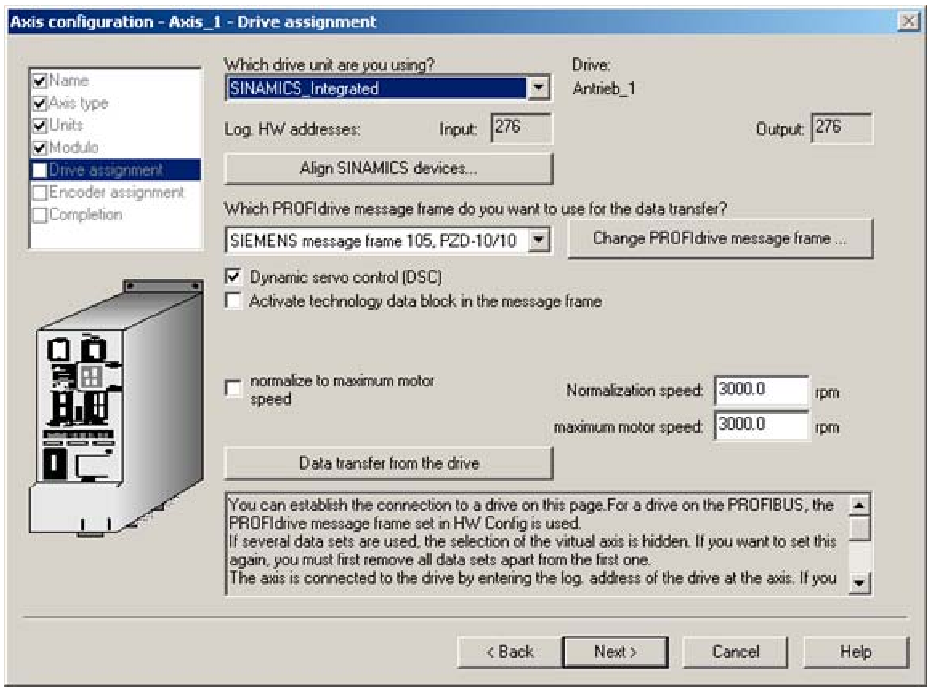
For more information, see Data set (Page 178).

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.

3.2 Linking digital drives

3.2.1 Overview of linking digital drives

Digital drives are linked to the SIMOTION Axis technology object via a PROFIdrive message frame.


Axis configuration - Axis_1 - Drive assignment

☑ Name
☑ Axis type
☑ Units
☑ Modulo
☐ Drive assignment
☐ Encoder assignment
☐ Completion

Which drive unit are you using? SIMANICS_Integrated
Drive: Antrieb_1
Log. HW addresses: Input: 276 Output: 276
Align SINAMICS devices...
Which PROFIdrive message frame do you want to use for the data transfer? SIEMENS message frame 105, PZD-10/10
Change PROFIdrive message frame ...
☑ Dynamic servo control (DSC)
☐ Activate technology data block in the message frame
☐ normalize to maximum motor speed
Normalization speed: 3000.0 rpm
maximum motor speed: 3000.0 rpm
Data transfer from the drive

You can establish the connection to a drive on this page.For a drive on the PROFIBUS, the PROFIdrive message frame set in HW Config is used.
If several data sets are used, the selection of the virtual axis is hidden. If you want to set this again, you must first remove all data sets apart from the first one.
The axis is connected to the drive by entering the log. address of the drive at the axis. If you

< Back | Next > | Cancel | Help

Figure 3-1 SINAMICS drive assignment

See also

Setting as a real axis with digital drive coupling (Page 36)

3.2.2 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*.

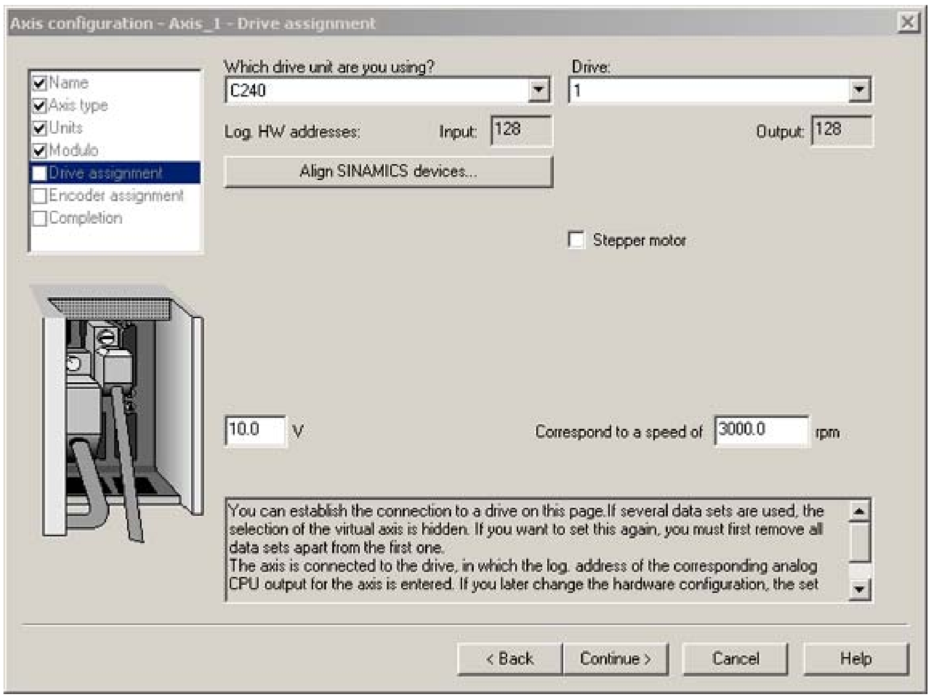
3.3 Linking analog drives to SIMOTION

Note

Note the following information regarding the 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.



Which drive unit are you using?: C240
Drive: 1
Log. HW addresses: Input: 128 Output: 128
☐ Stepper motor
10.0 V Correspond to a speed of 3000.0 rpm

You can establish the connection to a drive on this page.If several data sets are used, the selection of the virtual axis is hidden. If you want to set this again, you must first remove all data sets apart from the first one.
The axis is connected to the drive, in which the log. address of the corresponding analog CPU output for the axis is entered. If you later change the hardware configuration, the set

Figure 3-2 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. Also refer to Setting as a real axis with digital drive coupling (Page 36).

For more information, refer to the *ADI4 - Analog Drive Interface for 4 Axes Manual* and the *Distributed I/Os IM 174 PROFIBUS Module Manual*.

See also

Setting as a real axis with digital drive coupling (Page 36)

Setting as a real axis with analog drive link (Page 35)

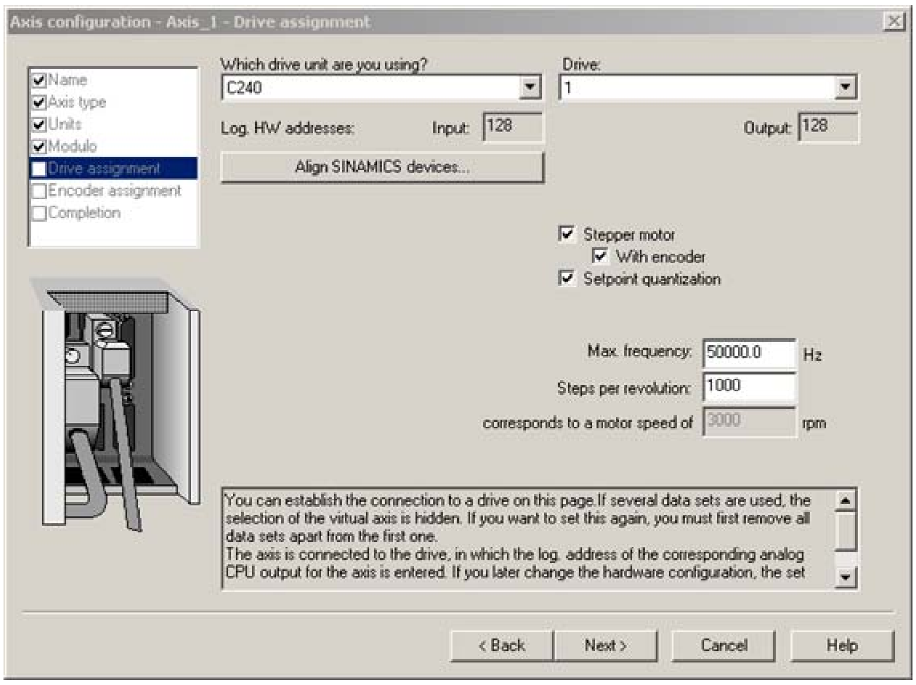
3.4 Axis with stepper motor connection

Note

Please note the following information regarding the 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 123)).


Axis configuration - Axis_1 - Drive assignment

- ☑ Name
- ☑ Axis type
- ☑ Units
- ☑ Modulo
- ☐ Drive assignment
- ☐ Encoder assignment
- ☐ Completion

Which drive unit are you using? C240

Drive: 1

Log. HW addresses: Input: 128 Output: 128

Align SINAMICS devices...

- ☑ Stepper motor
  - ☑ With encoder
- ☑ Setpoint quantization

Max. frequency: 50000.0 Hz

Steps per revolution: 1000

corresponds to a motor speed of 3000 rpm

You can establish the connection to a drive on this page.If several data sets are used, the selection of the virtual axis is hidden. If you want to set this again, you must first remove all data sets apart from the first one.
The axis is connected to the drive, in which the log. address of the corresponding analog CPU output for the axis is entered. If you later change the hardware configuration, the set

< Back | Next > | Cancel | Help

Figure 3-3 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 a stepper motor link is configured without an encoder, an encoder input is still automatically reserved and cannot be used, for example, for an external encoder.

With a rotary 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 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 Setting as a real axis with digital drive coupling (Page 36).

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

See also

Position control (Page 97)

Setting as a real axis with stepper drive C2xx (V3.2 and higher) (Page 43)

Stepper drives (Page 123)

3.5 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.

As of V4.1 SP1, the **Selected Parameters** tab of 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, rotary, and positioning axis types, the following object of a synchronous 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. In the menu bar, click **Axis > Expert > Expert list**. 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.

3.6 Automatic controller setting

3.6.1 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 controller.

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 223).

- 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, S120 - CU320, and S120 - CU310.

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 220))
2. Set the DSC position controller
(see also Automatic position controller setting (as of V4.1 SP1) (Page 223))

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

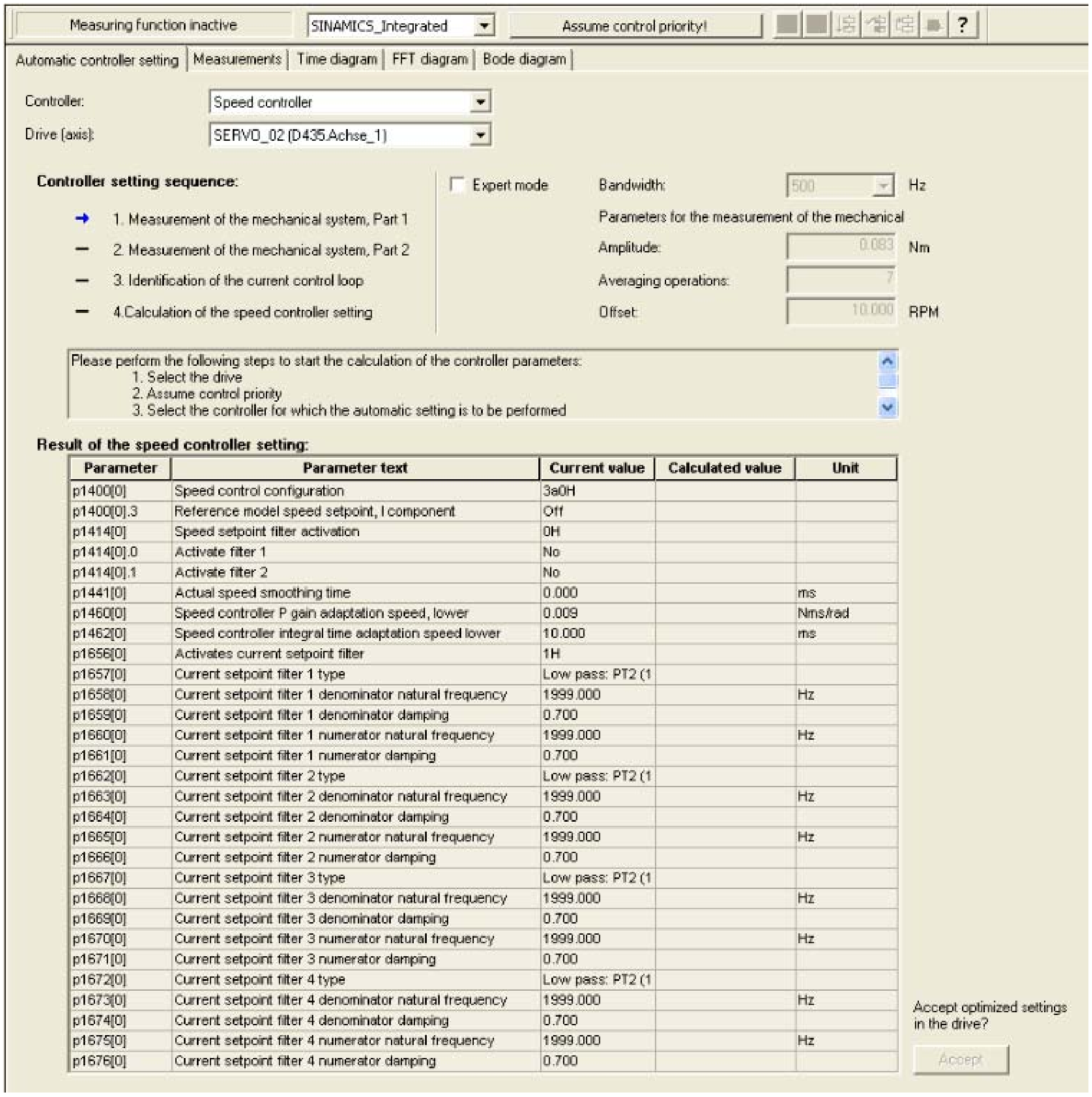
Overview of commissioning the position controller of positioning axes (Page 127)

Assigning automatic controller optimization (Page 305)

3.6.2 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.



Controller: Speed controller

Drive (axis): SERVO_02 (D435.Achse_1)

Controller setting sequence:

➜ 1. Measurement of the mechanical system, Part 1
— 2. Measurement of the mechanical system, Part 2
— 3. Identification of the current control loop
— 4.Calculation of the speed controller setting

Bandwidth:	500	Hz
Parameters for the measurement of the mechanical		
Amplitude:	0.083	Nm
Averaging operations:	7	
Offset:	10.000	RPM

Please perform the following steps to start the calculation of the controller parameters:
1. Select the drive
2. Assume control priority
3. Select the controller for which the automatic setting is to be performed

Result of the speed controller setting:

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	Low pass: PT2 (1		
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	Low pass: PT2 (1		
p1663[0]	Current setpoint filter 2 denominator natural frequency	1999.000		Hz
p1664[0]	Current setpoint filter 2 denominator damping	0.700		
p1665[0]	Current setpoint filter 2 numerator natural frequency	1999.000		Hz
p1666[0]	Current setpoint filter 2 numerator damping	0.700		
p1667[0]	Current setpoint filter 3 type	Low pass: PT2 (1		
p1668[0]	Current setpoint filter 3 denominator natural frequency	1999.000		Hz
p1669[0]	Current setpoint filter 3 denominator damping	0.700		
p1670[0]	Current setpoint filter 3 numerator natural frequency	1999.000		Hz
p1671[0]	Current setpoint filter 3 numerator damping	0.700		
p1672[0]	Current setpoint filter 4 type	Low pass: PT2 (1		
p1673[0]	Current setpoint filter 4 denominator natural frequency	1999.000		Hz
p1674[0]	Current setpoint filter 4 denominator damping	0.700		
p1675[0]	Current setpoint filter 4 numerator natural frequency	1999.000		Hz
p1676[0]	Current setpoint filter 4 numerator damping	0.700		

Figure 3-4 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 218).

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.

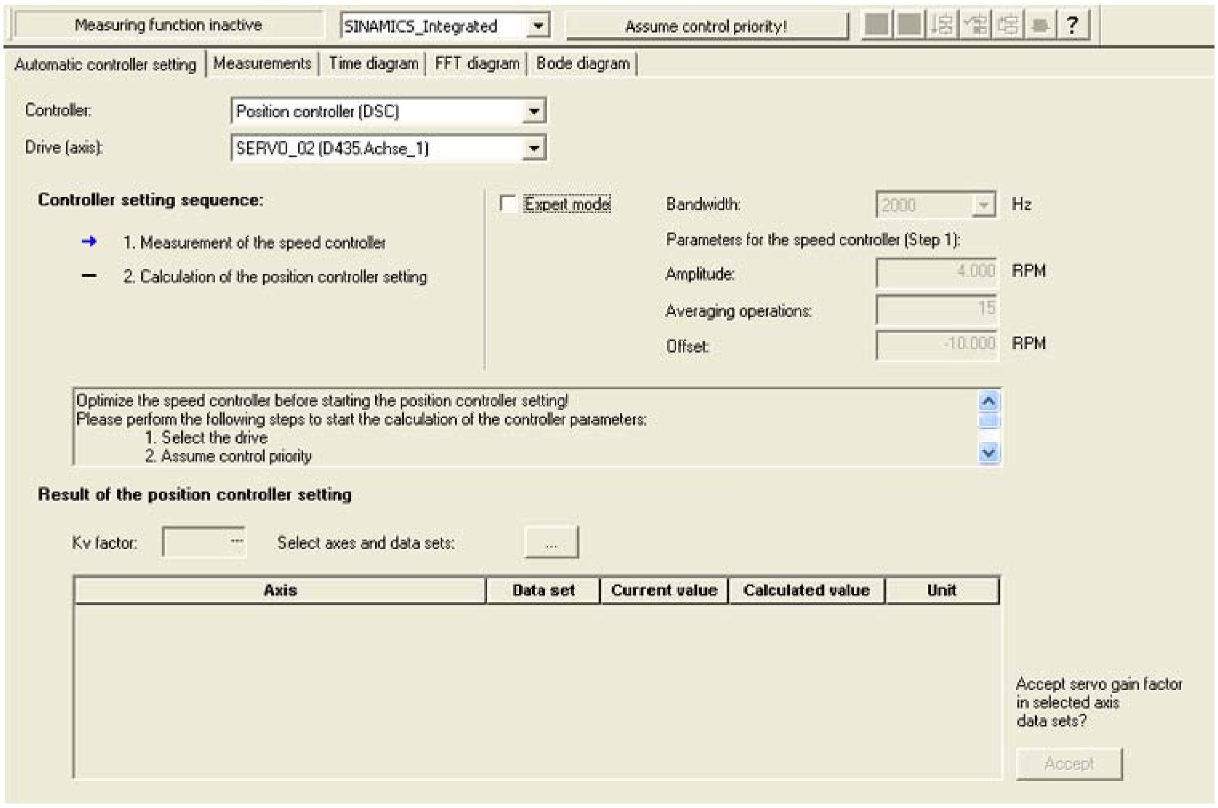
See also

SIMOTION measuring functions (Page 226)

Assigning automatic controller optimization (Page 305)

3.6.3 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.



Controller: Position controller (DSC)

Drive (axis): SERVO_02 (D435.Achse_1)

Controller setting sequence:

→ 1. Measurement of the speed controller

– 2. Calculation of the position controller setting

☐ Expert mode

Bandwidth:	2000	Hz
Parameters for the speed controller (Step 1):		
Amplitude:	4.000	RPM
Averaging operations:	15	
Offset:	-10.000	RPM

Optimize the speed controller before starting the position controller setting!
Please perform the following steps to start the calculation of the controller parameters:
1. Select the drive
2. Assume control priority

Result of the position controller setting

Kv factor: --- Select axes and data sets: ...

Axis	Data set	Current value	Calculated value	Unit

Accept servo gain factor in selected axis data sets?

Accept

Figure 3-5 Automatic controller setting - Position controller example

Requirements and boundary conditions

In addition to the General requirements (Page 218) for the automatic controller setting, the following requirements/boundary conditions apply:

- DSC is required for the position controller setting.
  - You are using a message frame that supports DSC (message frame 5, 6, 105, or 106).
  - If you have selected a message frame which does not support DSC, you must select one of the message frames mentioned above instead.

    As various presettings are required for DSC that can only be set when running through the axis wizard for the first time, you have to make these settings manually:

    - Static controller data: Precontrol On with weighting factor 100% (NumberOfDataSets.DataSet_1.ControllerStruct.PV_Controller.preCon = YES and NumberOfDataSets.DataSet_1.ControllerStruct.PV_Controller.kpc = 100%)

    - Static controller data: Balancing filter (extended balancing filter active) (NumberOfDataSets.DataSet_1.ControllerStruct.PV_Controller.balanceFilterMode = MODE_2)

    - Static controller data: Fine interpolator (constant-acceleration interpolation) (FineInterpolator._type = CUBIC_MODE)

    - Dynamic controller data: Equivalent time for speed control loop 0.0 and equivalent time for position control loop 0.0 (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 with 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 check is performed.
  Align the configurations of the SIMOTION control with the drive, using the "Data transfer from the drive" button in the axis wizard, for example.
  For SINAMICS, the following parameters are used in the drive:
  P2000: Normalization speed
  P1082: Maximum speed
  See also Setting as a real axis with digital drive coupling (Page 36).
- 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 replacement time constant of the position controller must be changed manually by the user (PostionTimeConstant = 1/Kv).
- A load-side vibration is not considered 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 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 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 save 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 Actual to RAM function (*Target System - Copy Actual 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 226)

Assigning automatic controller optimization (Page 305)

3.7 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.

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 3- 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_QFAXIS_WITH_OPEN_LOOP_FORCE_CONTROL)
5	Hydraulic axis with pressure/force control (Q valve) (REAL_QFAXIS_WITH_CLOSED_LOOP_FORCE_CONTROL)
6	Real hydraulic axis with force/pressure specification (P-valve) (REAL_QFAXIS_WITH_OPEN_LOOP_FORCE_CONTROL_ONLY)
7	Real electric axis with master value output via encoder signal simulation (REAL_AXIS_WITH_SIGNAL_OUTPUT)

Note

Prerequisites

- 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 the 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.

Table 3- 2 Possible signal types

Signal type
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 3- 3 Possible superimposition points (internalServoSettings.~ structure)

Name	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 108)

Manipulated variable superimposition (Page 116)

3.8 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. The control panel can be used for the following tasks, for example:

- Performing a function test on individual axes
- Traversing axes during commissioning before the user program is available or if only parts of the user program are available
- Traversing the axis for optimization purposes (controller setting)
- Axis homing
- Enabling and disabling an axis
- Axis homing or absolute encoder adjustment
- Acknowledging axis alarms

⚠ WARNING
You must observe the relevant safety notices when implementing this function. Failure to do so can result in personal injury and property damage.

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).
- The SIMOTION device must be in STOP_U mode.

You can find further information in the SCOUT online help ("Axis control panel" index) and the SIMOTION D4x5 Commissioning Manual.

Part II Hydraulic Functionality 4

4.1 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 19)

General information about axes (Page 15)

5 Fundamentals of hydraulic functionality

5.1 Axis settings / drive assignment

5.1.1 Overview of axis settings / drive assignment

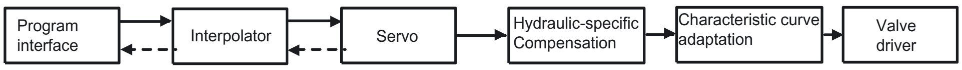

Program interface
Interpolator
Servo
Hydraulic-specific Compensation
Characteristic curve adaptation
Valve driver

Figure 5-1 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)

5.1.2 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

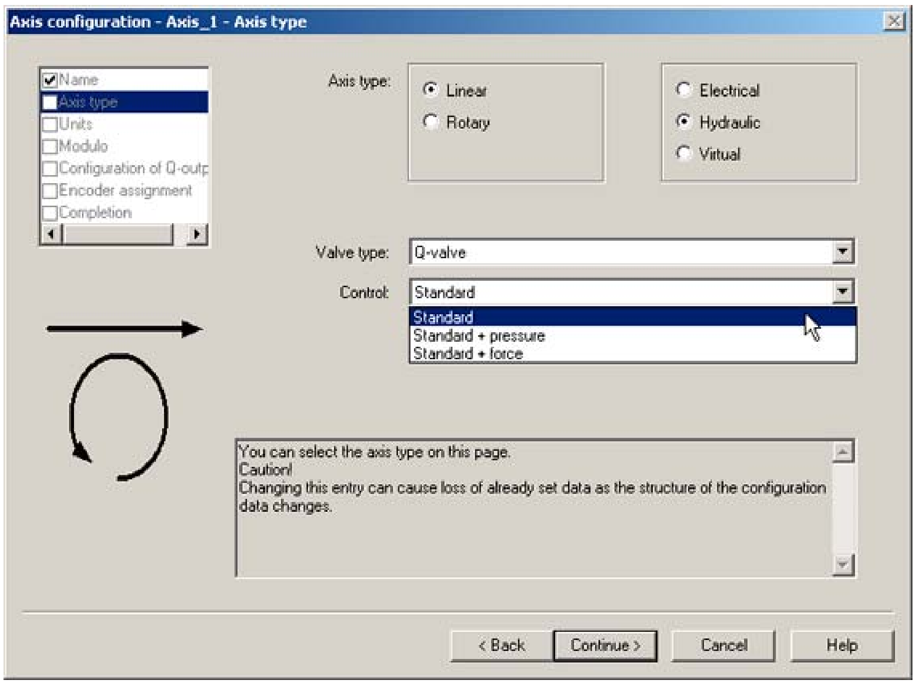
Application examples for hydraulic axes can be found on the Utilities & Applications CD under "FAQs > Technology".

See also

Access to the same final controlling element from multiple axes (Page 256)

5.1.3 Setting as a real axis with Q valve only

With hydraulic axes with Q valve only, the functions for traversing the axis, for velocity limiting, for force/pressure control, and for force/pressure limiting are available as with the electric axis.



- [✓ Name]
- Axis type
- Units
- Modulo
- Configuration of Q-outp
- Encoder assignment
- Completion

Axis type: [✓ Linear] Rotary

Electrical [✓ Hydraulic] Virtual

Valve type: Q-valve

Control: Standard

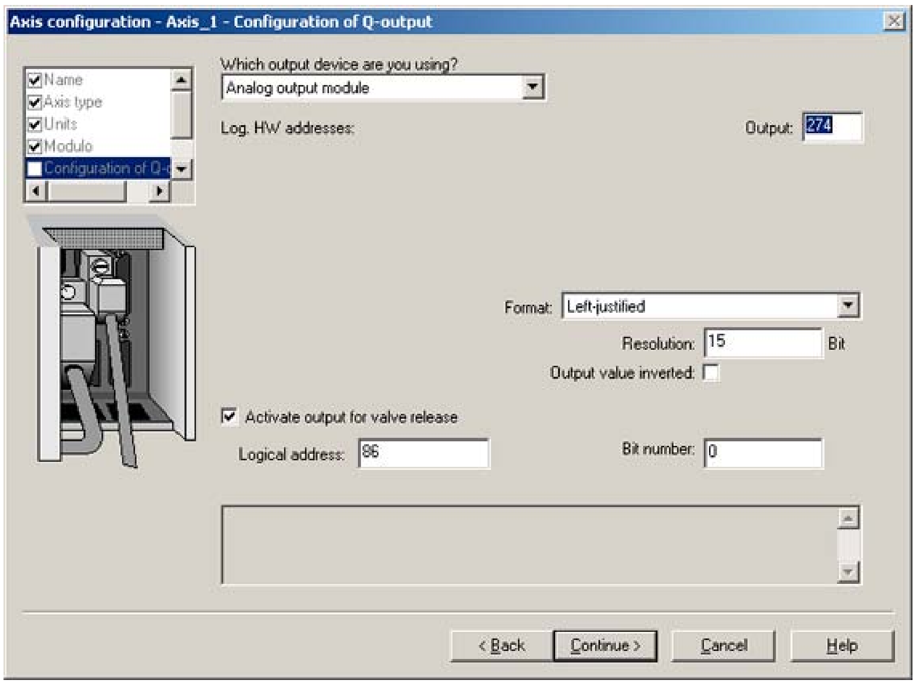
Standard
Standard + pressure
Standard + force

You can select the axis type on this page.
Caution!
Changing this entry can cause loss of already set data as the structure of the configuration data changes.

Figure 5-2 Setting the axis type

Table 5- 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


Axis configuration - Axis_1 - Configuration of Q-output

Which output device are you using? Analog output module

Log. HW addresses: Output: 274

Format: Left-justified

Resolution: 15 Bit

Output value inverted: ☐

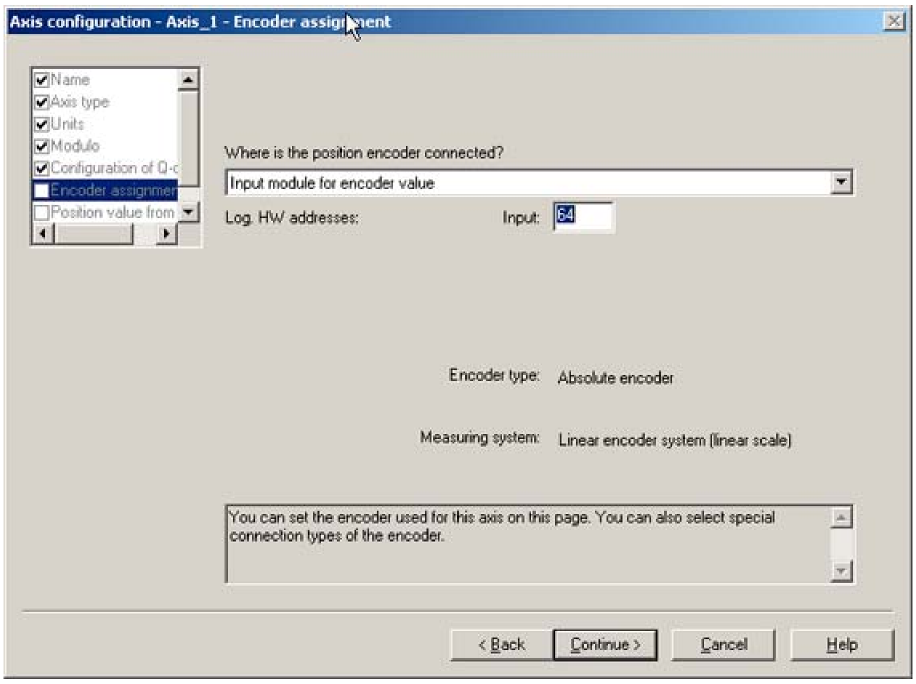
☑ Activate output for valve release

Logical address: 86

Bit number: 0

Figure 5-3 Example of configuration for the Q output

The bit resolution of the analog output module is set under Resolution.


Axis configuration - Axis_1 - Encoder assignment

Where is the position encoder connected? Input module for encoder value

Log. HW addresses: Input: 64

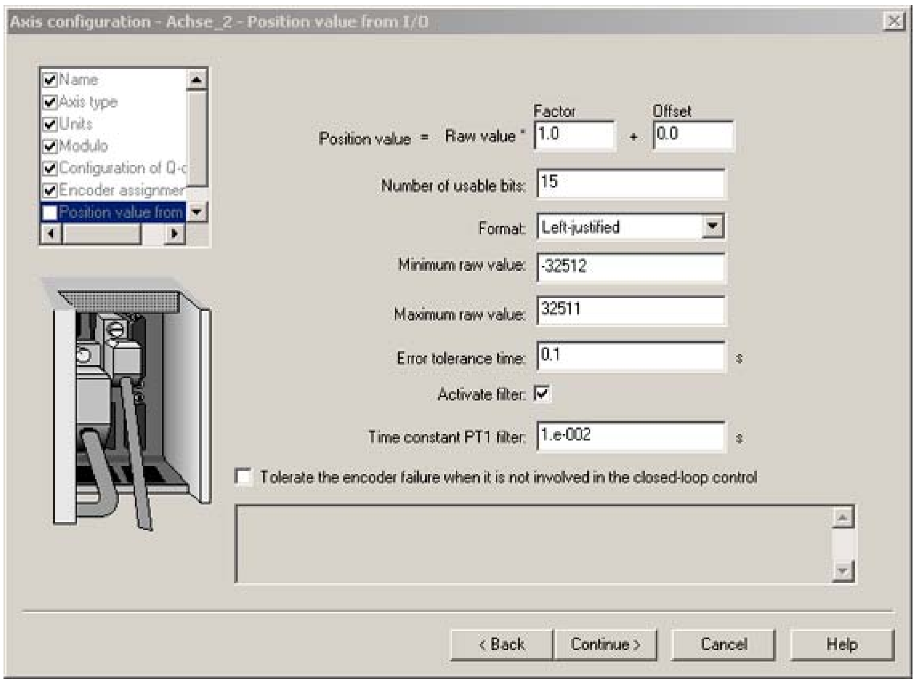
Encoder type: Absolute encoder

Measuring system: Linear encoder system (linear scale)

You can set the encoder used for this axis on this page. You can also select special connection types of the encoder.

Figure 5-4 Example of configuration for the encoder assignment

The logical hardware address of the input module can be found in the HW Config in SCOUT.



Position value = Raw value * Factor 1.0 + Offset 0.0

Number of usable bits: 15

Format: Left-justified

Minimum raw value: -32512

Maximum raw value: 32511

Error tolerance time: 0.1 s

Activate filter: ✓

Time constant PT1 filter: 1.e-002 s

☐ Tolerate the encoder failure when it is not involved in the closed-loop control

Figure 5-5 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 27)

TypeOfAxis configuration data (Page 29)

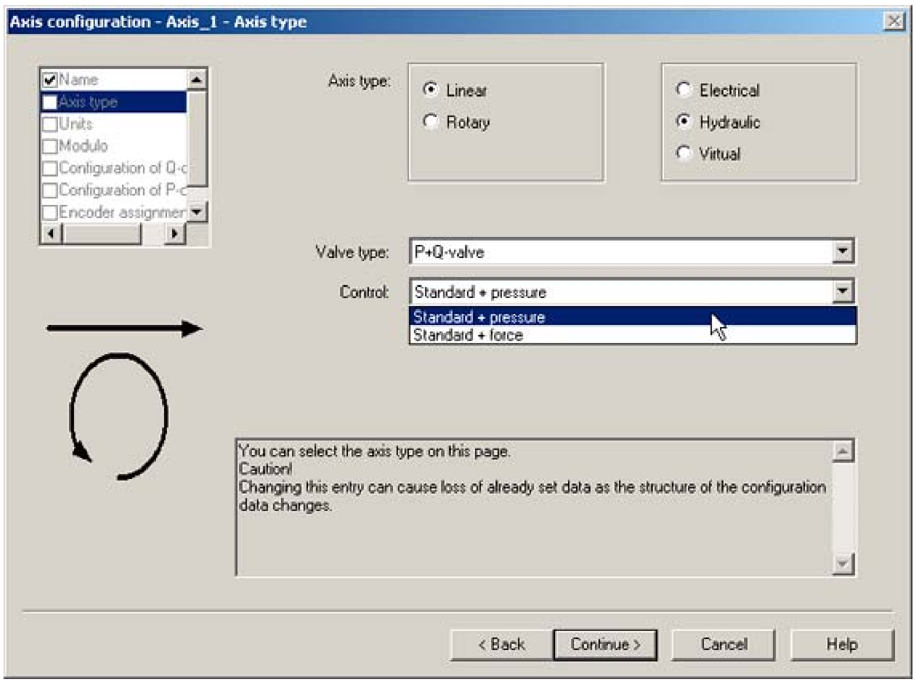
5.1.4 Setting as a real axis with Q valve + P valve/F output

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).



Axis type: [✓ Linear] Rotary — Electrical [✓ Hydraulic] Virtual

Valve type: P+Q-valve

Control: Standard + pressure

Standard + pressure
Standard + force

You can select the axis type on this page.
Caution!
Changing this entry can cause loss of already set data as the structure of the configuration data changes.

Figure 5-6 Setting the axis type

Table 5- 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 27)

TypeOfAxis configuration data (Page 29)

5.1.5 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.

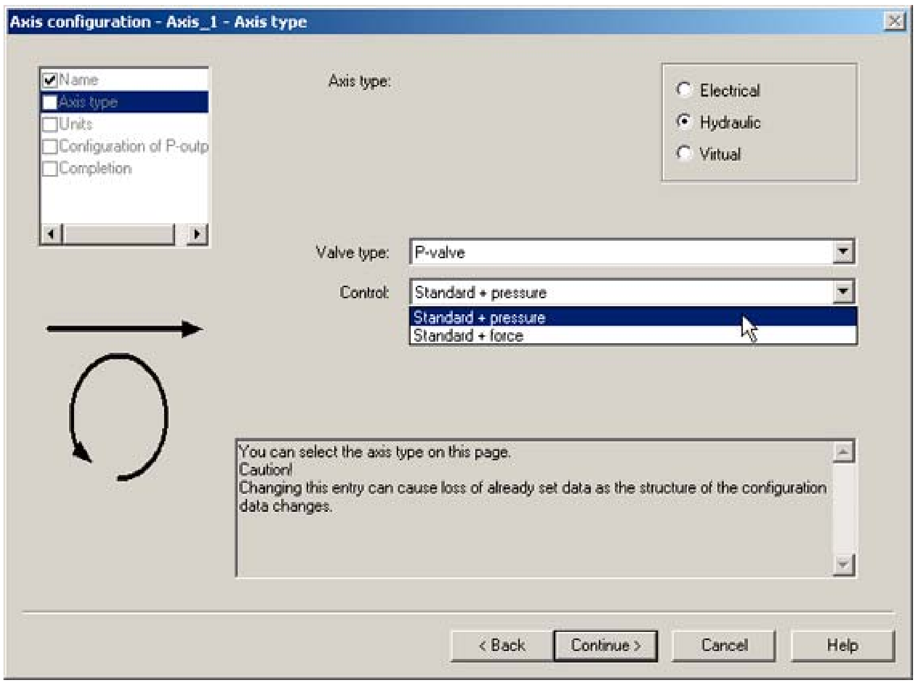
The following commands are possible on this axis:

- _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.



Axis type: ( ) Electrical [✓ Hydraulic] ( ) Virtual

Valve type: P-valve

Control: Standard + pressure

Standard + pressure
Standard + force

You can select the axis type on this page.
Caution!
Changing this entry can cause loss of already set data as the structure of the configuration data changes.

Figure 5-7 Creating an axis with P valve only

Table 5- 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 19)

Setting for the hydraulic axis type (Page 27)

TypeOfAxis configuration data (Page 29)

5.1.6 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 45).

5.2 Input limits, technological limiting functions

Information about the system input limits and technological limitations is contained in Input limits, technological limiting functions (Page 63).

5.3 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 63)

5.4 Defaults

Information about default settings for system variables can be found in Defaults (Page 68).

5.5 Homing

5.5.1 Overview of homing

A short summary for homing, absolute encoder and incremental encoder is contained in Overview of homing (Page 69).

5.5.2 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)

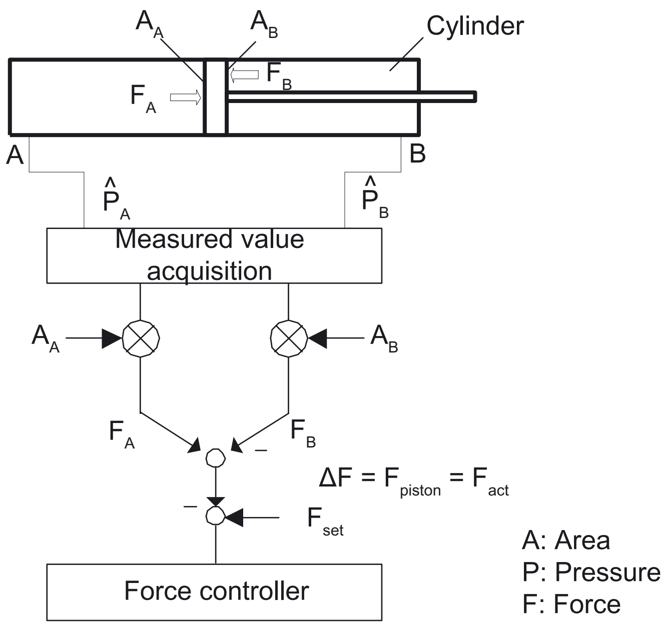


Figure 5-8 Example of a differential pressure measurement

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 differential pressure measurement must be configured on the same technology object.

The differential pressure measurement may also be applied to the electric axis.

See also

Using the expert list for an axis (Page 217)

5.5.3 Differential position measurement (V3.2 and higher)

Information on differential position measurement is contained in Differential position measurement (V3.2 and higher) (Page 85)

5.6 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 86)

5.7 Motion profiles

The motion profiles that can be applied are the same as for the electrical axis.

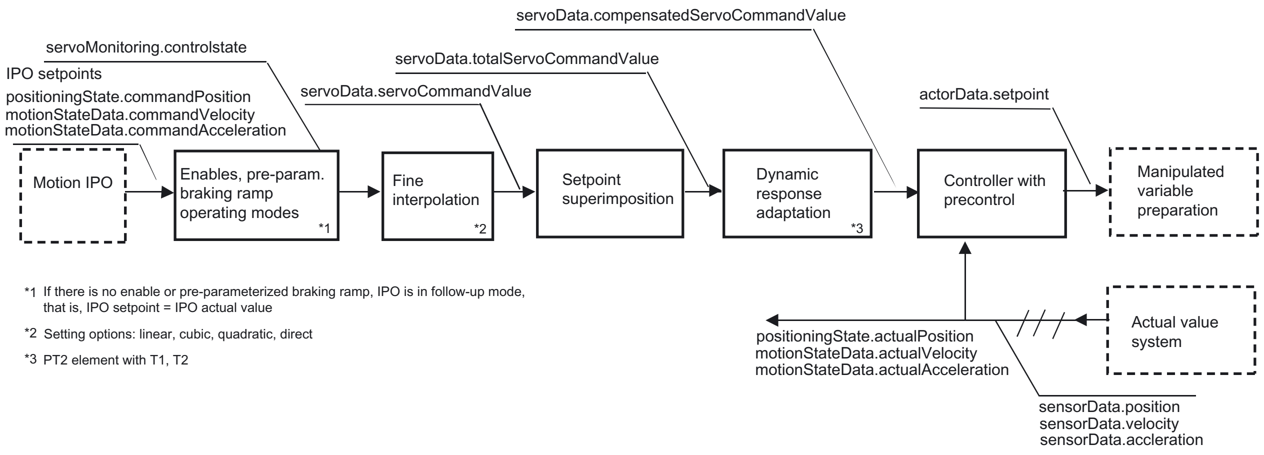
See also

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

5.8 Hydraulic axis with position control/velocity control

5.8.1 Position control for setting a positioning axis with hydraulic functionality

Block diagram of the hydraulic axis with closed-loop control:

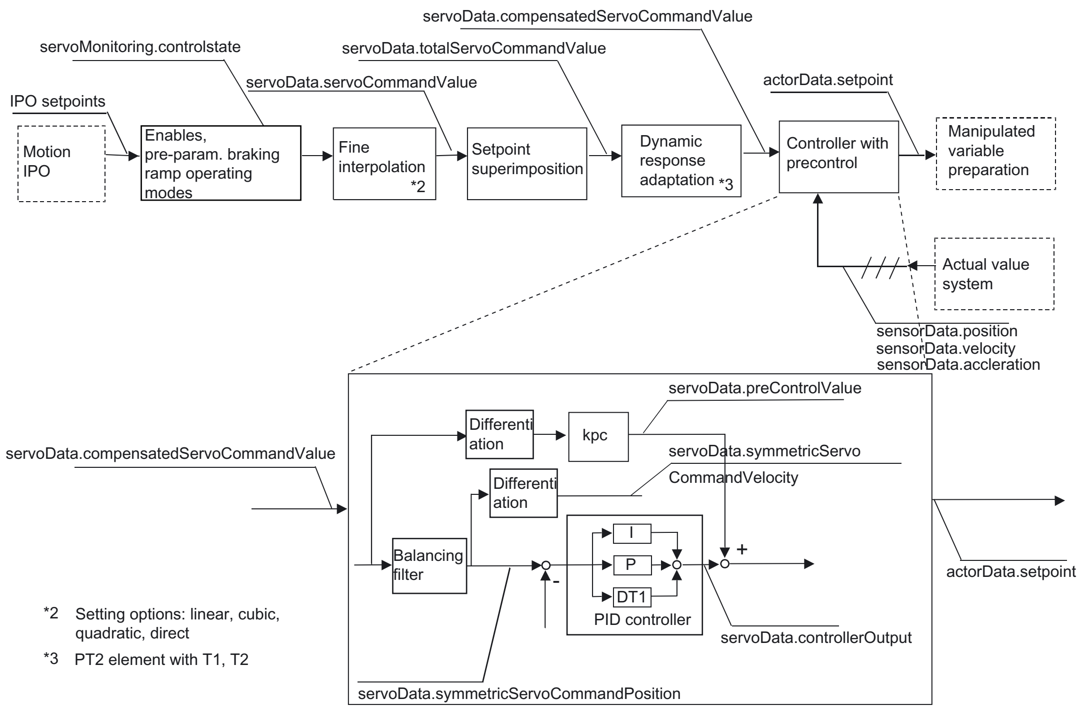

servoData.compensatedServoCommandValue
servoMonitoring.controlstate
servoData.totalServoCommandValue
IPO setpoints
positioningState.commandPosition
motionStateData.commandVelocity
motionStateData.commandAcceleration
servoData.servoCommandValue
actorData.setpoint
Motion IPO
Enables, pre-param. braking ramp operating modes *1
Fine interpolation *2
Setpoint superimposition
Dynamic response adaptation *3
Controller with precontrol
Manipulated variable preparation
Actual value system
positioningState.actualPosition
motionStateData.actualVelocity
motionStateData.actualAcceleration
sensorData.position
sensorData.velocity
sensorData.accleration

*1 If there is no enable or pre-parameterized braking ramp, IPO is in follow-up mode, that is, IPO setpoint = IPO actual value

*2 Setting options: linear, cubic, quadratic, direct

*3 PT2 element with T1, T2

Figure 5-9 Overview of hydraulic axis with closed-loop control


servoData.compensatedServoCommandValue
servoMonitoring.controlstate
servoData.totalServoCommandValue
servoData.servoCommandValue
actorData.setpoint
IPO setpoints
Motion IPO
Enables, pre-param. braking ramp operating modes
Fine interpolation *2
Setpoint superimposition
Dynamic response adaptation *3
Controller with precontrol
Manipulated variable preparation
Actual value system
sensorData.position
sensorData.velocity
sensorData.accleration
servoData.preControlValue
Differentiation
kpc
servoData.symmetricServoCommandVelocity
servoData.compensatedServoCommandValue
Differentiation
I
P
DT1
PID controller
Balancing filter
+
-
servoData.controllerOutput
actorData.setpoint
servoData.symmetricServoCommandPosition

*2 Setting options: linear, cubic, quadratic, direct

*3 PT2 element with T1, T2

Figure 5-10 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 positioning 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 96)

Position control (Page 97)

5.8.2 Velocity controller when setting speed-controlled axis with hydraulic functionality

When an axis is set as a speed-controlled axis with hydraulic functionality, the velocity controller is calculated in SIMOTION.

With the electric speed-controlled axis, the velocity controller is included in the drive and the controller specifies the speed setpoint.

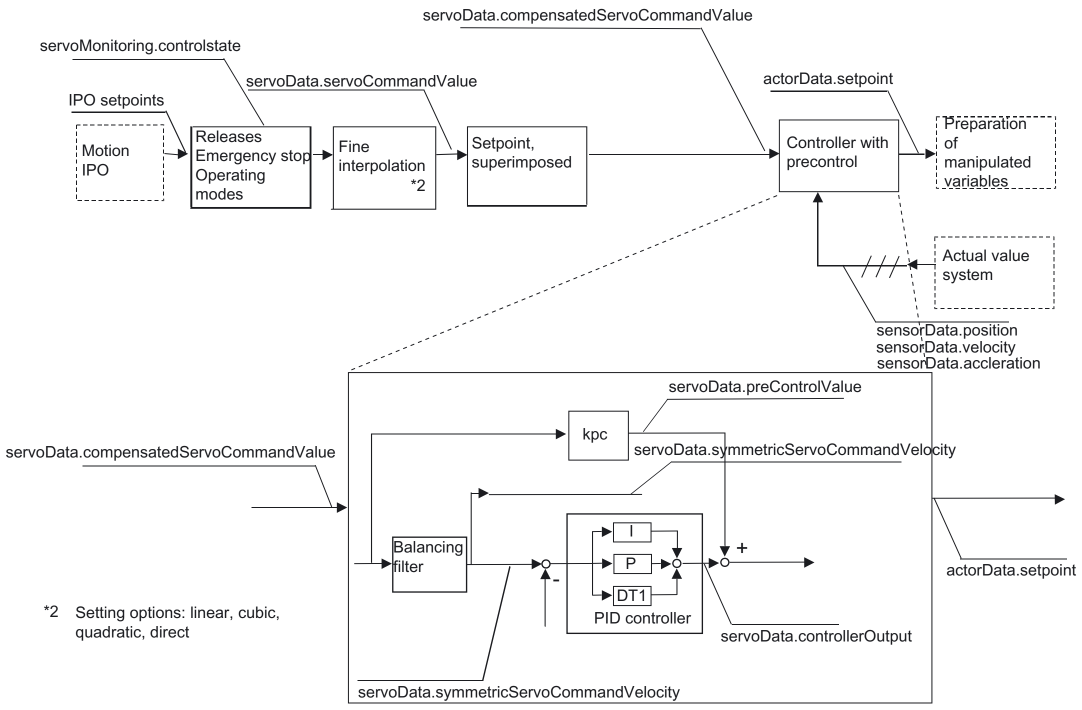

servoData.compensatedServoCommandValue
servoMonitoring.controlstate
servoData.servoCommandValue
actorData.setpoint
IPO setpoints
Motion IPO
Releases Emergency stop Operating modes
Fine interpolation *2
Setpoint, superimposed
Controller with precontrol
Preparation of manipulated variables
Actual value system
sensorData.position
sensorData.velocity
sensorData.accleration
servoData.preControlValue
kpc
servoData.compensatedServoCommandValue
servoData.symmetricServoCommandVelocity
Balancing filter
I
P
DT1
PID controller
+
-
servoData.controllerOutput
actorData.setpoint
servoData.symmetricServoCommandVelocity
*2 Setting options: linear, cubic, quadratic, direct

Figure 5-11 Block diagram of the velocity controller

Setting an axis as a speed-controlled 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 speed-controlled 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 96)

5.8.3 Preparation of manipulated variables for axis with hydraulic functionality

Preparation of manipulated variables for axis with hydraulic functionality, Q output

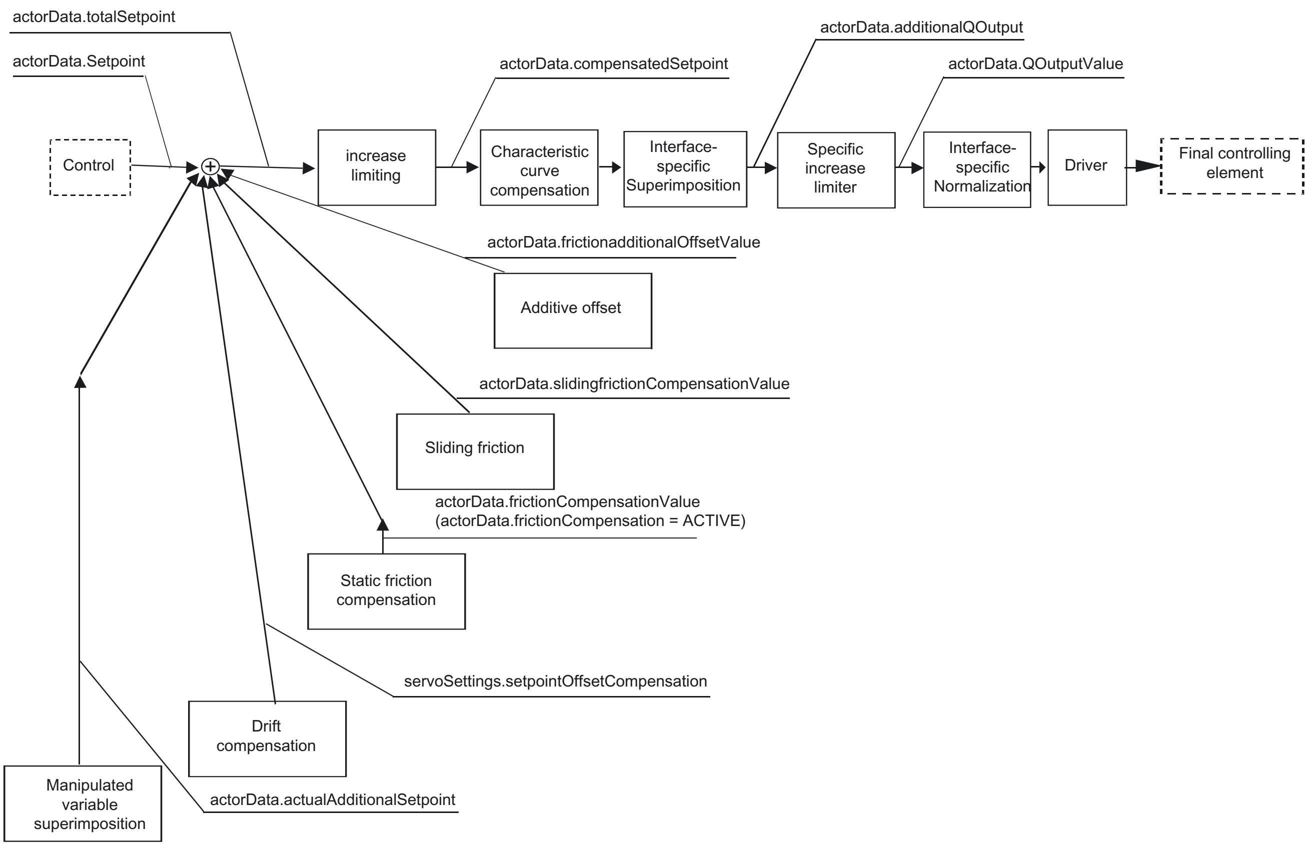

actorData.totalSetpoint
actorData.Setpoint
actorData.additionalQOutput
actorData.compensatedSetpoint
actorData.QOutputValue
Control
increase limiting
Characteristic curve compensation
Interface-specific Superimposition
Specific increase limiter
Interface-specific Normalization
Driver
Final controlling element
actorData.frictionadditionalOffsetValue
Additive offset
actorData.slidingfrictionCompensationValue
Sliding friction
actorData.frictionCompensationValue
(actorData.frictionCompensation = ACTIVE)
Static friction compensation
servoSettings.setpointOffsetCompensation
Drift compensation
Manipulated variable superimposition
actorData.actualAdditionalSetpoint

Figure 5-12 Preparation of manipulated variables for axis with hydraulic functionality, Q output

Preparation of manipulated variables for axis with hydraulic functionality, F output

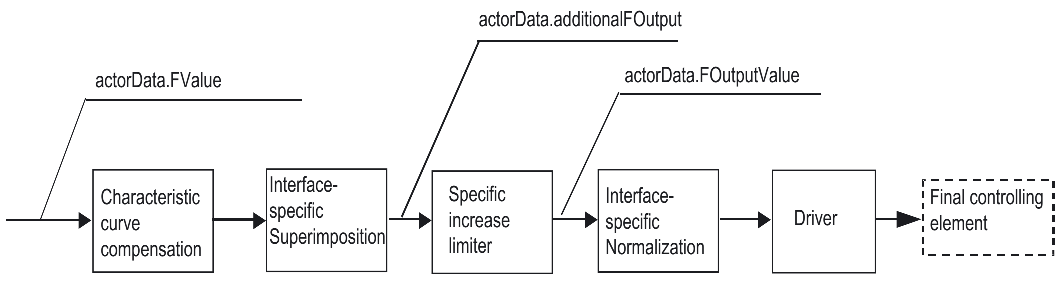

actorData.additionalFOutput
actorData.FValue
actorData.FOutputValue
Characteristic curve compensation
Interface-specific Superimposition
Specific increase limiter
Interface-specific Normalization
Driver
Final controlling element

Figure 5-13 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 96)

5.8.4 Manipulated variable filtering (as of V4.1 SP1)

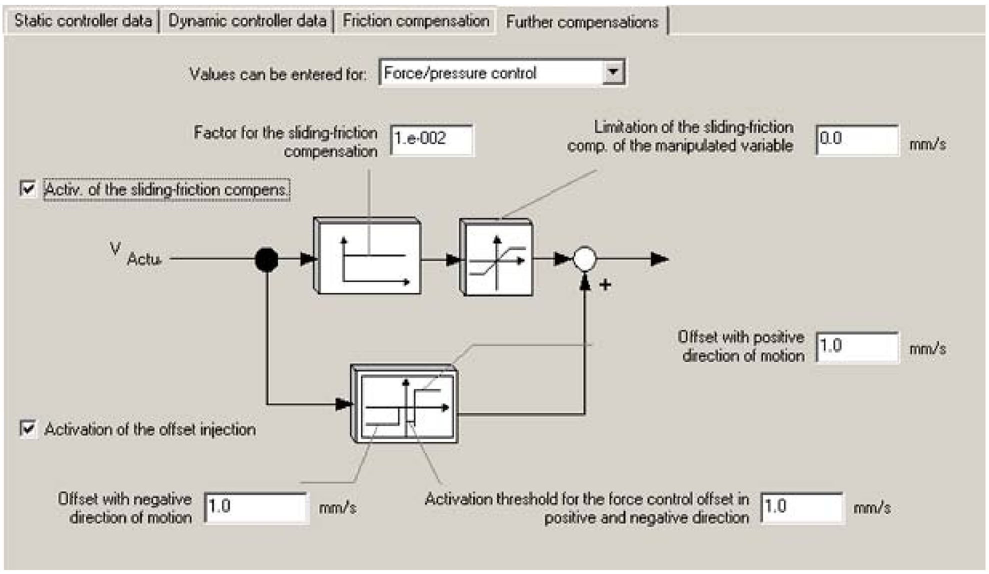
A manipulated variable can be set as a PT1 filter in the setpointFilter configuration data element. This filter acts after the controller and after the precontrol value and the additive manipulated variable value (additionalSetpoint) have been added.

A change in the filter data takes effect immediately.

5.8.5 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.



Values can be entered for: Force/pressure control

Factor for the sliding-friction compensation: 1.e-002

Limitation of the sliding-friction comp. of the manipulated variable: 0.0 mm/s

☑ Activ. of the sliding-friction compens.

Offset with positive direction of motion: 1.0 mm/s

☑ Activation of the offset injection

Offset with negative direction of motion: 1.0 mm/s

Activation threshold for the force control offset in positive and negative direction: 1.0 mm/s

Figure 5-14 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 96)

5.8.6 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.

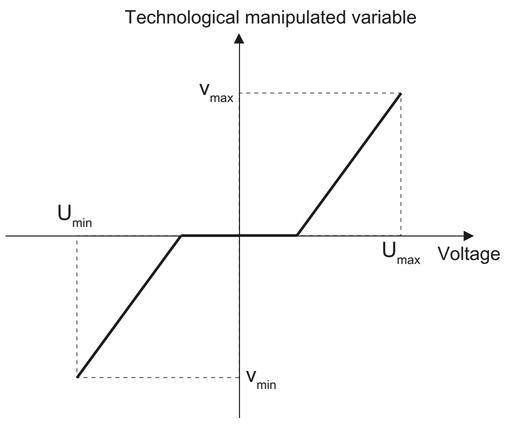


Figure 5-15 Characteristic curve setting for hydraulic axis

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 in the permissible 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

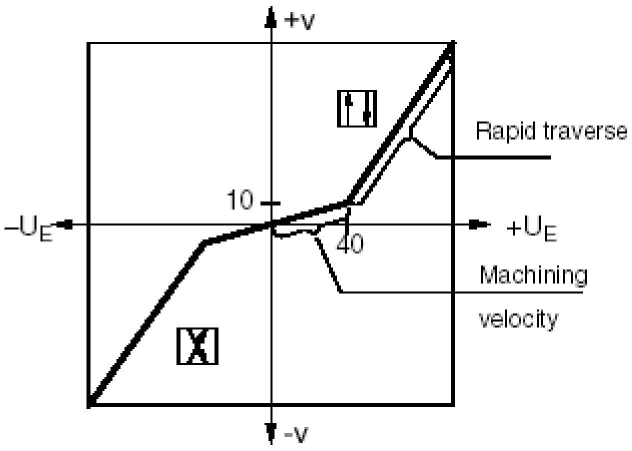
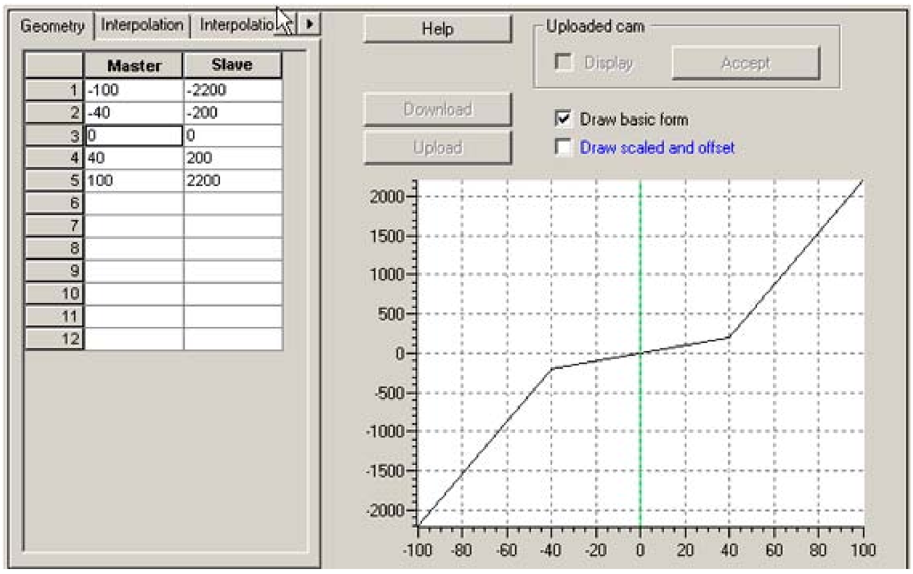


Figure 5-16 Valve characteristic curve from the manufacturer's catalog



	Master	Slave
1	-100	-2200
2	-40	-200
3	0	0
4	40	200
5	100	2200
6		
7		
8		
9		
10		
11		
12		

[✓] Draw basic form
[ ] Draw scaled and offset

Figure 5-17 Assigning characteristic curve parameters in SCOUT

At 100% manipulated variable, a velocity of 2200 mm/s is reached in the example (positioning axis).

See also

Overview of positioning axis with position control (Page 96)

5.8.7 Access to the same final controlling element from multiple axes

The axis with hydraulic functionality has its own activation/deactivation command **_enableQFAxis()/_disableQFAxis()**.

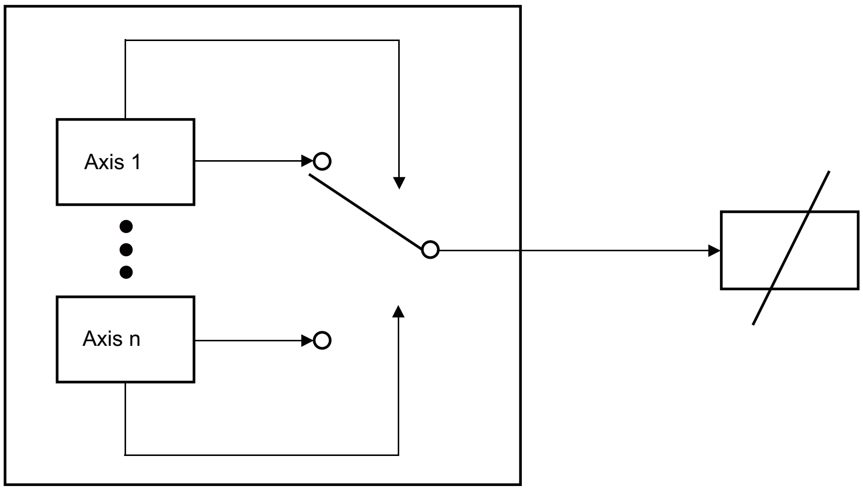


Figure 5-18 Axis with hydraulic functionality: one valve for several axes

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 96)

5.9 Travel to fixed endstop

The travel to fixed endstop functionality is not available for hydraulic axes.

See also

Travel to fixed endstop (Page 156)

5.10 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 165)

5.11 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 174)

5.12 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.

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.

5.13 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.

5.14 Force/pressure limiting with hydraulic speed-controlled axes with Q valve only (V4.0 and higher)

Force/pressure limiting on the hydraulic speed-controlled axis is available (V4.0 and higher).

With the hydraulic speed-controlled axis, the pressure limiting component is added to the velocity setpoint.

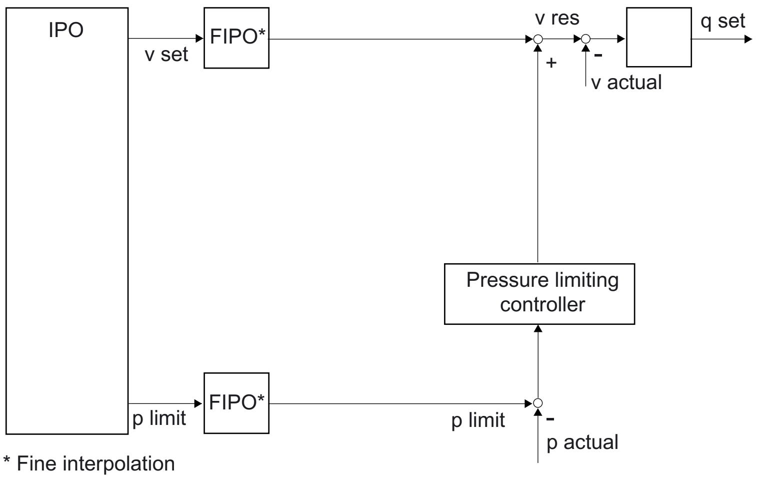


Figure 5-19 Overview of control structure for force/pressure limiting on hydraulic speed-controlled axis

---

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 speed-controlled 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.

5.15 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 6

6.1 Overview of commands

Note

You can also find information about the system functions in the *SIMOTION Cam Technology Package list manual*.

6.1.1 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.

6.1.2 Commands for specification and control of motion on the axis

Table 6- 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 switchover 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()	Readout of 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**.				

Command	Meaning	D	P	G
Axis motions				
_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

Command	Meaning	D	P	G
_getMotionStateOfAxisCommand()	Reading current phase of motion	X	X	X
_bufferAxisCommandId()	Save CommandId	X	X	X
_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 6- 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
_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
_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 6- 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 6- 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.

6.1.3 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.

6.2 Enables, stop and continue commands, resets

6.2.1 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 be signalled 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.

As of V4.1 SP1, the **_enableQFAxis()** command allows you to activate the hydraulic positioning axis in non-position-controlled mode using parameter **movingMode**:= SPEED-CONTROLLED. When activating in speed mode, command parameter **servoControlMode:=**ACTIVE must be set so that the setpoint path is activated in the servo.

The closed-loop velocity-controlled hydraulic axis cannot be moved with open-loop velocity control via movingMode:=SPEED_CONTROLLED.

Note

If, for example, **_disableAxis()** is called immediately after **_enableAxis()**, the commands can displace one another from the command buffer.

Note

- With **_enableAxis()**, the position controller in SIMOTION is enabled immediately, provided nothing different is set in the command parameters.
- 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.

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 controller 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 limiting is available.

System variables

Table 6- 5 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)
.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.)
.velocityControllerServo Monitoring.controlstate	Active / Inactive	Velocity controller on the hydraulic speed-controlled 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 STW1 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 ZSW 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 Setting as a real axis with digital drive coupling (Page 36).

For information on defining the reaction to technological alarms, refer also to Section Adjustable response to RELEASE_DISABLE (Page 312).

Note

ZSW1 is initialized by the drive and the **actorMonitoring.driveState** and **actorMonitoring.power** system variables are generated in ZSW1 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 the 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 189)

Enabling force/pressure control depending on switchover conditions (Page 274)

Setting as a real axis with digital drive coupling (Page 36)

6.2.2 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.

6.2.3 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 endstop) and force/pressure limiting are retained.

Active and pending motion commands on the axis are aborted and cannot be resumed.

Note

Exception

The **stopEmergency()** command with **stopDriveMode** = STOP_WITH_COMMAND_VALUE_ZERO deactivates the torque reduction and the Travel to fixed endstop command is canceled

---

Status

The **stopEmergencyCommand.state:= ACTIVE** status is set. Motion commands on the axis are not active in this status.

Table 6- 6 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

If necessary, the **stopEmergencyCommand**:=ACTIVE status lasts longer than the **_disableAxis()** command is active so that the **stopEmergencyCommand**:=INACTIVE status can be explicitly checked before the enables are set once more.

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
- Stop with maximum dynamic values on the axis (STOP_WITH_MAXIMAL_DECELERATION)
- Stop 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 (V3.2 and higher)

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 142)

6.2.4 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.

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.

6.2.5 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 speed-controlled 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.

6.2.6 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.

6.2.7 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 6- 7 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 301)

6.2.8 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.

System variables

Table 6- 8 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

6.2.9 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, provided 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.

6.3 Commands for axis motions

6.3.1 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 controller 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 controller is switched off.

    Note

    Once a new project has been downloaded to the controller, 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 6- 9 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

Overview of homing (Page 69)

6.3.2 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.

Table 6- 10 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 142)

6.3.3 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.

The following table provides an overview of the possible combinations of direction specification, axis type, and positioning mode.

Table 6- 11 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 6- 12 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 143)

Positioning with blending (Page 144)

6.3.4 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.

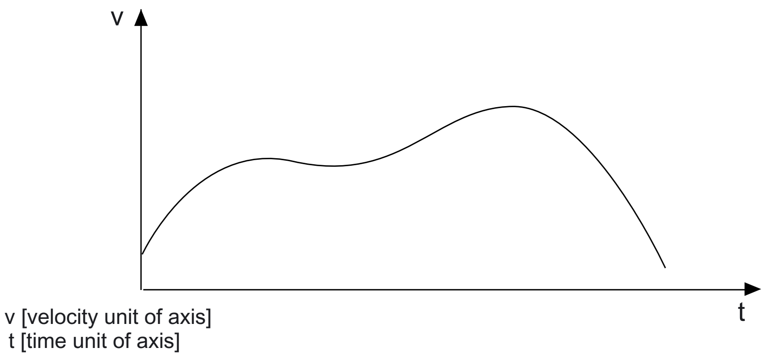


Figure 6-1 Example of a time-related velocity profile

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 186)

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

6.3.5 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.

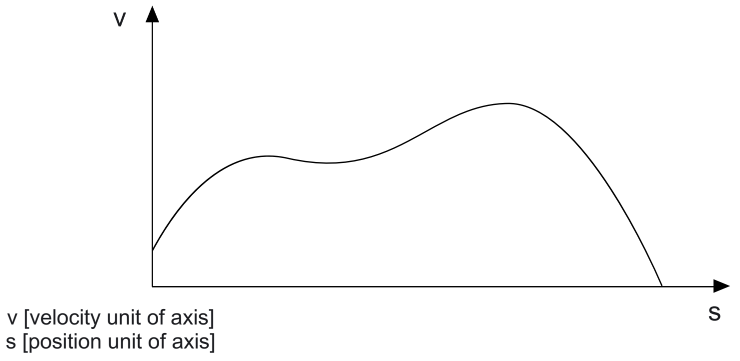


Figure 6-2 Example of a position-related velocity profile

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 186)

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

6.3.6 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.

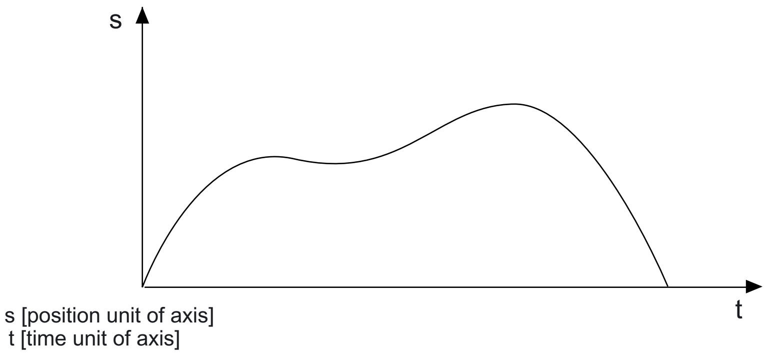


Figure 6-3 Example of a time-related position profile

The status and values are displayed in the elements of the positionTimeProfileCommand system variable.

See also

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

Behavior at the end of the profile (V3.2 and higher) (Page 186)

6.3.7 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 6- 13 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 150)

Technology data (Page 159)

6.3.8 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.

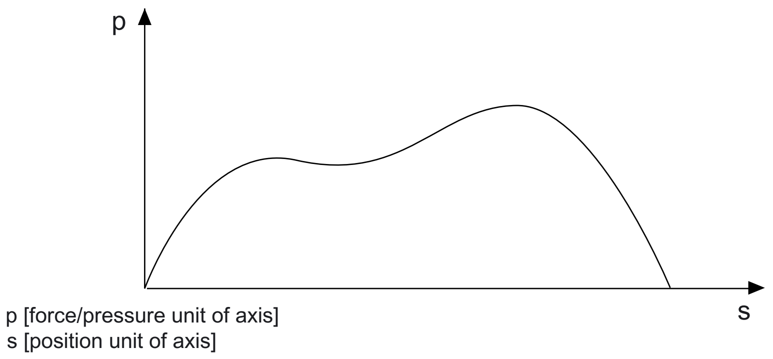


Figure 6-4 Example of a position-related force/pressure limiting profile

The status and values are displayed in the elements of the **forceLimitingCommand** system variable.

See also

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

6.3.9 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

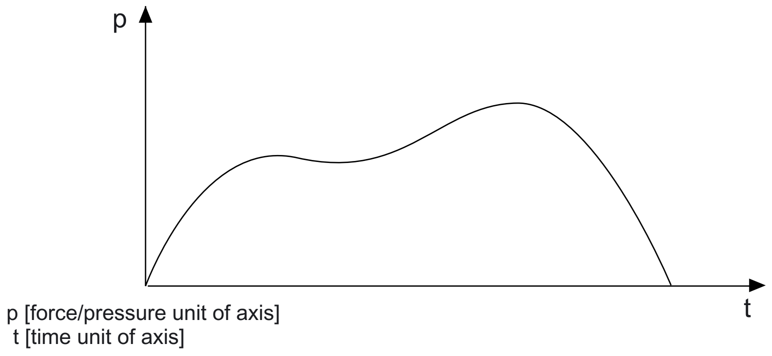


Figure 6-5 Example of a time-related limiting profile

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

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

6.3.10 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.

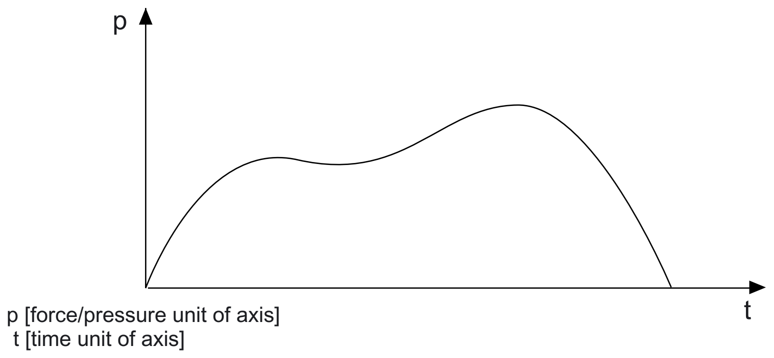


Figure 6-6 Example of a time-related force/pressure profile

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

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

6.3.11 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.

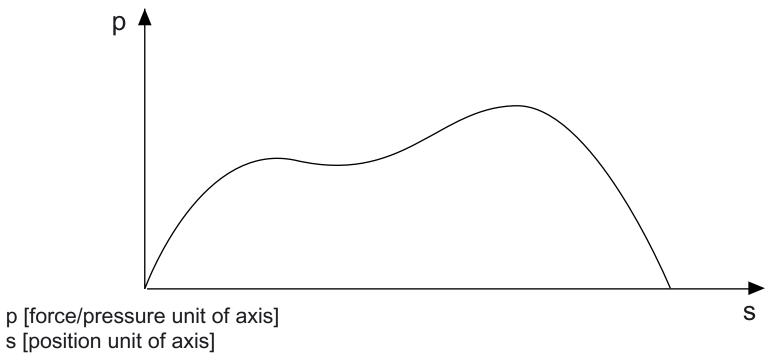


Figure 6-7 Example of a position-related force/pressure profile

The status and values are displayed in the elements of the **forcePositionProfileCommand** system variable.

See also

Overview of traversing with user-defined motion and force/pressure profiles (Page 181)

6.3.12 Enabling/disabling force/pressure limiting

The **_enableForceLimitingValue()** command enables force/pressure limiting on the axis. 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. 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 6- 14 System variables for force/pressure limiting

Variable	State	Meaning
forceLimitingCommand.state	ACTIVE/INACTIVE	Activation status of a command for force limiting on the axis

See also

Overview of force/pressure limiting (Page 174)

6.3.13 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 6- 15 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 173)

6.3.14 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 173)

6.3.15 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 173)

6.3.16 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 156)

6.4 Commands for defining the coordinate system

6.4.1 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.

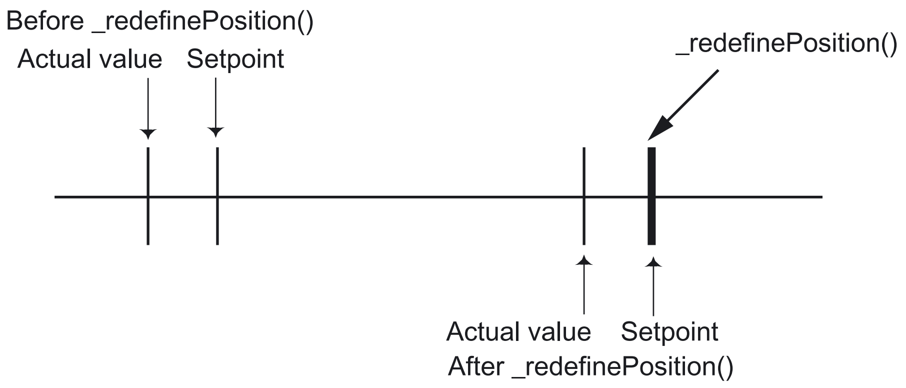


Figure 6-8 Absolute offset of axis position relative to setpoint

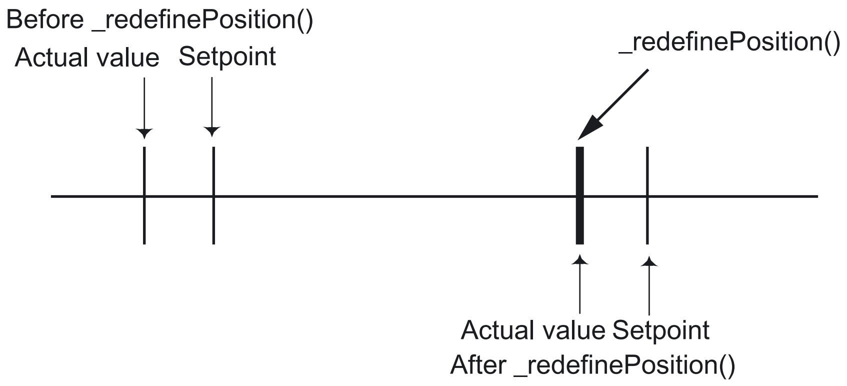


Figure 6-9 Absolute offset of axis position relative to actual value

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 148)

6.5 Simulation commands

6.5.1 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.

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 **_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).

The **_resetAxis()** command switches the axis back from 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 6- 16 System variables for program simulation

Variable	Status	Meaning
Simulation	ACTIVE/INACTIVE	Simulation status

See also

Setting as a real axis without drive (axis simulation) (Page 45)

6.6 Information functions / command buffers

6.6.1 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.

6.6.2 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 301)

6.6.3 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 301)
- 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 301)
- ABORTED
  Execution of the command has been aborted

System variables

None

See also

Saving the Command ID (Page 301)

6.6.4 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.numberOfBufferedComandId** configuration data element. This must be managed by the user program. If the buffer overflows, the command will be rejected with a return value.

6.6.5 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.

6.6.6 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.

6.6.7 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.

6.6.8 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.

6.6.9 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.

6.6.10 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 6- 17 System variables for activating data sets

Variable	Status/Type	Meaning
dataSetMonitoring	See the StructDataSetMonitoring data type as defined in the reference list of the system variables	Information for data set changeover • Status • 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 movements of the position controller if differences in position are identified.

The data sets must have the same structure, i.e. if DS_1 has DSC, DS_2 must have DSC too.

See also

Data set overview (Page 178)

6.6.11 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 178)

6.6.12 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 178)

6.6.13 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
  Limitation of the manipulated variable of the force controller
  PID controller setting
  Controller type
  Effective direction of the manipulated variable
  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 read on the drive axis.

6.6.14 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 written on the speed-controlled axis.

6.6.15 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.

6.6.16 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

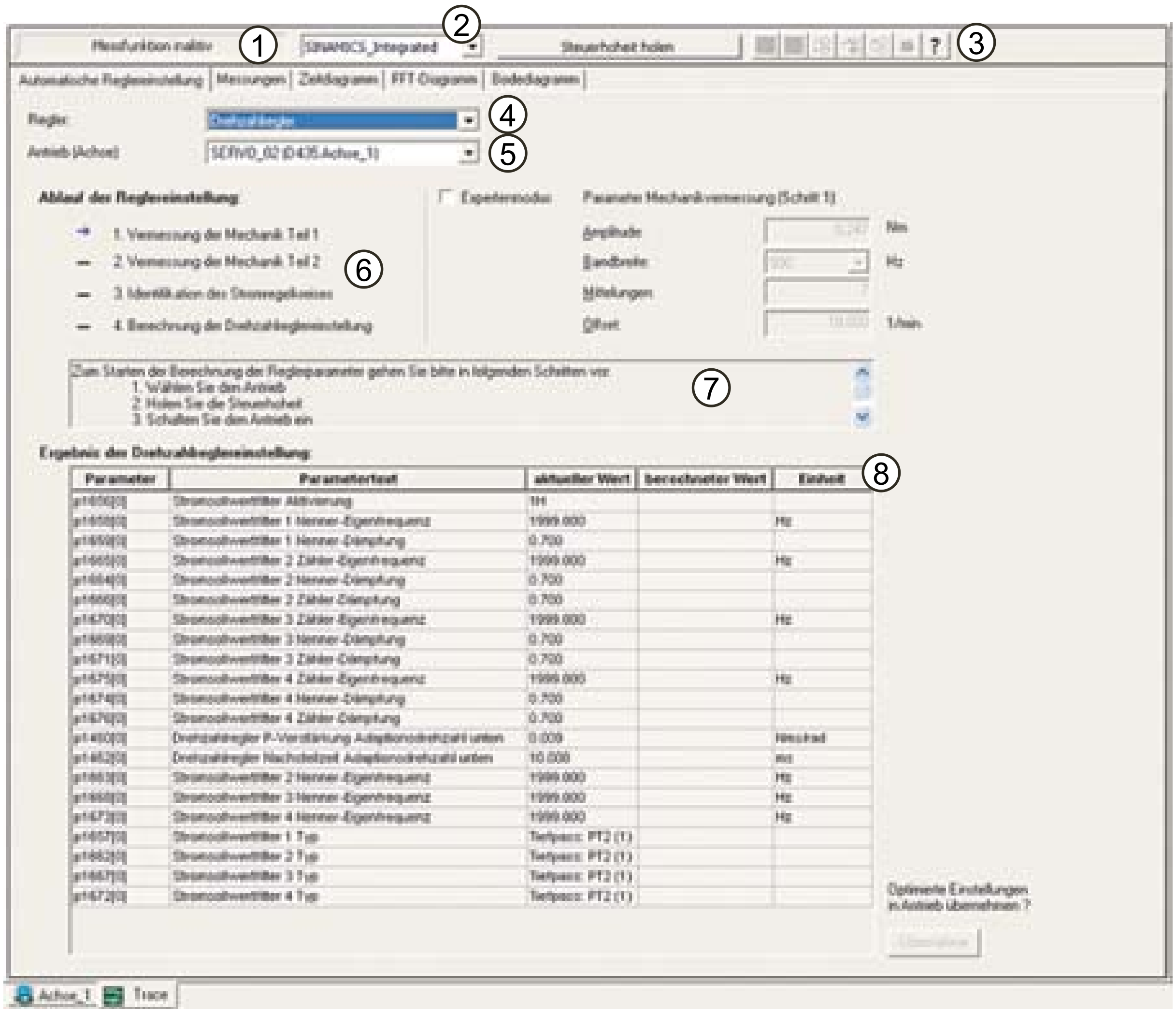
The specific data for the axis with hydraulic functionality in the data set cannot be read on the speed-controlled axis.

6.6.17 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.

6.7 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.



Regler: Drehzahlregler

Antrieb (Achse): SERVO_02 (D435.Achse_1)

Ablauf der Reglereinstellung

1. Vermessung der Mechanik Teil 1
2. Vermessung der Mechanik Teil 2
3. Identifikation des Stromregelkreises
4. Berechnung der Drehzahlreglereinstellung

☐ Expertenmodus

Parameter Mechanikvermessung (Schritt 1)

Amplitude	0.242	Nm
Bandbreite	500	Hz
Mittelungen	7	
Offset	10.000	1/min

Zum Starten der Berechnung der Reglerparameter gehen Sie bitte in folgenden Schritten vor:
1. Wählen Sie den Antrieb
2. Holen Sie die Steuerhoheit
3. Schalten Sie den Antrieb ein

Ergebnis der Drehzahlreglereinstellung

Parameter	Parametertext	aktueller Wert	berechneter Wert	Einheit
p1656[0]	Stromsollwertfilter Aktivierung	1H		
p1658[0]	Stromsollwertfilter 1 Nenner-Eigenfrequenz	1999.000		Hz
p1659[0]	Stromsollwertfilter 1 Nenner-Dämpfung	0.700		
p1665[0]	Stromsollwertfilter 2 Zähler-Eigenfrequenz	1999.000		Hz
p1664[0]	Stromsollwertfilter 2 Nenner-Dämpfung	0.700		
p1666[0]	Stromsollwertfilter 2 Zähler-Dämpfung	0.700		
p1670[0]	Stromsollwertfilter 3 Zähler-Eigenfrequenz	1999.000		Hz
p1669[0]	Stromsollwertfilter 3 Nenner-Dämpfung	0.700		
p1671[0]	Stromsollwertfilter 3 Zähler-Dämpfung	0.700		
p1675[0]	Stromsollwertfilter 4 Zähler-Eigenfrequenz	1999.000		Hz
p1674[0]	Stromsollwertfilter 4 Nenner-Dämpfung	0.700		
p1676[0]	Stromsollwertfilter 4 Zähler-Dämpfung	0.700		
p1460[0]	Drehzahlregler P-Verstärkung Adaptionsdrehzahl unten	0.009		Nms/rad
p1462[0]	Drehzahlregler Nachstellzeit Adaptionsdrehzahl unten	10.000		ms
p1663[0]	Stromsollwertfilter 2 Nenner-Eigenfrequenz	1999.000		Hz
p1668[0]	Stromsollwertfilter 3 Nenner-Eigenfrequenz	1999.000		Hz
p1673[0]	Stromsollwertfilter 4 Nenner-Eigenfrequenz	1999.000		Hz
p1657[0]	Stromsollwertfilter 1 Typ	Tiefpass: PT2 (1)		
p1662[0]	Stromsollwertfilter 2 Typ	Tiefpass: PT2 (1)		
p1667[0]	Stromsollwertfilter 3 Typ	Tiefpass: PT2 (1)		
p1672[0]	Stromsollwertfilter 4 Typ	Tiefpass: PT2 (1)		

Optimierte Einstellungen in Antrieb übernehmen ?

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 6-10 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 218)

6.8 Technological alarms

6.8.1 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 ramp values.
  - Active commands (IPO) are aborted and signaled back to the user program with an error code.
  - 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.
  - Active commands (IPO) are aborted and signaled back to the user program with an error code.
  - The position controller remains active.
  - The drive 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 drive and position controller are activated with _enableAxis(), the axis can travel again (provided that 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 Setting as a real axis with digital drive coupling (Page 36) and Adjustable response to RELEASE_DISABLE (Page 312).
  - 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 147)

6.8.2 Adjustable response to RELEASE_DISABLE

The drive response to the RELEASE_DISABLE alarm response can be set (with a digital drive link).

See Setting as a real axis with digital drive coupling (Page 36).

See also

Alarm reactions (Page 309)

6.8.3 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.

The failure of an encoder not involved in closed-loop control is indicated by alarm 20015.

Part IV External Encoder - Description 7

7.1 External encoder overview

The External Encoder technology object provides the functionality in the system for connecting an external encoder without an axis, e.g., a shaft encoder on a press. An external encoder behaves like a positioning axis from which only the actual position value is evaluated.

Information of 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.
- **Output Cam technology object**

  The External Encoder technology object is used to provide a reference value.
- **Cam Track technology object**

  The External Encoder technology object is used to provide a reference value.
- **Measuring Input technology object**

  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 8

8.1 Encoders that can be connected

A variety of incremental and absolute encoders can be connected, depending on the runtime system.

External encoders can be connected to the following devices:

- C2xx
- ADI4 (at least one axis must be created on the ADI4)
- IM174 (at least one axis must be created on the IM174)
- Encoder input to DP drives (second encoder)

  On the SIMODRIVE 611U, the second encoder interface on a Double Motor Module can be used to connect an external encoder for SIMOTION.

  For MASTERDRIVES, a second encoder can be connected by means of an encoder module (DSC is not possible).
- SMC encoder with electric axes

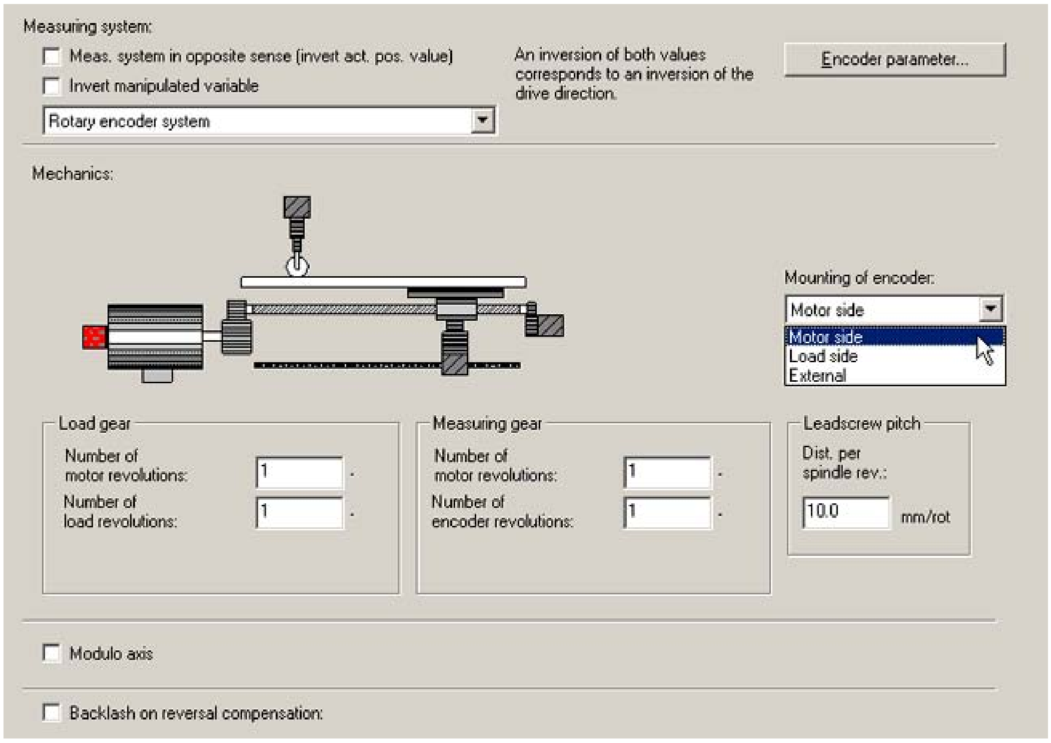
  With SINAMICS it is possible to configure multiple encoders, from which a maximum of two encoder values per drive can be transmitted in each axis message frame. The second encoder can be used freely - for example for axes with hydraulic functionality or for external encoders.
- PROFIBUS encoder (via message frame type 81 in accordance with PROFIdrive Profile V3)

Note

A free encoder in a PROFIdrive axis message frame can only be used for an External Encoder technology object in cyclic operation if an Axis technology object is created on the PROFIdrive message frame and is not deactivated. (Hint: For message frames with two encoders, each encoder is available for one External Encoder technology object if the axis is configured as a speed-controlled axis without encoder.)

For further information, refer to the relevant hardware manuals.

8.2 Mounting type


Measuring system:

- [ ] Meas. system in opposite sense (invert act. pos. value)
- [ ] Invert manipulated variable

Rotary encoder system

An inversion of both values corresponds to an inversion of the drive direction.

Mechanics:

Mounting of encoder: Motor side

Load gear

Number of motor revolutions:	1	-
Number of load revolutions:	1	-

Measuring gear

Number of motor revolutions:	1	-
Number of encoder revolutions:	1	-

Leadscrew pitch

Dist. per spindle rev.:	10.0	mm/rot

- [ ] Modulo axis
- [ ] Backlash on reversal compensation:

Figure 8-1 Definition of the mounting type of an external encoder in SIMOTION SCOUT

Methods for mounting an 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.

8.3 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 81)

8.4 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

8.5 Encoder assignment and terminology

The encoder type is specified with the encoder mode.

Table 8- 1 Definable encoder mode depending on the encoder type

	Encoder type		
Encoder mode	**Absolute encoder**	**Cyclic absolute encoder**	**Incremental encoder**
Endat (encoder data interface)	x	x	x
SSI (synchronous serial interface)	x	x	
Sinusoidal			x
Rectangle			x
Resolver		x[1)]	x
Analog encoder (value in the I/O area)	x		

1) Possible with single-pole-pair resolver only

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

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 message frame (for more information, see *Encoder interconnection via PROFIdrive message frame*).

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 8- 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 message frame (applies to SINAMICS, SIMODRIVE 611U and MASTERDRIVES)			
Incremental encoder	Rectangle	2,048	-
	Sinusoidal	2,048	-
	Resolver	2,048	-
	Endat	2,048	-
Absolute encoder	Endat	2,048	512
	SSI	1	1
...Cyclic absolute	Resolver (only pole pair number 1 possible)	2,048	512
	Endat	2,048	512
	SSI	1	1
PROFIBUS absolute encoder in PROFIdrive encoder message frame			
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).

Default setting:

- Fine resolution (**~Encoder.~ResolutionMultiplierCyclic**) = 0
- Fine resolution of the absolute encoder in Gn_XIST2 (**AbsEncoder.absResolutionMultiplierAbsolute**) =0

See also

Encoder list (Page 322)

8.6 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 on the Utilities & Applications CD under "FAQs > Drives > Parameters of the connectable encoders" and in the online help (index search via Encoder parameter assignment).

8.7 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)

See also

Encoder assignment and terminology (Page 319)

Encoder list (Page 322)

8.8 Encoder interface using the PROFIdrive message frame

The encoder values are transmitted in the PROFIdrive message frame (see Table *Message frame 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 message frame.

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.

See also the Encoder list.

Note

For SINAMICS drives, 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.

Encoder value via PROFIdrive axis message frames

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

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

Encoder value via PROFIdrive encoder message frame 8x

PROFIBUS/PROFINET encoders in accordance with the specification "Profile for DP-V2 Encoders Version 3.2" with message frame type 81 can be used. These encoders can be assigned freely. See also **PROFIBUS absolute encoder via PROFIdrive encoder message frame** in the next chapter.

Inconsistent configuration

If there are any differences between the configuration data in SIMOTION and the parameter settings for the encoder in the drive, the technological alarm

error 20005: Device type:2, log.address:1234 faulty. (Bit:0, reason: 0x80h)

is triggered as soon as an online connection is established between the controller and the drive/encoder.

For SINAMICS and SIMODRIVE, 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).

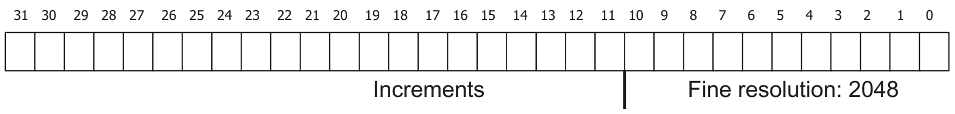


Figure 8-2 Example composition of the 32-bit encoder data of the cyclic actual value Gn_XIST1

Example of for an encoder with number of encoder pulses = 2,048 (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 2,048 (the default value depends on the encoder mode setting; see Table *Default settings for fine resolution in SIMOTION*).

SIMODRIVE 611U:

Table 8- 3 Settings

SIMOTION		611U	
Encoder pulses per revolution 1)	=2,048	P1007	=2,048
Fine resolution 2)	=0 (≡ 2,048)	P1042	=11

1) Inc/AbsEncoder.Inc/AbsResolution

2) Inc/AbsEncoder.Inc/AbsResolutionMultiplierCyclic

SINAMICS:

Table 8- 4 Settings

SIMOTION		SINAMICS	
Encoder pulses per revolution 1)	=2,048	P408	=2,048
Fine resolution 2)	=0 (≡ 2,048)	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 passed in Gn_XIST_2 (n = 1 or 2, number of encoder), they will be sent 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 = 2,048 (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 8- 5 Setting the encoder data

Encoder pulses per revolution 1)	2,048
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 (= 2,048)

1) AbsEncoder.AbsResolution
2) AbsEncoder.absDataLength
3) AbsEncoder.absResolutionMultiplierAbsolute
4) AbsEncoder.AbsResolutionMultiplierCyclic

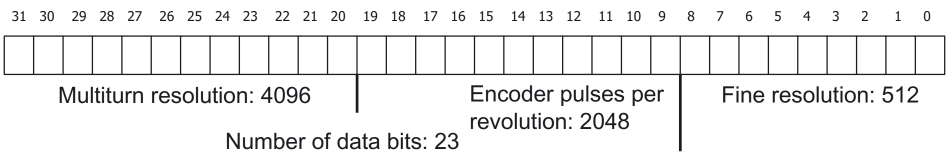


Figure 8-3 Example composition of the 32-bit encoder data of the absolute actual value Gn_XIST2

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 message frame

For SINAMICS and MASTERDRIVES, parameters are assigned for the number of pole pairs 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 = 2,048, 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 value encoder cyclical absolute, Endat, line count = 1, fine resolution = 2048, fine resolution absolute value = 2048, data width = 0)

See also the Encoder list.

PROFIBUS absolute encoder via PROFIdrive encoder message frame

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

See also the Encoder list.

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 4,096:
measuring steps per revolution = 4,096
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 in 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 message frame 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.

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*. You can find these documents on the *SIMOTION SCOUT Add-on* CD under *4_Additional_Documentation* in the document index.

See also

Encoder assignment and terminology (Page 319)

Encoder list (Page 322)

8.9 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
  - 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)

  Note that the measured value in accordance with the justification specification will be mapped to an internal 32-bit wide signed data value of the DINT type and the mapped value then tested with the maximum values (see maximum limits below) 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.

  For the VALUE_RIGHT_MARGIN_WITHOUT_SIGN setting, the maximum permitted encoder resolution or measured value width is 31 bits.

For the VALUE_LEFT_MARGIN setting, the measured value for the setting of the encoder resolution of measured value width is

– Mapped < 16 bits left-justified to an internal 16-bit wide data value, namely, to the least-significant byte 1 and byte 2 of the internal data value of the DINT type. The missing bits 15 minus the measured value width are right-padded with zeros and the most-significant byte 3 and byte 4 of the internal data value padded appropriately in accordance with the sign.
– Mapped ≥ 16 bits left-justified to the 32-bit wide data value and the missing 31 bits minus the measured value width right-padded with zeros.

For the VALUE_LEFT_MARGIN_WITHOUT_SIGN setting, the measured value for the setting of the encoder resolution of measured value width

– Mapped ≤ 16 bits left-justified to an internal 16-bit wide data value, namely, to the least-significant byte 1 and byte 2 of the internal data value of the DINT type. The missing 16 bits minus the measured value width are right-padded with zeros and the most-significant byte 3 and byte 4 filled with zeros.
– > Mapped > 16 bits left-justified to the 32-bit wide value and the missing 32 bits minus the measured value width right-padded with zeros.

  Because this mapped measured value is tested with the maximum limits of DINT data type, only the zero value is possible for the VALUE_LEFT_MARGIN_WITHOUT_SIGN setting and a measured value width > 16 bits as value of the most-significant bit in the measured value. This means the measuring range is limited to maximum 50% of the measured value width.

- 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.

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

The following bits can be configured for the direct value as absolute value in the I/O area (as of V4.1 SP1):

- **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 controller.

The controller supports the following options:

- The peripheral module displays the new measured value in an update bit/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)

8.10 Overflow bit for modulo counting

The number of modulo revolutions (V3.2 and higher) is described in the section **Actual value measurement / Actual value system**.

See also

Actual value measurement / actual value system (Page 110)

8.11 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 110)

8.12 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.extrapolationFilter.

The extrapolated actual values are displayed in extrapolationData.position and extrapolationData.velocity.

See also

Actual value measurement / actual value system (Page 110)

8.13 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.

8.14 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**.

8.15 Synchronization / Homing

8.15.1 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.

8.15.2 Synchronization/homing with incremental encoders

- **Direct homing / setting the home position**

  Direct homing is set using the synchronizingMode:=DIRECT_HOMING function parameter.

  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 using the synchronizingMode:=DIRECT_HOMING_RELATIVE function parameter.

  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/on-the-fly homing**

  Passive homing is set using the synchronizingMode:=PASSIVE_HOMING function parameter.

  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 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 79)

Direct homing/setting the home position (Page 80)

Relative direct homing/relative setting of home position (V3.2 and higher) (Page 80)

8.15.3 Synchronization/homing with absolute encoders

Direct homing/setting the home position

Direct homing is set using the synchronizingMode:=DIRECT_HOMING function parameter.

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 using the synchronizingMode:=DIRECT_HOMING_RELATIVE function parameter.
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 synchronizingMode:=ENABLE_OFFSET_OF_ABSOLUTE_ENCODER function parameter. The absolute encoder offset is set in the **absHomingEncoder.absshift** configuration data.

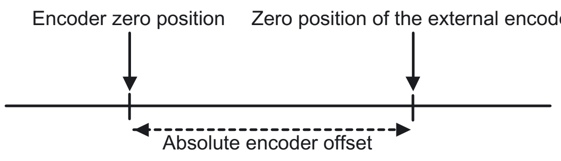


Figure 8-4 The external encoder value is the encoder zero position plus the absolute encoder offset.

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 NV RAM and remains in effect until the next absolute encoder adjustment. This function must therefore be executed once when the controller is commissioned.

The offset value and its calculation are set for the **configuration**.

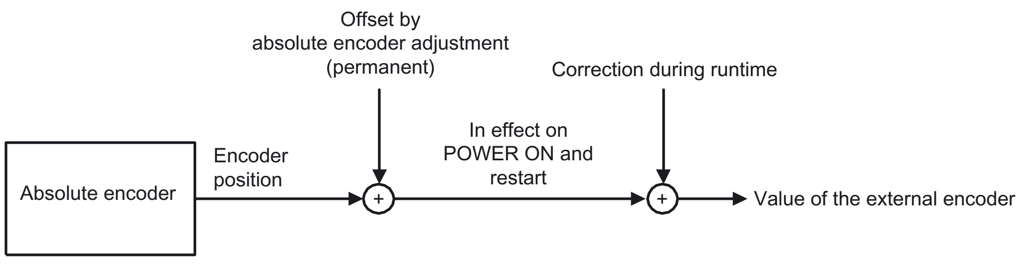

Offset by absolute encoder adjustment (permanent)
Correction during runtime
Absolute encoder
Encoder position
In effect on POWER ON and restart
+
+
Value of the external encoder

Figure 8-5 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 controller, the stored offset is no longer available.

  If the controller 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 calibration

Absolute encoder homing / absolute encoder adjustment (Page 81)

Direct homing/setting the home position (Page 80)

Relative direct homing/relative setting of home position (V3.2 and higher) (Page 80)

Programming External Encoders/Reference 9

9.1 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*.

9.2 Technological alarms

9.2.1 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