Migration Guide SIMATIC S7-31xT to SIMATIC S7-1500(T)

SIMATIC S7-1500T / TIA Portal V14

https://support.industry.siemens.com/cs/ww/en/view/109743136

Siemens Industry Online Support

Warranty and Liability

Note

The Application Examples are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The Application Examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are used correctly. These Application Examples do not relieve you of the responsibility to use safe practices in application, installation, operation and maintenance. When using these Application Examples, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these Application Examples at any time without prior notice.
If there are any deviations between the recommendations provided in these Application Examples and other Siemens publications − e.g. Catalogs − the contents of the other documents have priority.

We do not accept any liability for the information contained in this document.
Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data etc., described in this Application Example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act ("Produkthaftungsgesetz"), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract ("wesentliche Vertragspflichten"). The damages for a breach of a substantial contractual obligation are, however, limited to the foreseeable damage, typical for the type of contract, except in the event of intent or gross negligence or injury to life, body or health. The above provisions do not imply a change of the burden of proof to your detriment.
Any form of duplication or distribution of these Application Examples or excerpts hereof is prohibited without the expressed consent of the Siemens AG.

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.
In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.
Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.
Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit http://www.siemens.com/industrialsecurity.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.
To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under http://www.siemens.com/industrialsecurity.

Table of Contents

Warranty and Liability 2

1 Motivation 6

1.1 Objective of this document 6
1.2 Hardware and software components 6
1.2.1 Validity 6
1.2.2 Components used 7

2 Comparison between the CPU31xT and the S7-1500(T) 9

2.1 General overview 9
2.1.1 SIMATIC S7-31xT architecture 9
2.1.2 SIMATIC S7-1500(T) architecture 10
2.1.3 SIMATIC S7-1500(T) execution system 11
2.1.4 Error concepts 11
2.1.5 Differences in the range of functions 12
2.2 Technologies compared 12
2.3 Technology functions compared 13

3 Sample Migration 18

3.1 Overview 18
3.1.1 Advantages 18
3.1.2 Restrictions 18
3.1.3 Requirement 19
3.1.4 Preparing the STEP 7 project 19
3.2 Step 1: Migrating the user project to TIA Portal 20
3.3 Step 2: Replacing the unspecified CPU 22
3.4 Step 3: Generating technology objects 26
3.5 Step 4: Customizing the program 26
3.6 Step 5: Customizing the HMI 27

4 Programming Tips 28

4.1 Transferring axis objects to user FBs 28
4.1.1 Layered concept of the Axis technology object 28
4.1.2 Transferring axis types to user function blocks 30
4.2 Transferring other technology objects 32
4.2.1 Cam technology object 33
4.2.2 Output Cam technology object 35
4.2.3 Cam Track technology object 36
4.2.4 Measuring Input technology object 38
4.2.5 External Encoder technology object 40
4.3 Creating a cam with the editor 40
4.4 Notes on using technology objects 44
4.4.1 Interpolating a cam 44
4.4.2 Scaling a cam 44
4.4.3 Synchronization using a cam 46
4.4.4 Synchronization of gearing at a standstill 49
4.4.5 Synchronization of camming at a standstill 51
4.4.6 Changing a cam on-the-fly 52
4.4.7 Canceling a synchronous operation 52
4.5 Programming functions for multiple axes 53
4.5.1 Moving the instance data of a function block 54
4.5.2 Arranging the technology objects in an array 54
4.5.3 Example: Acknowledging alarms on many axes in a loop 56
4.5.4 Example: Accessing axis data from the HMI 57
4.6 Providing axis data in the user program 59
4.7 Organization blocks for motion control 60

4.7.1 Basic information about motion control OBs 60
4.7.2 Clock reduction (CPU firmware V1.5 and higher) 61
4.7.3 Processing the motion control functionality 61
4.7.4 Using the MC-PreServo and MC-PostServo OBs 61
4.8 Motion control resources 64
4.8.1 Motion control resources provided 64
4.8.2 Number of cams provided 65
4.8.3 Motion control resources consumed 66
4.8.4 Checking the motion control resource consumption 66

5 Appendix 68

5.1 Information about the technologies 68
5.1.1 Single-axis motion functions 68
5.1.2 Gearing and camming 68
5.1.3 Output cams and cam tracks, measuring inputs and external encoders 69
5.1.4 Path objects 69
5.1.5 Pressure control 69
5.1.6 Hydraulic axes 69
5.2 PLCopen blocks compared 69
5.2.1 MC_Power 72
5.2.2 MC_Reset 75
5.2.3 MC_Home 76
5.2.4 MC_Stop 79
5.2.5 MC_Halt 81
5.2.6 MC_HaltSuperImposed 83
5.2.7 MC_ChangeDataSet 84
5.2.8 MC_MoveAbsolute 85
5.2.9 MC_MoveRelative 88
5.2.10 MC_MoveAdditive 90
5.2.11 MC_MoveSuperImposed 92
5.2.12 MC_MoveVelocity 94
5.2.13 MC_MoveJog 97
5.2.14 MC_MoveToEndPos 99
5.2.15 MC_SetTorqueLimit / MC_TorqueLimit 101
5.2.16 MC_SetCharacteristics 103
5.2.17 MC_ForceLimiting 105
5.2.18 MC_ForceControl 108
5.2.19 MC_GearIn / MC_GearIn 111
5.2.20 MC_GearIn / MC_GearInPos 114
5.2.21 MC_GearOut 117
5.2.22 MC_GearInSuperImposed 118
5.2.23 MC_GearOutSuperImposed 120
5.2.24 MC_Phasing / MC_PhasingRelative 122
5.2.25 MC_Phasing / MC_PhasingAbsolute 124
5.2.26 MC_PhasingSuperImposed 126
5.2.27 MC_CamIn 128
5.2.28 MC_CamOut 132
5.2.29 MC_CamInSuperImposed 133
5.2.30 MC_CamOutSuperImposed 136
5.2.31 MC_CamClear 138
5.2.32 MC_CamSectorAdd 139
5.2.33 MC_CamInterpolate / MC_InterpolateCam 141
5.2.34 MC_GetCamPoint / MC_GetCamFollowingValue 143
5.2.35 MC_SynchronizedMotionSimulation 145
5.2.36 MC_GetCamPoint / MC_GetCamLeadingValue 146
5.2.37 MC_CamSwitch / MC_OutputCam 148
5.2.38 MC_CamSwitchTime / MC_OutputCam 150

5.2.39 MC_CamTrack 152
5.2.40 MC_ReadCamTrackData 155
5.2.41 MC_WriteCamTrackData 156
5.2.42 MC_MeasuringInput / MC_MeasuringInput 157
5.2.43 MC_MeasuringInput / MC_MeasuringInputCyclic 159
5.2.44 MC_MeasuringInput / MC_AbortMeasuringInput 161
5.2.45 MC_ExternalEncoder 163

6 Links & Literature 165

7 History 165

1 Motivation

1.1 Objective of this document

The aim of this documentation is to show users of the Technology CPU 31xT-2 DP / Technology CPU 31xT-3 PN/DP options for migrating to the new CPUs of the S7-1500 and S7-1500T series.

SIMATIC S7-1500 CPUs

SIMATIC S7-1500 CPUs are SIMATIC standard CPUs. They all have selected integrated motion control functions and can therefore be directly used for simple motion control applications without additional hardware and software.

SIMATIC S7-1500T CPUs

SIMATIC S7-1500T CPUs are SIMATIC CPUs that are specifically tailored to the requirements of higher-quality motion control applications and have a greater range of motion control functions than SIMATIC standard CPUs.

Using these CPUs in TIA Portal does not require additional software. S7-1500T CPUs are fully integrated in the TIA Portal engineering framework.

Structure of this document

The aim of this documentation is to answer questions involved in the planning or considerations regarding the migration from the previous Technology CPU 31xT to a SIMATIC S7-1500 / SIMATIC S7-1500T by addressing the following topics in greater detail:

- Comparison of the technologies or technology objects included in the CPUs.
- Comparison of the technology functions or PLCopen blocks that can be used in the individual CPUs.
- Sample migration of a STEP 7 V5.5 user project with the aid of TIA Portal for further use as the basis for a project with a SIMATIC S7-1500 / SIMATIC S7-1500T.
- Various programming tips to make migrating from STEP 7 V5.5 to TIA Portal and SIMATIC CPUs of the S7-1500 / S7-1500T series easier.
- The Appendix compares the interfaces of the PLCopen blocks in the different CPUs, including information about aspects that need to be considered, especially when transferring a user program.

To migrate a user project from STEP 7 V5.5 to TIA Portal, it is sufficient to read Chapters 1 to 3. All other chapters of this document are intended to provide you with additional information for revising the user project in TIA Portal after the successful migration.

1.2 Hardware and software components

1.2.1 Validity

This documentation is valid for

- TIA Portal or STEP 7 V14 and higher
- SIMATIC S7-1500 firmware version V2.0 and higher or SIMATIC S7-1500T firmware version V2.0 and higher

Note Migrating a project requires that the migration functionality of TIA Portal, STEP 7 V5.x, S7 Technology V4.x and, if necessary, WinCC flexible 2008 with the required service packs (SP) and hotfixes (HF) be installed on the PG/PC. Otherwise, the TIA Portal migration wizard cannot perform the migration.

1.2.2 Components used

This documentation was created with the following components:

Hardware components

Table 1-1

Component	No.	Article number	Note
SIMATIC Technology CPU 31xT			
Technology CPU 317T-2 DP	1	6ES7317-6TK13-0AB0	CPU317T-2 DP, 1024KB Firmware version: V2.7.2
Technology CPU 317T-3 PN/DP	1	6ES7317-7TK10-0AB0	CPU317T-3 PN/DP, 1024KB Firmware version: V3.2.12
SIMATIC S7-1500			
SIMATIC CPU 1516-3 PN/DP	1	6ES7516-3AN01-0AB0	CPU 1516-3 PN/DP, 1MB FOR PROGRAM, 5MB FOR DATA Firmware version: V2.0.3
SIMATIC S7-1500T			
SIMATIC CPU 1515T-2 PN	1	6ES7515-2TM01-0AB0	CPU 1515T-2 PN, 750KB FOR PROGRAM, 3MB FOR DATA Firmware version: V2.0.3

Software components

Table 1-2

Component	No.	Article number	Note
SIMATIC Technology CPU 31xT			
SIMATIC STEP 7 V5.5 software	1	6ES7810-4CC10-0YA5	STEP7 V5.5 SP4 – FLOATING LICENSE
S7 Technology V4.2 option package	1	6ES7864-1CC42-0YA5	
SIMATIC S7-1500 / SIMATIC S7-1500T			
TIA Portal V14	1		Always included in the individual components as an engineering framework.
STEP 7 Professional V14	1	6ES7822-1AA04-0YC5	SIMATIC STEP 7 PROF. V14 FLOATING LICENSE; engineering software in TIA Portal.

Sample files and projects

The following list contains all files and projects that are used in this example.

Table 1-3

Component	Note
109743136_S7-1500T_ MigrationGuide_DOC_v10_en.pdf	This document.

2 Comparison between the CPU31xT and the S7-1500(T)

The following chapters compare the CPU types of the CPU31xT from Step7 V5.5 and S7 Technology V4.2 with the CPU S7-1500 and S7-1500T from TIA Portal V14 and provide a comparison of the available functionalities.

2.1 General overview

The two CPU types of the SIMATIC S7-31xT and SIMATIC S7-1500(T) essentially differ in their architectures and the sequence models of the PLC program.

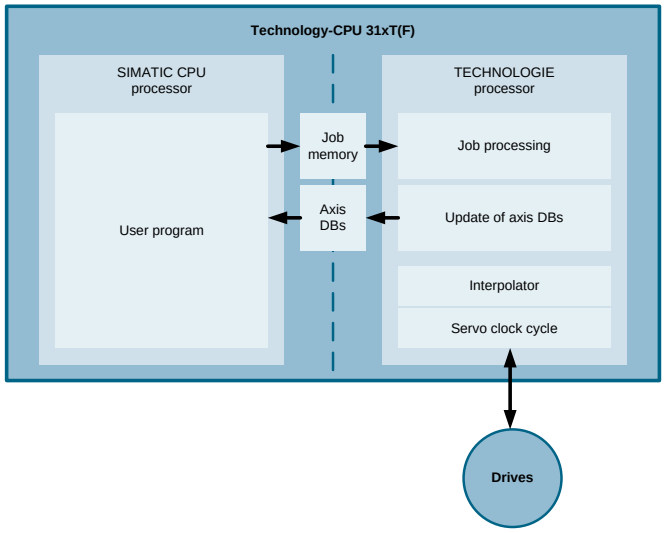
2.1.1 SIMATIC S7-31xT architecture

The previous Technology CPU 31xT was implemented as a dual-processor system, with one processor dedicated to processing the SIMATIC user program and the second one dedicated to processing the technology functions. Both processors basically run asynchronously to one another.

To control the technology, for example the axes connected to the technology CPU, the SIMATIC CPU uses the PLCopen function blocks of the technology functions to send jobs to the technology processor via a job memory. At the same time, the technology processor returns the current status of the running jobs to the SIMATIC CPU.

The status of the technology objects can be determined in the SIMATIC CPU using the technology data blocks, for example the axis DBs, that are cyclically updated by the technology processor.

Figure 2-1 Technology CPU 31xT(F)



In the architecture of the previous Technology CPU 31xT, the job memory and the technology data blocks are thus the link between the two CPU processors.

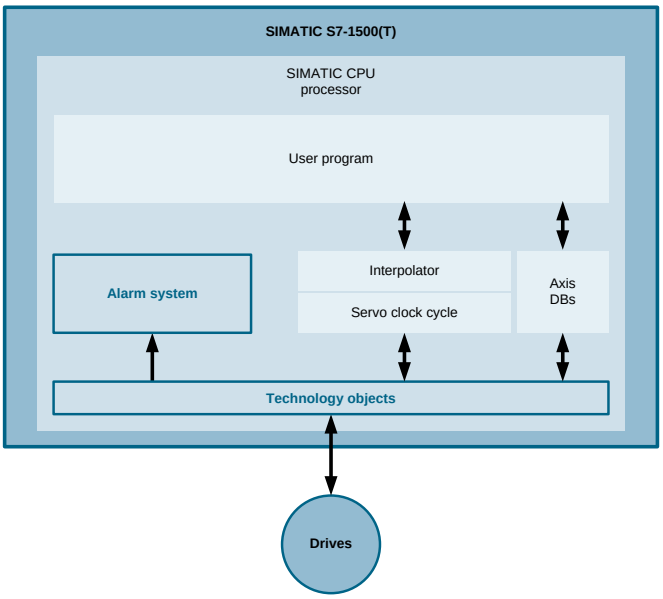
2.1.2 SIMATIC S7-1500(T) architecture

In contrast, the CPUs of the SIMATIC S7-1500 and SIMATIC S7-1500T series are implemented as a high-performance single-processor system. Therefore, the technology is an integral part of the SIMATIC CPU.

This allows the user program to send new technology jobs at any time via the PLCopen function blocks that become effective immediately in the next interpolator cycle.

The technology objects are accessed via the technology data blocks, for example the axis DBs. However, the current status of the technology objects and the parameters of a technology object can now be accessed symbolically directly from the user program without using a PLCopen function block.

Figure 2-2 SIMATIC S7-1500(T)



The major advantage of this single-processor architecture of the CPUs of the SIMATIC S7-1500 and SIMATIC S7-1500T series is the permanent availability of the parameters and data of the technology or technology objects. It is not necessary to request this data using PLCopen functions and there is no deterministic job processing behavior of the PLCopen function blocks as these are executed directly in the next interpolator cycle.

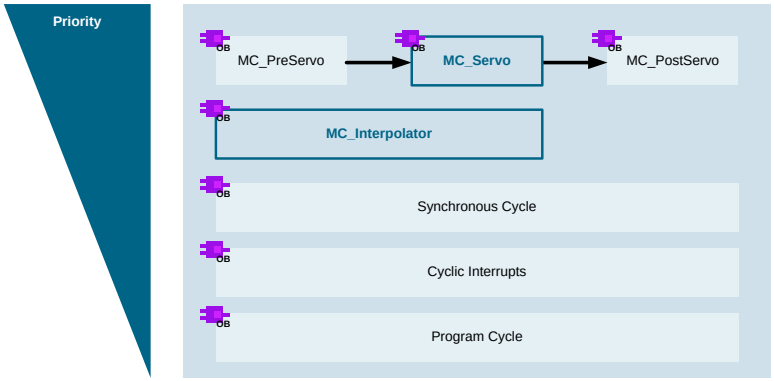
Note The PLCopen functions for data exchange between the PLC and the technology part in the SIMATIC S7-31xT (e.g., MC_ReadSysParameter) are no longer necessary in the CPUs of the SIMATIC S7-1500 and SIMATIC S7-1500T series. The data of the technology objects can now be accessed directly from the user program, which is why these blocks are no longer supported in the SIMATIC S7-1500 / SIMATIC S7-1500T.

2.1.3 SIMATIC S7-1500(T) execution system

The execution system of the CPUs of the SIMATIC S7-1500 and SIMATIC S7-1500T series consists of multiple organization blocks (OBs) with different priority classes that can be used to appropriately structure a user program.

The technology functions of the SIMATIC S7-1500 and SIMATIC S7-1500T are computed in the MC_Servo and MC_Interpolator organization blocks that are managed by the SIMATIC S7-1500(T) operating system and cannot be accessed by the user. These organization blocks belong to a higher priority class and can therefore interrupt most other SIMATIC S7-1500(T) OBs to achieve high control quality and safe functioning of the technology functions.

Figure 2-3 SIMATIC S7-1500(T) execution system



For particularly dynamic tasks or special matching of I/O signals in the context of the technology functions, program parts of the user program can be programmed in the MC-PreServo organization block for processing directly before the position control cycle and in the MC-PostServo organization block for processing directly after the position control cycle. This virtually enables executing parts of the user program synchronously with the technology's position control cycle.

2.1.4 Error concepts

The two CPU types of the SIMATIC S7-31xT and SIMATIC S7-1500(T) also have different error concepts.

SIMATIC S7-31xT error concept

The SIMATIC S7-31xT reports errors in the technology at different locations in the user program:

- In the event of axis errors or errors on a technology object, the last three errors of the axis or technology object in the technology data block are reported to the user program.
- Other errors resulting from applying the technology functions to a technology object are reported to the user program via the Error or ErrorID output of the technology function's PLCopen function block.

Therefore, error handling of the SIMATIC S7-31xT must be implemented at these two reporting locations in the user program.

SIMATIC S7-1500(T) error concept

The SIMATIC S7-1500(T), by contrast, automatically reports axis errors or errors on a technology object to TIA Portal or the HMI without having to create a user program for this purpose. This information can also be viewed using the SIMATIC S7-1500(T) Web server.

Due to the architecture of the SIMATIC S7-1500(T), all error messages of the SIMATIC and the technology are directly displayed in an alarm window of TIA Portal.

In addition, the output error codes of the SIMATIC S7-1500(T)'s technology objects and technology functions were revised and now provide more detailed information about the problem that has occurred. The error code of the main problem is now directly output in the technology object's technology data block, for example for stopping an axis.

For evaluation in the user program, the SIMATIC S7-1500(T) contains one error word and one warning word for each technology object. The current error state of the technology object can be retrieved via their bits.

Note The bit map of the error word and warning word of the SIMATIC S7-1500(T) is not identical with the bit map from the SIMATIC S7-31xT. When migrating a user program, special attention must be paid to the transfer of the program.

Likewise, the error classes of a connected drive defined according to the PROFIdrive standard are automatically reported to the diagnostic system or HMI. However, to accurately determine the cause of the error, this cause must be explicitly analyzed on the drive.

2.1.5 Differences in the range of functions

Finally, you are provided with information on the differences in the range of functions of the SIMATIC S7-31xT and the SIMATIC S7-1500(T).

For the SIMATIC S7-1500(T) firmware version looked at here, the CPU's functional focus is on applications in the field of motion control in interaction with electrically driven axes.

Currently, the SIMATIC S7-1500(T) does not yet have special functions for hydraulic applications as integrated in the SIMATIC S7-31xT. Nevertheless, such functions can be created by application-specific adaptation in the SIMATIC S7-1500(T), for example, in the MC-PostServo organization block.

2.2 Technologies compared

The following list compares the technologies integrated in the different CPU types so that the required CPU type can be preselected before migrating from the CPU31xT to the S7-1500(T).

Table 2-1 Technologies compared

Technology	S7-31xT	S7-1500	S7-1500T
Single-axis motion functions	Yes	Yes	Yes
Gearing	Yes	Limited	Yes
Camming	Yes	No	Yes
Output cams and cam tracks	Yes	Yes	Yes
Measuring inputs	Yes	Yes	Yes
External encoders	Yes	Yes	Yes
Path objects	Yes	No	No
Pressure control	Yes	No	No
Hydraulic axes	Yes	Specific to application	Specific to application

Note In terms of gearing, the SIMATIC standard CPUs of the **S7-1500 series** provide the **relative gearing** functionality. This means that synchronization always takes place immediately and based on the dynamic response settings. The position reference between the master axis and the slave axis cannot be set. If you want to set the reference between the master axis and the slave axis, synchronize the axes when they are at a standstill.
When gearing is active, the reference can be changed by the MC_MoveSuperImposed technology function.

A CPU of the **S7-1500T series** allows **absolute gearing**, including setting a synchronous position, actual value coupling and phase shift between master axis and slave axis.

2.3 Technology functions compared

The following list compares the technologies functions available in the different CPU types sorted by the technologies listed in the previous chapter.

Note Due to the hardware configuration, different technology functions of the CPU31xT on which the motion control functions are computed on a separate processor are not required in the S7-1500 / S7-1500T. These technology functions are marked accordingly.

Note At present, the range of functions of the S7-1500 / S7-1500T is not yet complete. It will be continuously extended by future firmware versions.

Note All technology functions for motion functions in the S7-1500 / S7-1500T currently have no mode input for defining the override response of the technology function. Regarding their response, these technology functions override one another, i.e., a currently active motion function is overridden and thus stopped by starting another motion function for the same technology object.

Technology functions in the CPU types of the S7-1500 / S7-1500T series marked with the icon are only available in the S7-1500T CPUs.

Table 2-2 Technology functions compared

Technology function	S7-31xT	S7-1500 / S7-1500T
Single-axis motion functions		
Enable/disable axis	MC_Power	MC_Power
Home/set axis	MC_Home	MC_Home
Stop axis and prevent new traversing jobs	MC_Stop	[Not available]
Normal stop	MC_Halt	MC_Halt
Superimposed normal stop	MC_HaltSuperImposed	[Not available]
Change data set	MC_ChangeDataSet	In the user program, several parameters can be changed synchronously with the motion control organization blocks.
Move axis in jog mode	[Not available]	MC_MoveJog
Absolute positioning	MC_MoveAbsolute	MC_MoveAbsolute
Relative positioning	MC_MoveRelative	MC_MoveRelative
Relative positioning to current target position	MC_MoveAdditive	This function can be implemented specific to an application in the user program.
Superimposed positioning	MC_MoveSuperImposed	MC_MoveSuperImposed
Motion with speed preset	MC_MoveVelocity	MC_MoveVelocity
Move to fixed stop	MC_MoveToEndPos	MC_TorqueLimit
Change encoder	[Not available]	MC_SetSensor
Enable/disable torque limit	MC_SetTorqueLimit	MC_TorqueLimit
Enable valve characteristic	MC_SetCharacteristics	[Not available]
Force/pressure limiting	MC_ForceLimiting	[Not available]
Force/pressure control	MC_ForceControl	[Not available]
Gearing		
Start relative gearing	MC_GearIn	MC_GearIn
Start absolute gearing		MC_GearInPos

Technology function	S7-31xT	S7-1500 / S7-1500T
Stop gearing	MC_GearOut	Calling another motion function overrides the function.
Start superimposed gearing	MC_GearInSuperImposed	[Not available]
Stop superimposed gearing	MC_GearOutSuperImposed	[Not available]
Absolute phase shift between master axis and slave axis	MC_Phasing	MC_PhasingAbsolute
Relative phase shift between master axis and slave axis		MC_PhasingRelative
Superimposed phase shift between master axis and slave axis	MC_PhasingSuperImposed	[Not available]
Set active synchronous operation to "Simulation" mode	[Not available]	MC_SynchronizedMotion Simulation
Camming		
Start camming	MC_CamIn	MC_CamIn
Stop camming	MC_CamOut	Calling another motion function overrides the function.
Start superimposed camming	MC_CamInSuperImposed	[Not available]
Stop superimposed camming	MC_CamOutSuperImposed	[Not available]
Absolute phase shift between master axis and slave axis	MC_Phasing	[Not available]
Relative phase shift between master axis and slave axis		[Not available]
Superimposed phase shift between master axis and slave axis	MC_PhasingSuperImposed	[Not available]
Clear cams	MC_CamClear	Can be implemented in the user program by direct access to the cam (DB).
Add cam sector	MC_CamSectorAdd	Can be implemented in the user program by direct access to the cam (DB).
Interpolate cams	MC_CamInterpolate	MC_InterpolateCam
Read point on slave axis from cam	MC_GetCamPoint	MC_GetCamFollowing Value
Read point on master axis from cam		MC_GetCamLeading Value

2.3 Technology functions compared

Technology function	S7-31xT	S7-1500 / S7-1500T
Output cams and cam tracks		
Distance output cam	MC_CamSwitch	MC_OutputCam
Time-based output cam	MC_CamSwitchTime	MC_OutputCam
Cam track	MC_CamTrack	MC_CamTrack
Read cam track	MC_ReadCamTrackData	Can be implemented in the user program by direct access to the cam track (DB).
Write cam track	MC_WriteCamTrackData	
Measuring inputs		
Single measurement job	MC_MeasuringInput	MC_MeasuringInput
Cyclic measurement job		MC_MeasuringInput Cyclic
Abort measurement job		MC_AbortMeasuringInput
External encoders		
External encoder	MC_ExternalEncoder	The following functions can now also be applied to an external encoder: MC_Power MC_Reset MC_Home
Path objects		
Position path object absolute along a linear path	MC_MoveLinearAbsolute	[Not available]
Position path object relative along a linear path	MC_MoveLinearRelative	[Not available]
Position path object absolute along a circular path	MC_MoveCircularAbsolute	[Not available]
Position path object relative along a circular path	MC_MoveCircularRelative	[Not available]
Move path object along circular paths	MC_MoveCircles	[Not available]
Position path object absolute along a polynomial path	MC_MovePolynomAbsolute	[Not available]
Position path object relative along a polynomial path	MC_MovePolynomRelative	[Not available]
Move path object along interpolation point table	MC_MovePath	[Not available]
Synchronize path object with conveyor belt	MC_GroupSyncConveyorBelt	[Not available]
Configure offset between base coordinate system and object coordinate system	MC_SetCartesianTransform	[Not available]

2.3 Technology functions compared

Technology function	S7-31xT	S7-1500 / S7-1500T
Set object coordinate system to new conveyor belt position	MC_RedefineTrackingPos	[Not available]
Prepare and load interpolation point data for path segment	MC_PathSelect	[Not available]
Stop path motion	MC_GroupStop	[Not available]
Interrupt path motion	MC_GroupInterrupt	[Not available]
Continue path motion	MC_GroupContinue	[Not available]
Monitor zones	MC_ZoneCheck	[Not available]
Basic functions		
Acknowledge error	MC_Reset	MC_Reset
Activate/deactivate technology object	MC_ActivateTO	[Not available]
Activate/deactivate DP slave	MC_ActivateDPSlave	[Not available]
Read parameters	MC_ReadSysParameter	[Not necessary]
Write parameters	MC_WriteParameter	[Not necessary]
Read technology I/O	MC_ReadPeriphery	[Not necessary]
Write technology I/O	MC_WritePeriphery	[Not necessary]
Read data record	MC_ReadRecord	[Not necessary]
Write data record	MC_WriteRecord	[Not necessary]
Read drive parameters	MC_ReadDriveParameter	This function can be implemented specific to an application in the user program. For more information, please refer to Links & Literature \5\ of this document.
Write drive parameters	MC_WriteDriveParameter	

Note Chapter 5.2 of this documentation provides a detailed comparison of the PLCopen blocks and shows the differences between the individual function blocks.

3 Sample Migration

3.1 Overview

The examples provided in the following chapters give you an overview of how to migrate an existing SIMATIC S7-31xT user project to TIA Portal in order to use the user blocks of the previous project as the basis for a TIA Portal project with the SIMATIC S7-1500(T).

NOTICE	**Do not perform an uncontrolled migration of a user program!** Even if program parts and technology functions can be replaced by functions of the same name when migrating, this does not ensure the functional equality of the user program before and after the migration. Therefore, perform a comprehensive function test of your user program on the target machine after the migration.

3.1.1 Advantages

Migrating an existing SIMATIC S7-31xT user project to TIA Portal offers the following advantages:

- Transfer of the user blocks you have created, including all the necessary user-defined data types, to TIA Portal as a basis of development for your new TIA Portal project with the SIMATIC S7-1500(T).
- Transfer of the screens and functions of an included HMI as a basis of development for your new TIA Portal project.
- Time savings when establishing the basis of development for the new TIA Portal project.

3.1.2 Restrictions

Migrating an existing SIMATIC S7-31xT user project to TIA Portal involves the following restrictions:

- A migration of the full functionality of a user project with the SIMATIC S7-31xT to TIA Portal to the SIMATIC S7-1500(T) can currently not be ensured.
- Likewise, the functional equality of the technology functions of the SIMATIC S7-31xT and the SIMATIC S7-1500(T) cannot be ensured.
- Neither is it ensured that the block parameters of the technology functions of the SIMATIC S7-31xT and the SIMATIC S7-1500(T) are the same.
- Know-how protected blocks in the user project of the SIMATIC S7-31xT for which no password and no block source code exist cannot be migrated to TIA Portal.
  If necessary, remove the blocks from the SIMATIC S7-31xT user project before the migration.

3.1.3 Requirement

To migrate a STEP 7 project in TIA Portal, the following requirements must be met in TIA Portal or on the PG/PC on which the migration is to be performed:

- In TIA Portal, the Migration function must be installed.
- For the migration, the complete software with which the STEP 7 project to be migrated was created must be installed on the PG/PC.
  The reason for this is that TIA Portal uses this software when migrating; therefore, this software must be installed on the PG/PC, including the required licenses.

Note Migrating a project requires that the migration functionality of TIA Portal, STEP 7 V5.x, S7 Technology V4.x and, if necessary, WinCC flexible 2008 with the required service packs (SP) and hotfixes (HF) be installed on the PG/PC. Otherwise, the TIA Portal migration wizard cannot perform the migration and aborts the migration with an error message.

Note If you want to perform a migration, it is recommended to install, if possible, TIA Portal on the PG/PC that was previously used for developing the STEP 7 projects for the SIMATIC 31xT.

3.1.4 Preparing the STEP 7 project

Before the migration in TIA Portal, the STEP 7 project should be prepared as described here.

First, make sure that all of the following requirements are met:

- The complete STEP 7 project is available for the migration, i.e., the project to be migrated contains all block sources and user-defined data types.
- The STEP 7 project does not contain any know-how protected blocks.
  - If the project contains know-how protected blocks, please remove their know-how protection before the migration.
  - For know-how protected blocks, please also make sure that after removing know-how protection, any block sources and user-defined data types that may be necessary are included in the STEP 7 project.

Before performing the migration in TIA Portal, the STEP 7 project should be compiled one more time in its entirety in the previous engineering system:

- In the STEP 7 V5.x block folder, check the consistency of all blocks by completely recompiling all blocks of the block folder.
- If the STEP 7 project contains an HMI, remove the temporary files from the project in WinCC flexible.
- In WinCC flexible, completely regenerate the HMI and save the changes to the STEP 7 project.

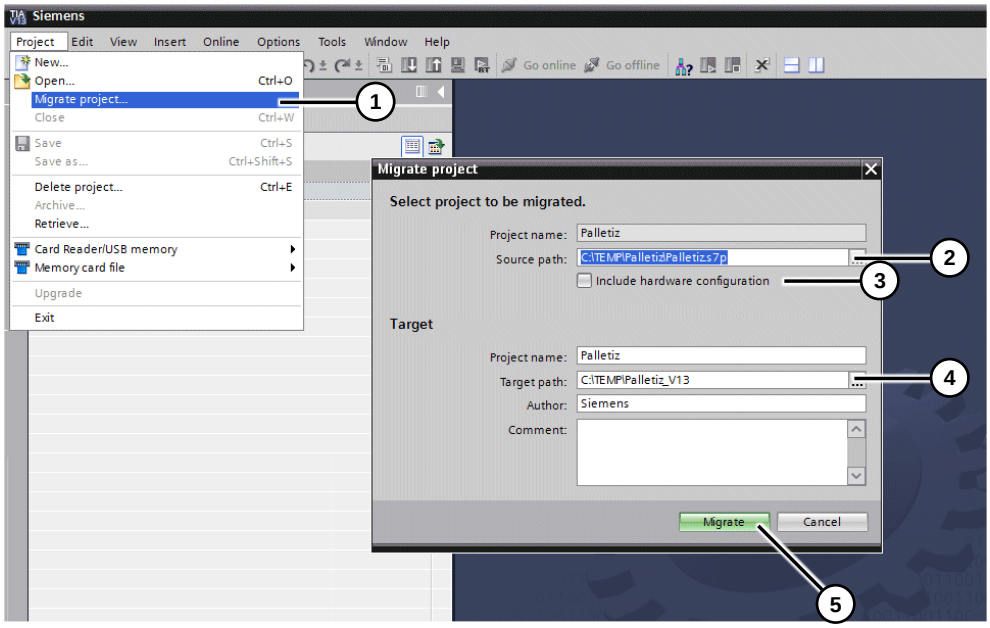
Note When checking the consistency or compiling the blocks in the STEP 7 V5.x block folder, warning messages occur when compiling the project's technology data blocks; however, these messages do not affect the subsequent migration to TIA Portal.

3.2 Step 1: Migrating the user project to TIA Portal

If all requirements are met on the PG/PC and the STEP 7 project to be migrated was prepared as described, you can start the migration in TIA Portal:

1. Start TIA Portal and open the Migrate project dialog.
2. Select the STEP 7 project to be migrated.
3. Select migration without hardware configuration.
4. Define the target project where you want to store the migrated data.
5. Start the migration.

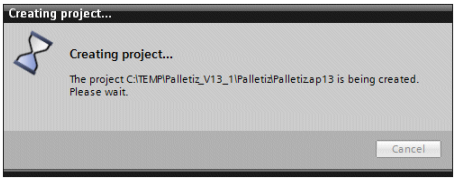
Figure 3-1 Migrating the user project to TIA Portal



Note The migration of a user project with the SIMATIC S7-31xT to TIA Portal must be performed without a hardware migration. Therefore, uncheck "Include hardware configuration" before starting the migration.

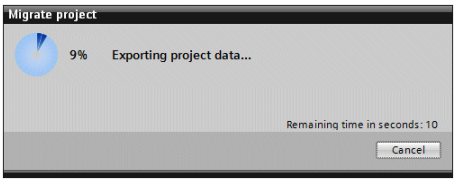
The TIA Portal target project to which the user project is to be saved after the migration is created first.

Figure 3-2 Creating target project



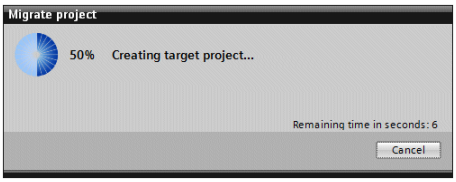
Then the project data is exported from the source project.

Figure 3-3 Exporting project data



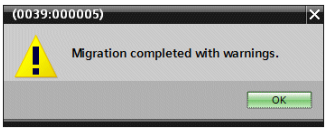
Finally, the project data is saved to the target project.

Figure 3-4 Saving data to target project



If errors or warnings have occurred during the migration, this is indicated by a message at the end of the process.

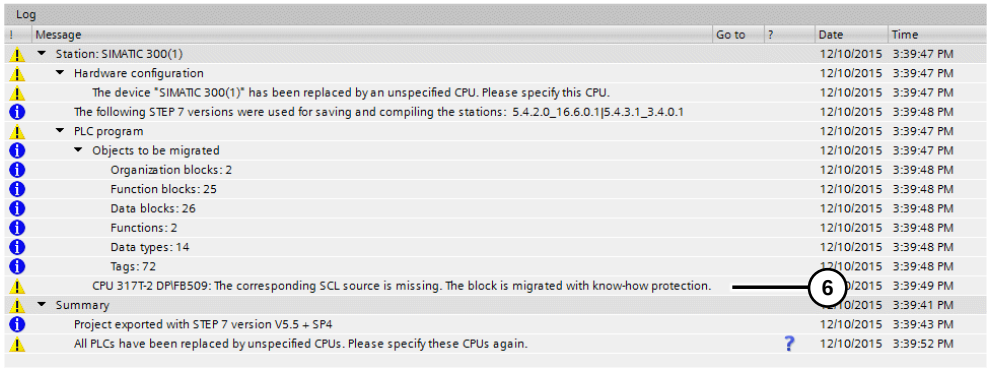
Figure 3-5 Completing migration



Refer to the migration log file for unusual activities that have occurred during the migration of the user project.

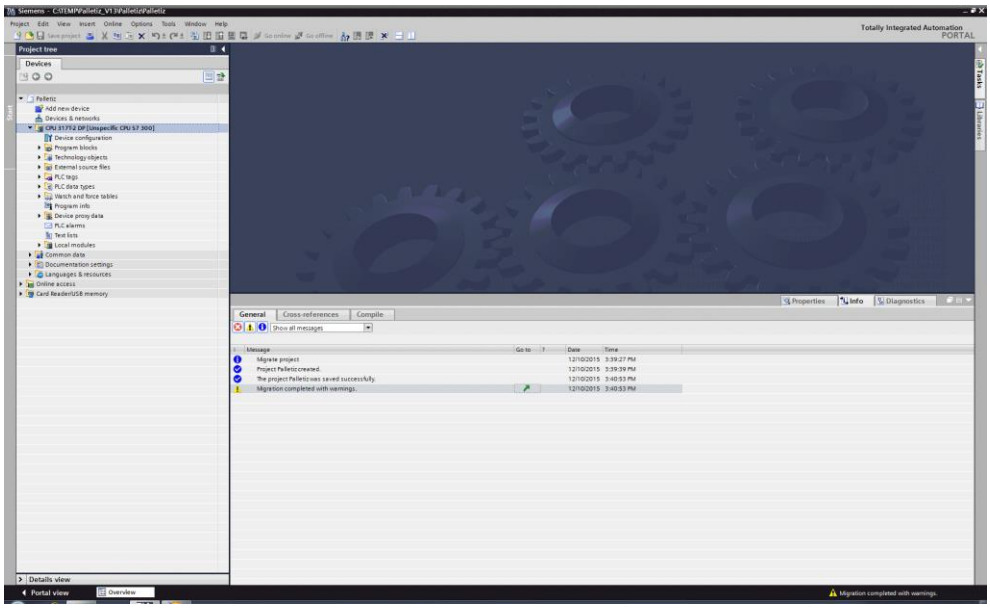
The following figure shows a migration log file. The message (6) indicates that block FB 509 was migrated without existing SCL source code and is therefore stored as a know-how protected block.

Figure 3-6 Migration log file



During the migration, the SIMATIC S7-31xT was replaced with an unspecified S7-300 CPU.

Figure 3-7 TIA Portal project after the migration



3.3 Step 2: Replacing the unspecified CPU

In the second step, replace the unspecified S7-300 CPU currently integrated in the TIA Portal project with a SIMATIC S7-1500 CPU.

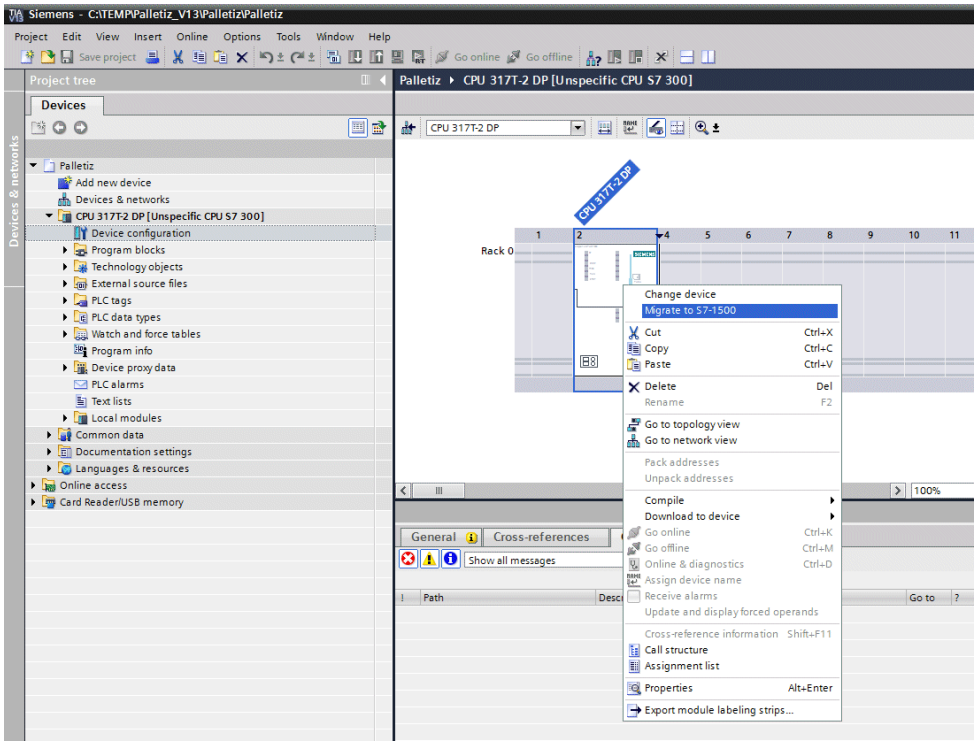
When migrating the CPU, use the below criteria to choose between a SIMATIC S7-1500 and a SIMATIC S7-1500T:

- What technologies do you want to use in the user program?
  Chapter 2.2 of this documentation provides more information about the technologies integrated in the two CPU types.

- What axis quantity frameworks do you want to control using the CPU?
  The TIA Selection Tool and SIZER for Siemens Drives configuration tools help you answer this question. In the "Links & Literature" chapter of this documentation, you will find links to these two tools.
- What cycle times do you want to achieve in the user program?
  For more information, please refer to the manuals or technical specifications of the individual CPUs or the TIA Selection Tool. In the "Links & Literature" chapter, you will find the link to this tool.
  For a first rough approximation, you can start with the position controller cycle settings from the S7-300 CPU. However, motion control should not require more than 50% of the CPU computing power.
- What technology functions do you want to use in the user program?
  More information can be found in Chapter 2.3 or Chapter 5.2 of this documentation.

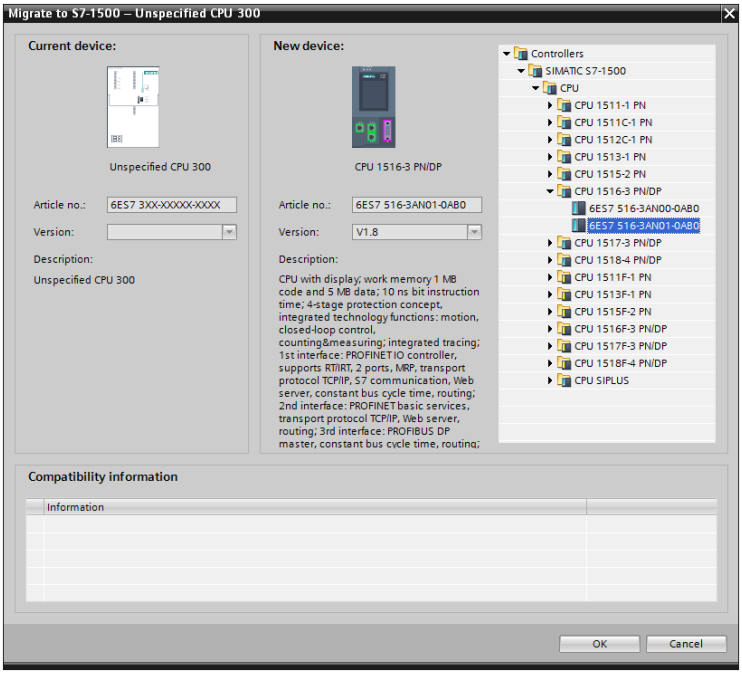
The CPU is replaced in the hardware configuration of TIA Portal with the aid of the CPU migration to S7-1500 function. This migrates the unspecified S7-300 CPU to a SIMATIC S7-1500 CPU.

Figure 3-8 CPU migration to S7-1500



Then the following selection dialog allows you to select the desired CPU; in this example: an S7-1516.

Figure 3-9 Target CPU selection dialog



Now an unspecified SIMATIC S7-300 CPU is migrated to a SIMATIC S7-1500(T) CPU.

Figure 3-10 Security note for testing the user program

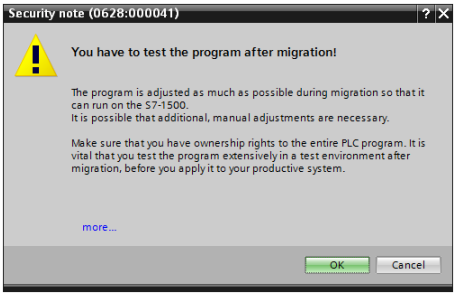
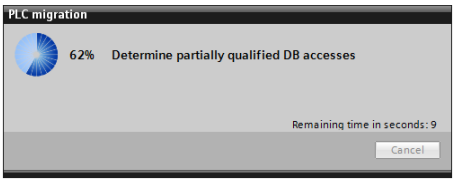


Figure 3-11 CPU migration



At the end of the migration, an error message is displayed. However, this is due to the fact that not all blocks of the user program, especially the technology data blocks of the technology objects used and various technology function blocks, can be applied to the user program of the S7-1500. The link in the info pane (1) for the migration log information allows you to display the exact reasons for the error message.

Figure 3-12 Migration completes with an error message

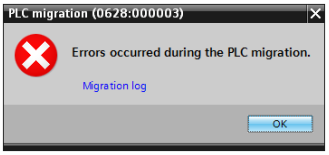
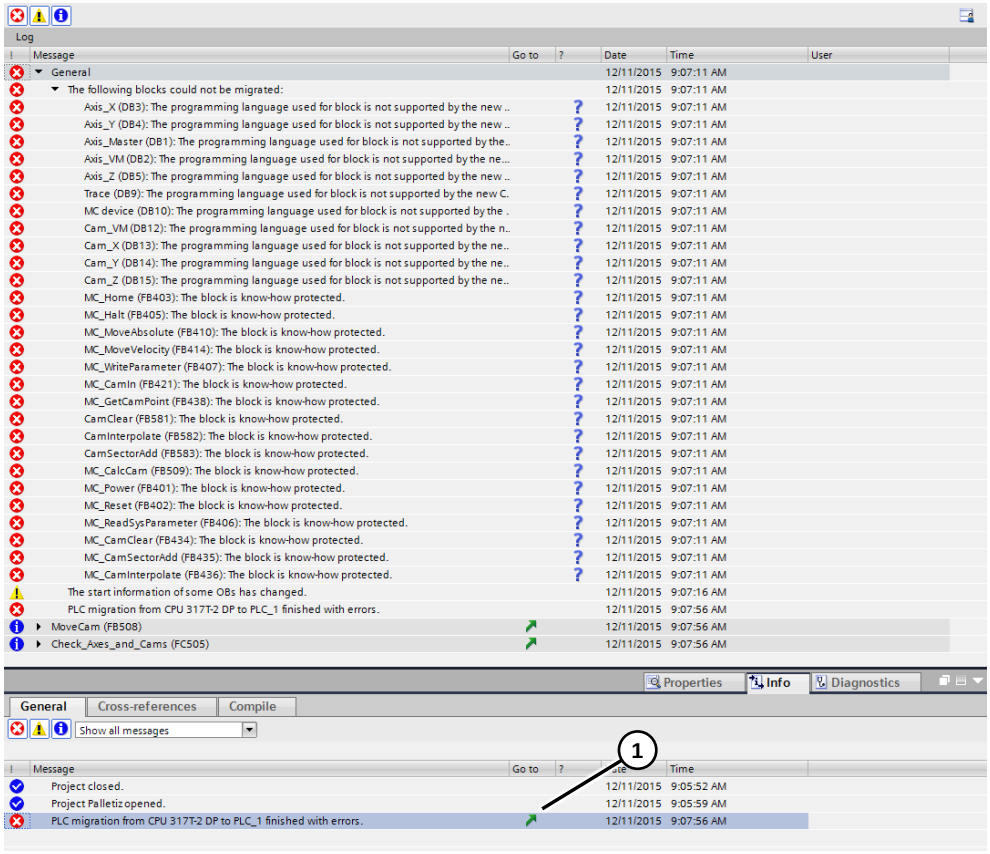


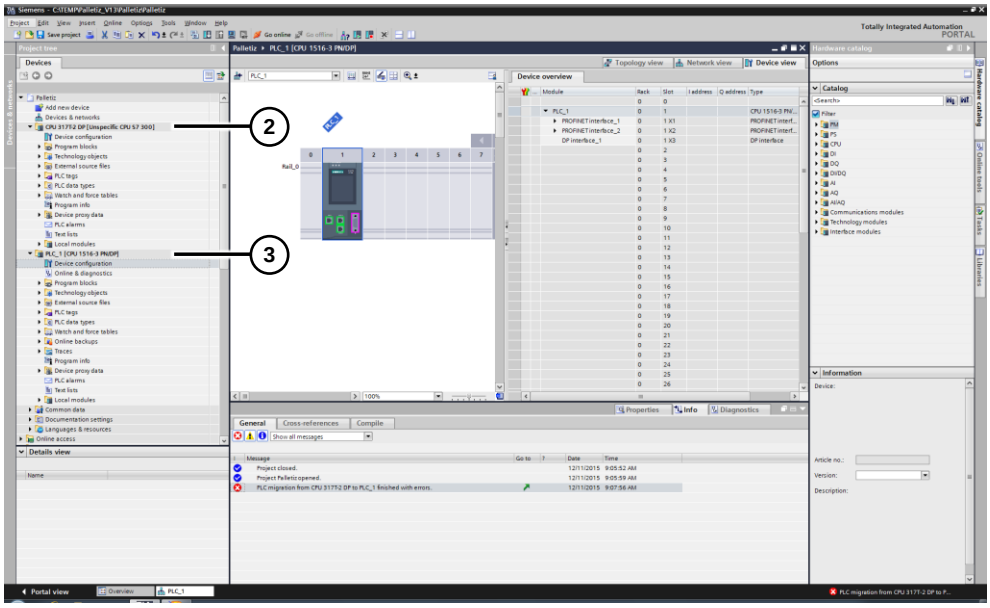
Figure 3-13 Display of the migration log information



After selecting and replacing the CPU, the migrated user project looks as follows:

- The first folder (2) contains the migration source project with the unspecified S7-300 CPU.
  This folder can be deleted when the user program has been migrated completely and successfully.
- The second folder (3) contains the migration target project with the SIMATIC S7-1500(T) CPU.
  This folder is then used to perform the next steps necessary for migrating the user program. These steps are described in the following chapters.

Figure 3-14 TIA Portal project after the migration



3.4 Step 3: Generating technology objects

Now you have to create the required technology objects such as axes, measuring inputs, cams and output cams in the migration target project.

The technology objects, especially the axes, must be configured based on the mechanical conditions of the machine or plant. The instructions given in the S7-1500(T) technology manual should be followed.

Note When creating technology objects in the migration target project, it may be useful to use the configuration of the technology objects in STEP 7 V5.x as a template or setting aid and apply the values set in this configuration.

3.5 Step 4: Customizing the program

To complete the migration of the user program, make the following changes in the user program:

- Integrate the technology objects generated in the previous chapter into the user program.
- Replace or integrate the required technology function blocks or PLCopen function blocks into the user program.
  Pay particular attention to a potentially changed meaning of the blocks' input parameters or block modes and customize the user program accordingly.
  In some cases, it is also necessary to replace technology functions that are not available in the S7-1500(T) with application solutions.

- Carefully check the programming and functionality of the remaining program parts of the user program.
- Compile the complete user program in the target project.

NOTICE	**Do not perform an uncontrolled migration of a user program!** Even if program parts and technology functions can be replaced by functions of the same name when migrating, this does not ensure the functional equality of the user program before and after the migration. Therefore, perform a comprehensive function test of your user program on the target machine after the migration.

3.6 Step 5: Customizing the HMI

Finally, customize any existing HMI. If the HMI was successfully migrated to TIA Portal, it is normally sufficient to reconnect the HMI tags to the appropriate tags from the user program and recompile the entire HMI project.

Note It is recommended to implement the tag connection between the HMI and the user program via a central global data block in the user program.

4 Programming Tips

The aim of this chapter is to provide support in creating the program using selected examples.

4.1 Transferring axis objects to user FBs

If you have to control multiple axes in an automation project in a SIMATIC controller, it is usually helpful to structure and encapsulate the functions of the individual axes in a higher-level function block.

The following chapters show you how to transfer an Axis technology object to a self-created function block via an input parameter in order to use the object in this block in interaction with the technology functions that are possibly called as a multi-instance in that place.

4.1.1 Layered concept of the Axis technology object

In terms of functionality, the Axis technology objects build on each other. This is also referred to as a layered concept of the Axis technology objects in the SIMATIC S7-1500 / S7-1500T.

The individual technology objects build on each other as shown in the below figure.

Figure 4-1 Layered concept of the Axis technology objects

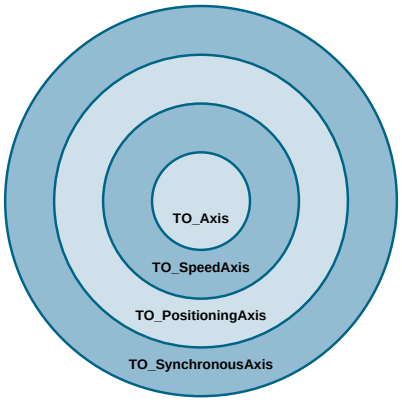


Table 4-1 The Axis technology objects

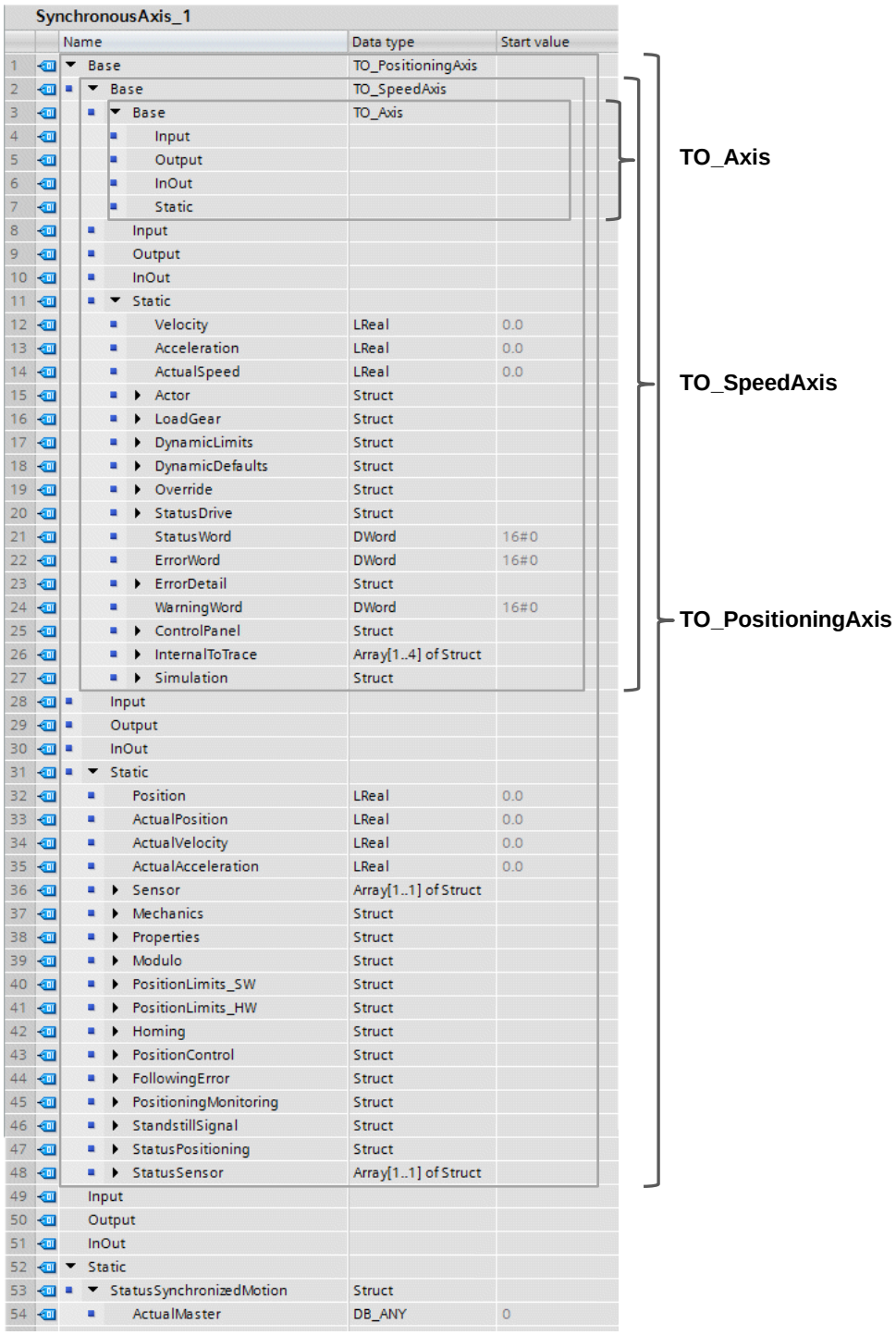
Technology object	Data type	Description
Axis base type	TO_Axis	This data type forms the basis of all Axis technology objects.
Speed Axis	TO_SpeedAxis	Considering the dynamic response settings, the Speed Axis technology object calculates speed setpoints and outputs them to the drive. All motions of the speed axis are speed-controlled.

Technology object	Data type	Description
Positioning Axis	TO_PositioningAxis	Considering the dynamic response settings, the Positioning Axis technology object calculates position setpoints and outputs appropriate speed setpoints to the drive. All motions of the positioning axis are position-controlled. For absolute positioning, the Positioning Axis technology object must know the physical position.
Synchronous Axis	TO_SynchronousAxis	The Synchronous Axis technology object contains all the functions of the Positioning Axis technology object. In addition, a synchronous axis can follow the motions of a master axis. The synchronous operation relationship between master axis and slave axis is defined by a function f(x).

This layered concept is also reflected in the structure of the technology data block of the technology objects of the SIMATIC S7-1500 / S7-1500T. To illustrate this, the following figure shows a technology data block of a synchronous axis.

- The TO_Axis data type forms the base type of the technology object; however, it does not provide any other functionality in the technology data block of the axis.
- The TO_SpeedAxis data type that has the functionality of a speed axis is based on this data type. This data type is also used to display all parameters of a speed axis in the technology data block, for example the current setpoint velocity (Velocity), the current setpoint acceleration (Acceleration) or the current actual speed of the motor (Actual Speed).
  In addition, this data type includes all parameters for configuring an axis as all axis types are based on a speed axis, i.e., open-loop speed control or closed-loop speed control by a drive.
- Then the TO_PositioningAxis data type that has the functionality of a positioning axis is based on this data type. In the technology data block, this data type provides all information about a positioning axis such as the setpoint position for a positioning operation (Position), the actual axis position (Actual Position), the actual axis velocity (Actual Velocity) and the actual axis acceleration (Actual Acceleration).
  In addition, this data type includes all related parameters of a positioning axis such as encoder data (Sensor), modulo settings (Modulo), travel path settings (PositionLimits_SW, PositionLimits_HW, Homing), position control settings (PositionControl) and the associated monitoring (FollowingError, PositionMonitoring, StandstillSignal).
- Finally, the technology data block includes, of course, the TO_SynchronousAxis data type of the actual synchronous axis. This data type includes all the data types listed above; however, it does not provide any further axis information.
  As additional information, the Synchronous Axis data type contains status data on the current synchronous operation with a master axis.

Figure 4-2 Layered concept using the example of the technology DB of a synchronous axis



4.1.2 Transferring axis types to user function blocks

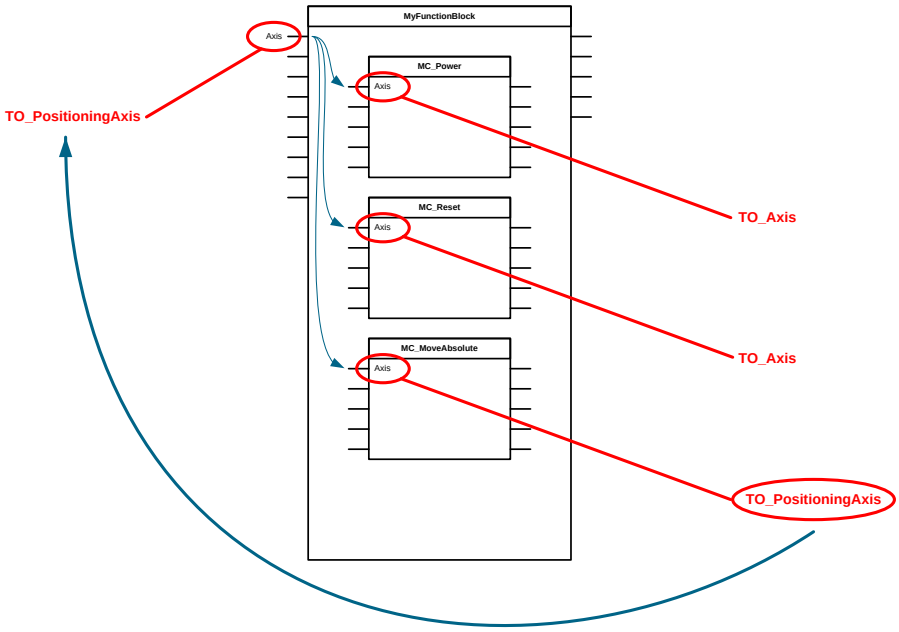
If you want to structure a program, it is generally useful to encapsulate the functions of individual axes in a user function block. It should be possible to transfer the Axis technology object to the user function block via an input parameter.

To achieve this, create an appropriate parameter in the user function block's interface definition. Manually set the parameter data type to the appropriate axis type to be transferred by manually entering one of the following data types in the parameter's data type settings:

- For a speed axis, enter the TO_SpeedAxis data type.
- For a positioning axis, enter the TO_PositioningAxis data type.
- For a synchronous axis, enter the TO_SynchronousAxis data type.

When selecting the correct data type, also pay attention to the technology functions included in the user function block. Select the data type such that it corresponds to the most significant data type of the technology functions used. When, for example, the MC_MoveAbsolute technology function that requires the TO_PositioningAxis axis data type is used in the user function block, the function block's input parameter must be at least of the TO_PositioningAxis or TO_SynchronousAxis type.

Figure 4-3 Transferring axis types to user block



Note The data type of the input parameter for transferring a technology object to a user function block must be entered manually in order to use it in the function block. It is not possible to directly select the data type from the drop-down list.

4.2 Transferring other technology objects

For structured programming or encapsulating the technological functions, you can also transfer the other available technology objects to a function block via input parameters.

The following table shows the required data types of the input parameters for the respective technology object.

Table 4-2 Data types of the other technology objects

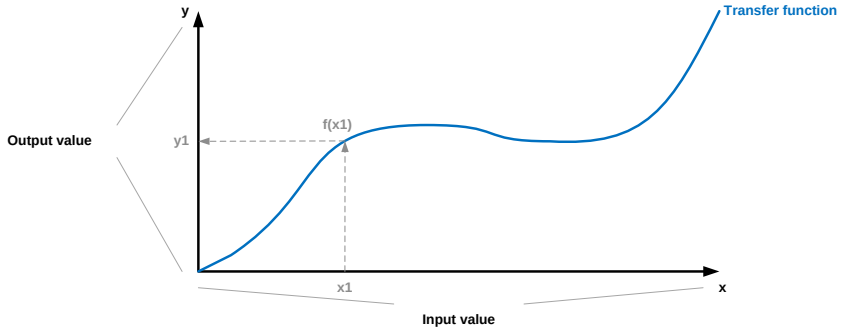
Technology object	Data type	Description
Cam	TO_Cam	The Cam technology object defines a transfer function y = f(x). On a unit-neutral basis, this transfer function describes the dependence of an output variable on an input variable, e.g. the position of a master axis relative to a slave axis. A Cam technology object can be used multiple times. The cam's transfer function can be composed of individual polynomial segments and/or defined by individual interpolation points of an interpolation point table.
Output Cam	TO_OutputCam	The Output Cam technology object generates switching signals depending on the position of an axis or external encoder. The switching states can be evaluated in the user program and connected to digital outputs. You can define one output cam per technology object. If necessary, assign the technology object to an Axis technology object when inserting into the user program or before compiling.
Cam Track	TO_CamTrack	The Cam Track technology object generates a switching signal sequence depending on the position of an axis or external encoder. A cam track can consist of up to 32 single output cams and be output to an output. The switching states can be evaluated in the user program or connected to digital outputs. If necessary, assign the technology object to an Axis technology object when inserting into the user program or before compiling.

Technology object	Data type	Description
Measuring Input	TO_MeasuringInput	When there is a signal change at the measuring input, the Measuring Input technology object detects the actual position of an axis or external encoder. If necessary, assign the technology object to an Axis technology object when inserting into the user program or before compiling.
External Encoder	TO_ExternalEncoder	The External Encoder technology object detects a position and provides it to the controller. The actual position detected by the external encoder can be used, for example, for the following functions: • Measured-value acquisition by a measuring input. • Position-dependent generation of switching signals and switching signal sequences by output cams and cam track with actual value reference. • As a master value for a synchronous axis (S7-1500T only).

4.2.1 Cam technology object

The Cam technology object provides a unit-neutral transfer function y = f(x) between an input variable x and an output variable y. For camming between two axes, for example, the master axis position is the input variable x while the slave axis position as the output variable y results from the cam's transfer function.

Figure 4-4 Cam as a transfer function

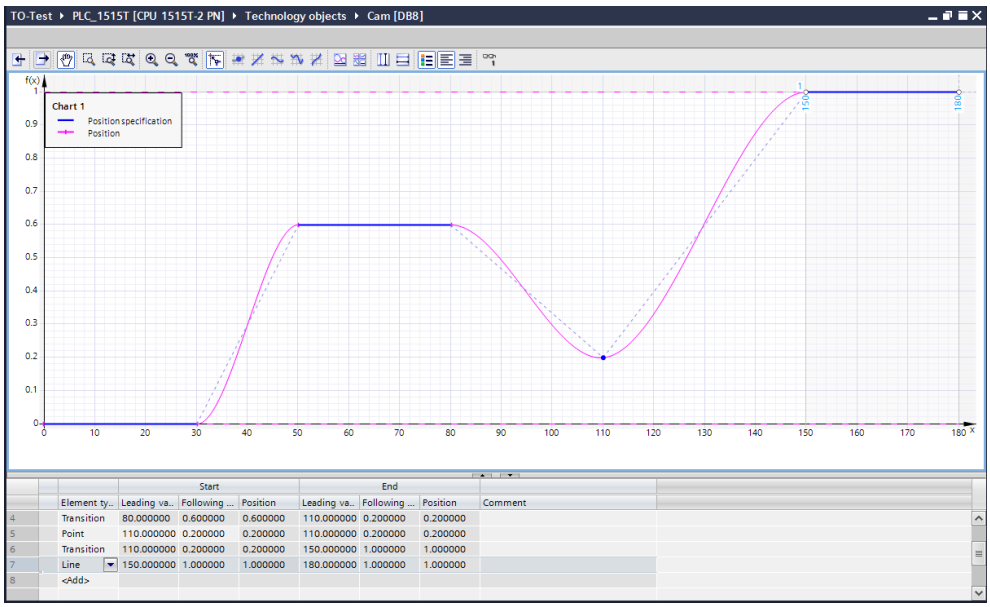


The cam's transfer function can be generated in the controller in two different ways:

- During the engineering phase of the user program in TIA Portal.
  To this end, TIA Portal provides a Cam Editor with a comprehensive range of functions. It provides helpful support in entering and optimizing the transfer function based on user-definable criteria.

- During user program runtime in the controller by the user program itself.
  The cam data of the transfer function can be specified or changed by the user program directly in the technology data block of the technology object.

Figure 4-5 Cam Editor

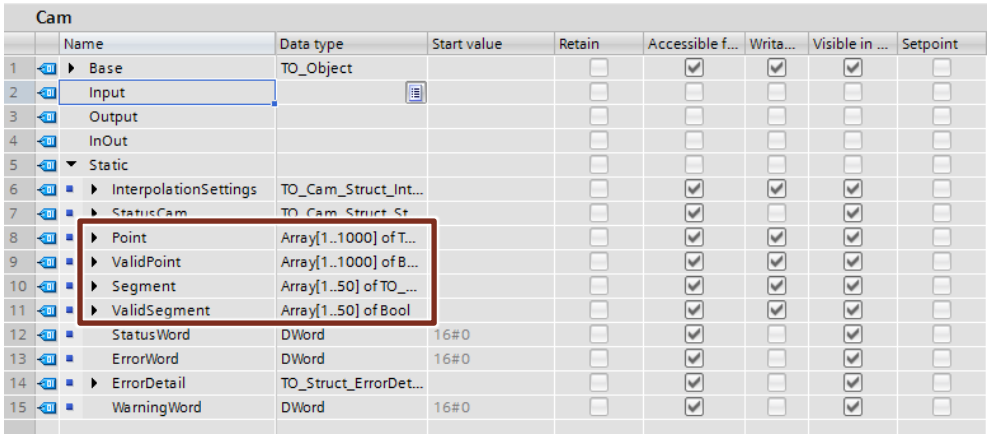


The cam data of the transfer function can also be defined in two ways during both the engineering phase using the Cam Editor and during runtime by the user program:

- As an interpolation point table by defining the pairs of values (x, y) for the input and output variable and the associated interpolation rule between the individual pairs of values. The following interpolation options are available:
  - Linear (connecting the interpolation points with a straight line)
  - Bezier spline (B-spline)
  - Cubic spline (C-spline)
- As a 6th order polynomial by defining the individual polynomial coefficients and any trigonometric (sine) part and the range of validity of the polynomial relative to the input variable.

The cam data of the transfer function can also be defined segment by segment as an interpolation point table in some cases and as a polynomial in other cases. If you want to define the cam data at runtime, two different tables are available in the technology data block for this purpose. One table for the interpolation points and one table for entering the polynomial segments. Do not forget to enable the individual table entries with the appropriate valid bit.

Figure 4-6 Technology data block of the cam



Note The cam data entered using the Cam Editor during the engineering phase is also stored in the technology object's data block in the appropriate tables. In these tables, the data can be read or changed during user program runtime.

Note Before its first use in a user program, the Cam technology object must be interpolated. In this context, it is irrelevant whether the cam was created in the user program at runtime or whether it was created and downloaded using TIA Portal.

4.2.2 Output Cam technology object

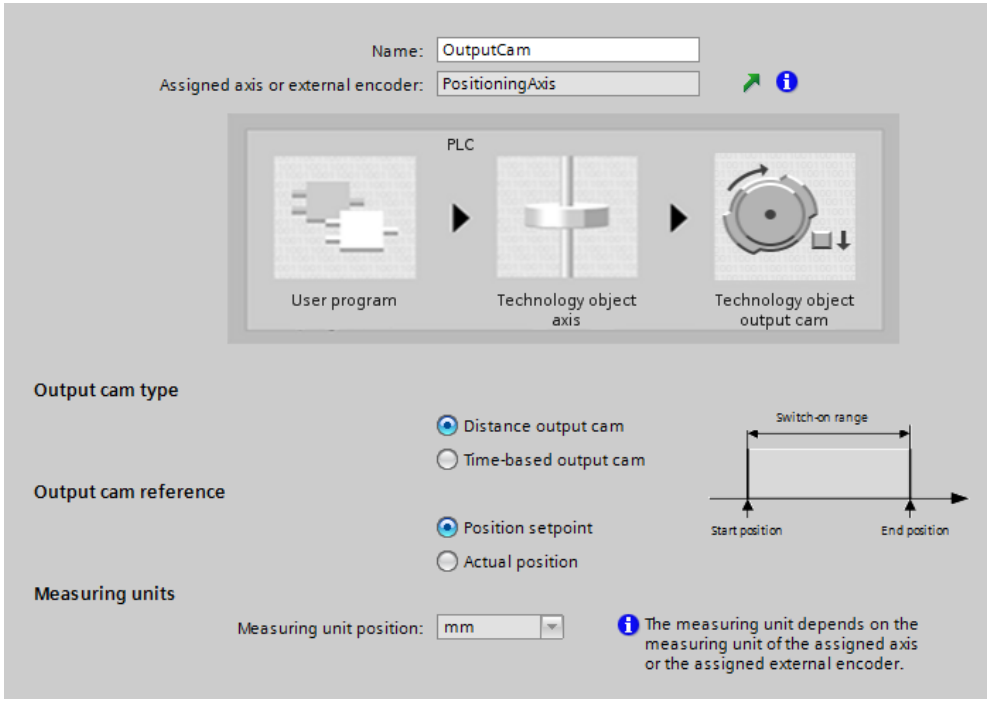
The Output Cam technology object generates switching signals depending on the position of an axis or external encoder.

When defining the output cam, you can choose between two output cam types:

- Distance output cam
  Start and end, i.e. the activation and deactivation position of the output cam, are each defined by a position value of the assigned axis.
- Time-based output cam
  The start of the output cam is defined by a position value of the assigned axis. The end or activation duration of the output cam is defined by a time value. Passing the defined start position starts switching signal output of the output cam. In this context, the direction of motion of the assigned axis is irrelevant. For this reason, the position of time-based output cams is not, as is the case with distance output cams, clearly defined in the position range of the assigned axis. Depending on the direction of motion, the output cam can be to the left or right of the defined start position.

In addition, you can define the evaluation of the axis position of the assigned axis for the output cam. The cam can be output either relative to the axis setpoint or relative to the current axis position. If switching signal output is relative to the current axis position, a suitable hysteresis should always be defined in the output cam's advanced settings to prevent the switching signal from toggling in the range of the output cam's start and end point.

Figure 4-7 Output Cam technology object



Note In the output cam's advanced settings, switching delay compensation can be performed at the signal outputs used for output cam output.

Note High-precision output of output cam switching signals can be achieved by using the TM Timer technology modules on the ET 200SP / ET 200MP distributed I/O system.

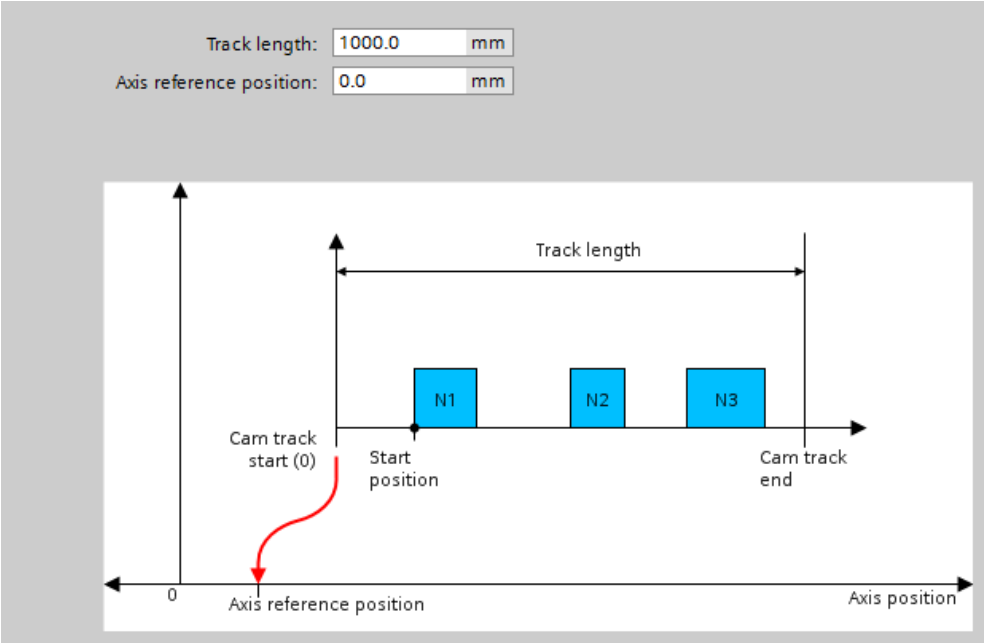
4.2.3 Cam Track technology object

The Cam Track technology object generates a switching signal sequence depending on the position of an axis or external encoder. A cam track can consist of up to 32 single output cams and be output to an output.

Here, just as with the Output Cam technology object, you can choose between the distance output cam and time-based output cam types and the setpoint and actual value axis reference of the assigned axis. However, the settings can only be made for all output cams of the cam track.

In addition, the cam track can also be homed to a defined position of the assigned axis. In this process, the cam track is defined with a user-definable track length starting at position zero. Using the home position, position zero can then be assigned to the cam track of a defined axis position.

Figure 4-8 Mapping the cam track to a defined axis range



In the cam track's advanced settings, a table allows you to additionally check the settings of the cam track's individual output cams.

Figure 4-9 Defining the start and end of the cam track's individual output cams

Set output cam type: distance output cam

Output cam	Valid	Start position		End position	
1	☐	0.0	mm	0.0	mm
2	☐	0.0	mm	0.0	mm
3	☐	0.0	mm	0.0	mm
4	☐	0.0	mm	0.0	mm
5	☐	0.0	mm	0.0	mm
6	☐	0.0	mm	0.0	mm
7	☐	0.0	mm	0.0	mm
8	☐	0.0	mm	0.0	mm
9	☐	0.0	mm	0.0	mm
10	☐	0.0	mm	0.0	mm
11	☐	0.0	mm	0.0	mm
12	☐	0.0	mm	0.0	mm
13	☐	0.0	mm	0.0	mm
14	☐	0.0	mm	0.0	mm
15	☐	0.0	mm	0.0	mm
16	☐	0.0	mm	0.0	mm
17	☐	0.0	mm	0.0	mm
18	☐	0.0	mm	0.0	mm
19	☐	0.0	mm	0.0	mm
20	☐	0.0	mm	0.0	mm

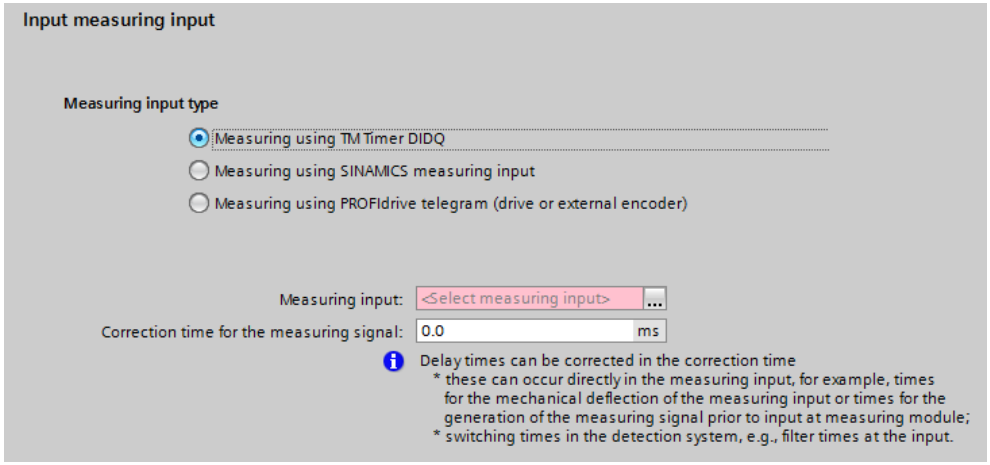
4.2.4 Measuring Input technology object

When there is a signal change at the measuring input, the Measuring Input technology object detects the actual position of an axis or external encoder.

As a measuring input, the following input types can be defined at the measuring input:

- Measuring using TM Timer DIDQ
  Technology module of the ET 200SP / ET 200MP distributed I/O system for high-precision measurements. Measured-value acquisition is implemented with the aid of time stamps and output on the technology function used for measured-value acquisition as a high-precision position value.
- Measuring using SINAMICS measuring input
  Use of an input on the control unit of the SINAMICS drive system that can be parameterized in the drive as a measuring input or a technology module on the SINAMICS drive system. This, too, allows high-precision measurements. Here, too, measured-value acquisition is implemented with the aid of time stamps and output on the technology function used for measured-value acquisition as a high-precision position value.
- Measuring using PROFIdrive telegram
  When measuring using the PROFIdrive telegram, the measuring input is connected directly to the drive unit and the measurement is implemented in the drive. Using the PROFIdrive telegram, the drive or encoder module directly returns the detected position value to the technology object.
  When measuring using the PROFIdrive telegram, only one measuring input for an actual value or encoder in the PROFIdrive telegram can be active at a time. However, not more than two measuring inputs can be configured for an actual value or encoder in the PROFIdrive telegram.

Figure 4-10 Selecting the signal input for the measuring input



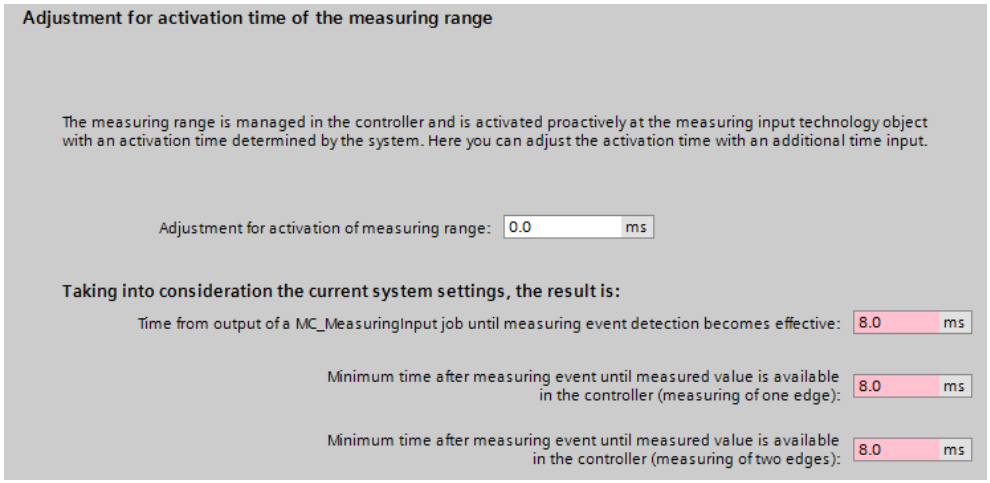
Note Using the Measuring Input technology object, you can also perform measurements on virtual axes. However, the "Measuring using PROFIdrive telegram" input type cannot be used for this purpose.

The measuring input's advanced settings allow you to make various compensation settings for the measuring input signal to increase the accuracy of the measurements taken. The time of measurement is corrected during this process.

Corrections can be necessary, for example, in the following cases:

- Consideration of times for mechanical deflection of the measuring input.
- Consideration of times for generating the measuring signal in front of the input on the measurement module, for example the dead time of a photoelectric barrier or similar times.
- Consideration of filter times at the measuring signal input or filter times for the measuring inputs on a SINAMICS drive unit.

Figure 4-11 Compensation setting for measuring input



Note The parameterized correction times are always included in measured-value acquisition. This applies to both measured-value acquisition using time stamps and direct measured-value acquisition in the drive using the PROFIdrive telegram.

By selecting the suitable technology function for the measuring input, you can choose between the following two measurement types:

- Single measurement using MC_MeasuringInput
  One measurement job allows edge-precise acquisition of up to two measured values. The measurement can be performed for two rising edges, two falling edges or one rising and one falling edge each.
- Cyclic measurement using MC_MeasuringInputCyclic
  Cyclic measurement allows edge-precise acquisition of up to two measured values in each servo clock cycle.
  In this mode, however, prompt processing in each servo clock cycle must be ensured in the user program; otherwise, the measured values could be overwritten by the MC_MeasuringInputCyclic technology function in the next servo clock cycle and would therefore be lost.

Note Cyclic measurement can only be used in conjunction with measured-value acquisition using time stamps. This mode also requires that measured-value acquisition be connected, e.g. the drive or distributed I/O system, using isochronous data exchange (IRT).

4.2.5 External Encoder technology object

The External Encoder technology object detects a position and provides it to the controller.

The external encoder has almost the same setting options as the measurement system of an Axis technology object.

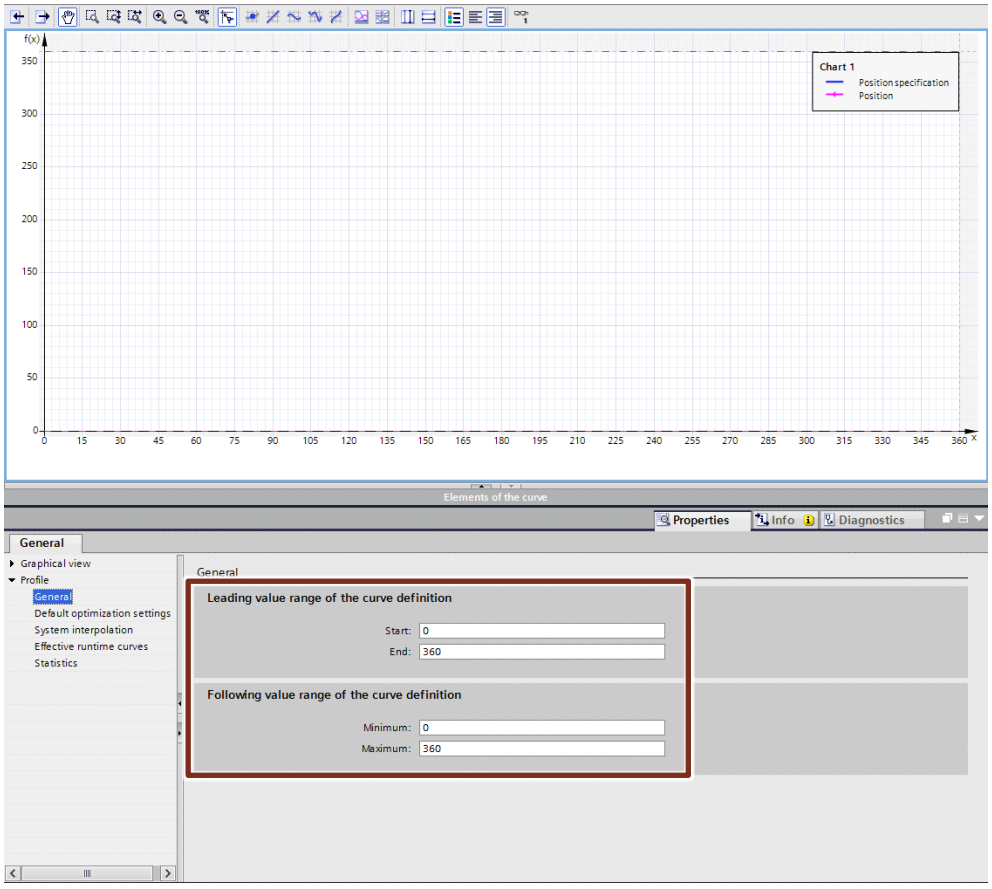
Note An external encoder cannot be used for position control of an axis, i.e., for integration into the technology object's position control loop.

4.3 Creating a cam with the editor

The aim of this chapter is to provide a brief introduction to using the TIA Portal Cam Editor. This should enable you to create a desired cam as quickly as possible or create a basis for further editing and optimizing the cam.

First set the definition range for the master axis and the slave axis in the cam properties in the general profile settings.

Figure 4-12 Setting the definition range of the cam

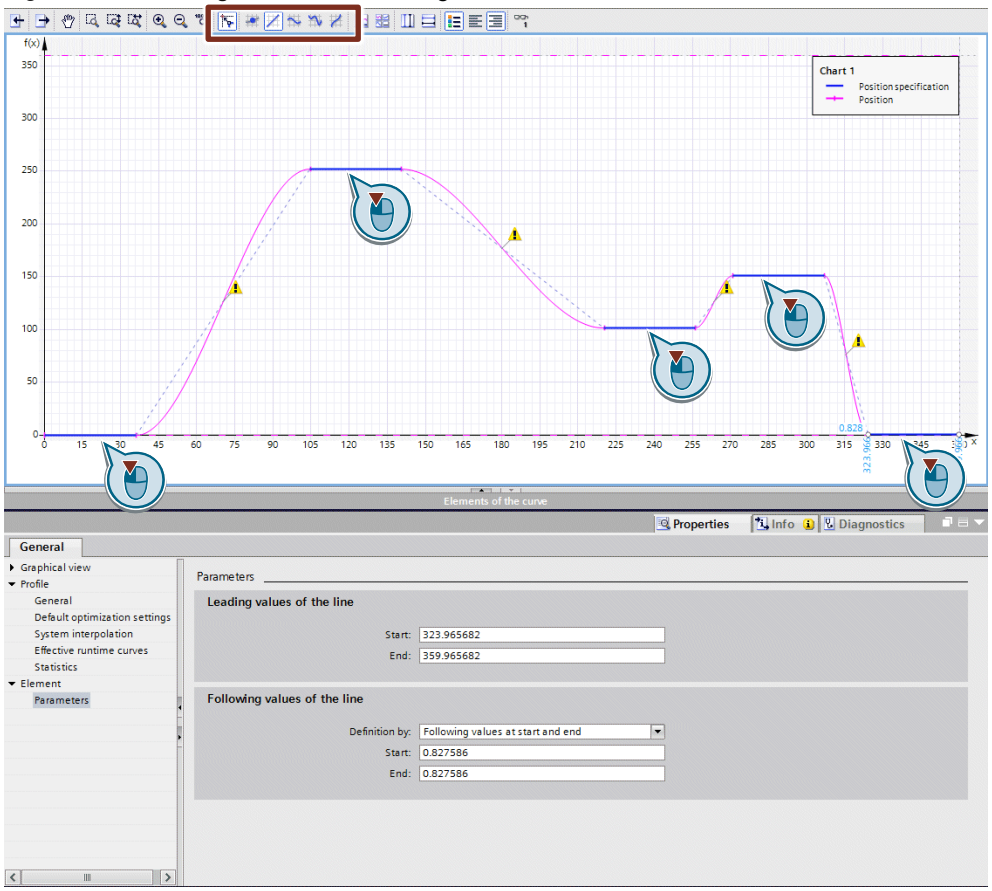


Note To have the greatest possible range for editing the cam in the Cam Editor, it is recommended to minimize the cam elements overview table. The necessary settings can be made in the properties of the individual elements.

The next step allows you to roughly distribute, in the cam's canvas, the exactly defined motion segments (work segments) of the slave axis, e.g. dwell ranges, i.e., zero speed ranges of the slave axis or 1:1 synchronous operation ranges between master axis and slave axis, as a 45° line in the cam definition range.

The desired line type can be selected in the top part of the canvas for the cam.

Figure 4-13 Entering the cam's "work segments"



Note At this point of the cam definition, a rough distribution regarding the position and order of the known motion segments is perfectly sufficient.

The motion segments are precisely adjusted and defined in the next step. In the cam element's Properties dialog, this is done individually for each cam element.

Note

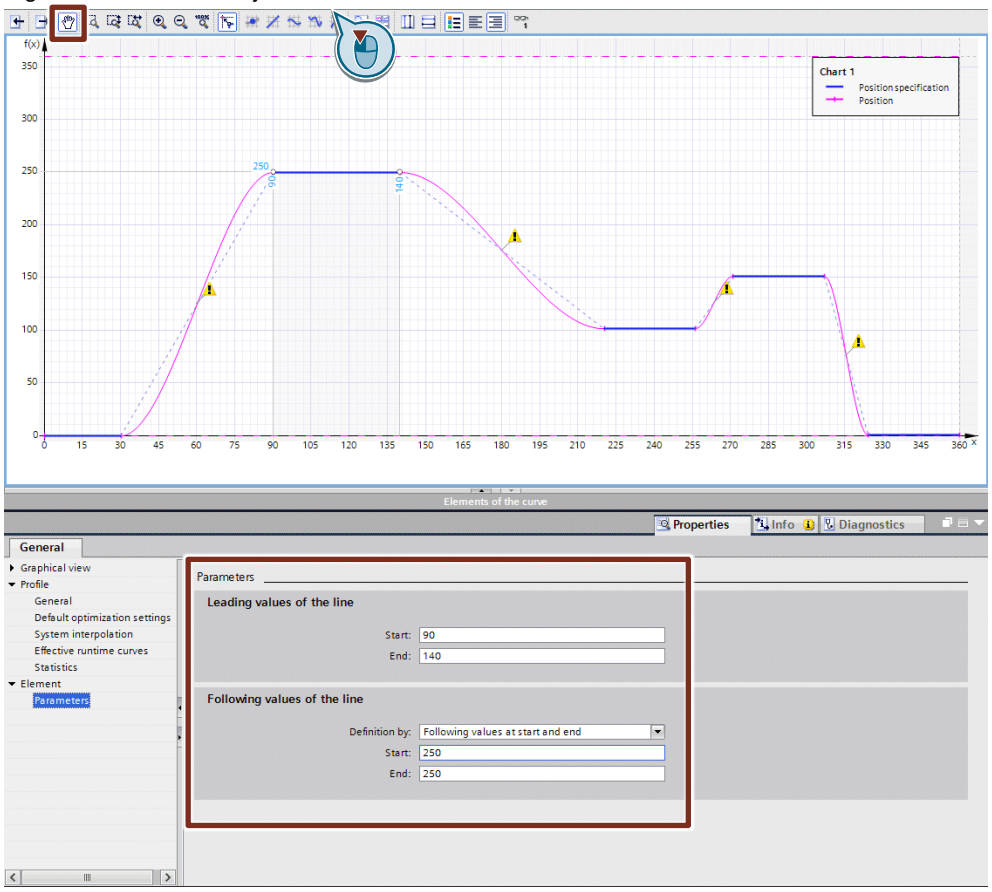
The inserted cam sectors are automatically inserted by the Cam Editor based on the set conditions.

The editor's default settings are normally sufficient for a first cam development. Elements added automatically will be optimized in a later step.
The basic settings for automatically added elements can be changed in the Properties dialog in Default optimization settings and System interpolation.

Now any motion segment inserted manually can be clicked in the Cam Editor's canvas and the element's exact position can be set in the Properties dialog of this cam element.

This exactly aligns and arranges the precisely defined motion segments of the slave axis within the cam. Areas where an exact correlation between the motion of master axis and slave axis is not required are closed via the automatically added elements of the Cam Editor.

Figure 4-14 Precise adjustment of the individual cam sectors



Note

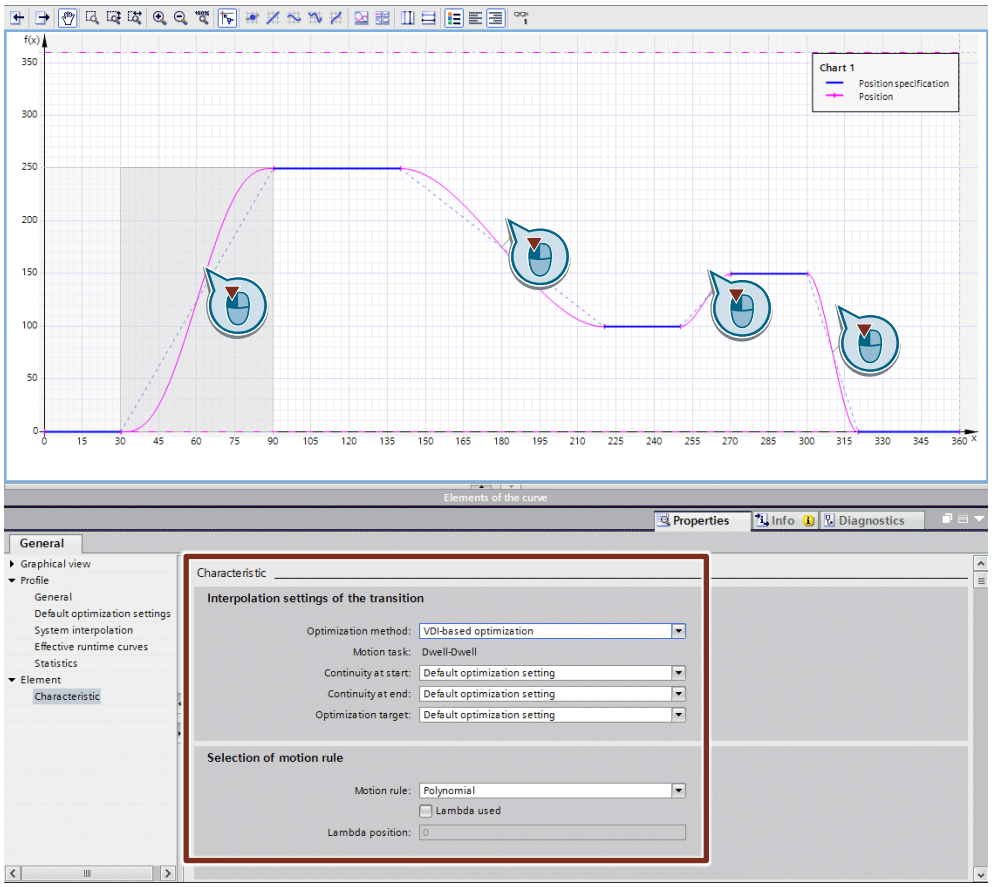
Warning symbols on the cam's automatically added elements indicate a violation of the general optimization settings in Default optimization settings.

In the last step, the user clicks the motion segments automatically inserted by the Cam Editor and makes the interpolation settings for these motion segments such that the Cam Editor no longer displays warning messages or the discontinuities in the appropriate points are accepted.

For a first approximation for creating the cam, it is normally sufficient to change the optimization method setting to VDI-based optimization.

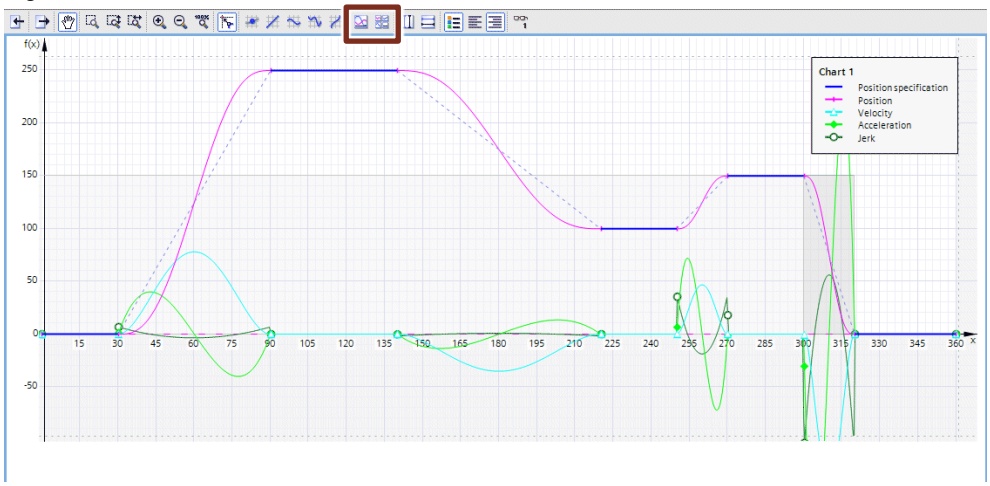
Of course, other settings specified in greater detail for the cam element's interpolation settings can be made in the Properties dialog of each motion segment. These settings allow the user to further adjust and optimize the cam. If necessary, additional cam elements, as shown in the first step, can be added to the cam to adjust the cam to the desired motional sequence.

Figure 4-15 Changing the interpolation settings of the intermediate segments



Finally, the two buttons at the top end of the cam canvas allow the user to display the derivatives of the defined cam. Using these graphs, the velocity, acceleration and jerk of the slave axis can be controlled relative to the master axis. If there are any anomalies, the defined cam can be further optimized, if necessary.

Figure 4-16



Note

To check limit violations, the dynamic properties of existing axes can be copied to the editor by a key press or specified manually in the Profile properties dialog in Effective runtime curves to display the derivatives and indicate dynamics violations.

In the Graphical view properties dialog, Charts and curves, the display of the additional dynamic curves can be further influenced or hidden.

4.4 Notes on using technology objects

4.4.1 Interpolating a cam

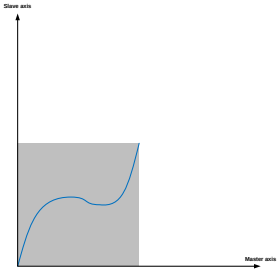
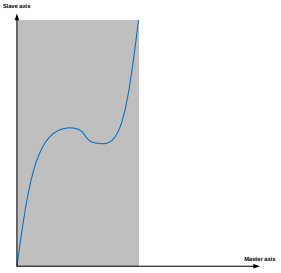
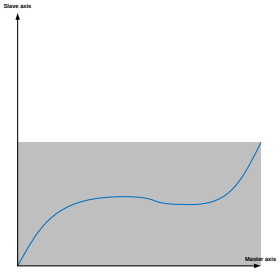
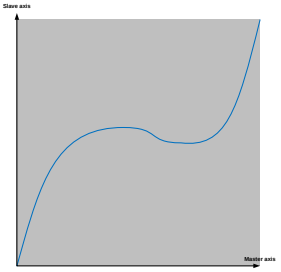
Before its first use in a user program, the Cam technology object must be interpolated using the MC_InterpolateCam technology function. In this context, it is irrelevant whether the cam was created in the user program at runtime or whether it was created and downloaded using TIA Portal.

4.4.2 Scaling a cam

When starting camming using the MasterScaling and SlaveScaling parameters, the MC_CamIn technology function allows you to scale the defined cam in the x- and y-direction.

The following table shows an example of the effects of the two parameters, MasterScaling and SlaveScaling, on the cam.

Table 4-3 Scaling a cam

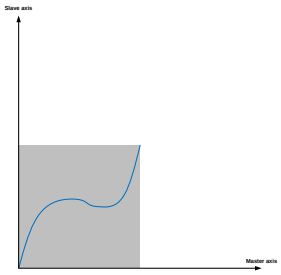
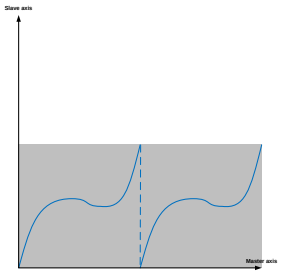
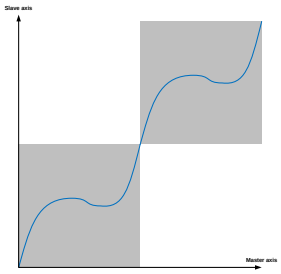
	SlaveScaling = 1.0	**SlaveScaling = 2.0**
MasterScaling = 1.0		
MasterScaling = 2.0		

Furthermore, the ApplicationMode parameter allows you to define the behavior of the slave axis outside the definition range of the cam, i.e., when the master axis exits the master axis position range defined in the cam or by the scaling:

- ApplicationMode = 0
  The cam is executed only once. If the master axis exits the definition range of the cam, camming is stopped.
  This mode can be used, for example, if the motion defined in the cam is to be executed only once but in its entirety.
- ApplicationMode = 1
  The cam is appended in the master axis direction. The master axis can therefore no longer exit the definition range of the cam.
  To avoid setpoint jumps on the slave axis, it should be ensured, in this mode, that the cam continues steadily at the start and end (slope of the curve) and that the slave axis position matches at the cam's start and end point.
- ApplicationMode = 2
  The cam is appended in the master axis direction and in the slave axis direction.
  Here, too, it has to be ensured that the cam continues steadily, especially regarding the slope of the curve at the start and end point.

The following figures illustrate the cam synchronization behavior with the different settings of the ApplicationMode parameter:

Table 4-4 Settings of the ApplicationMode parameter

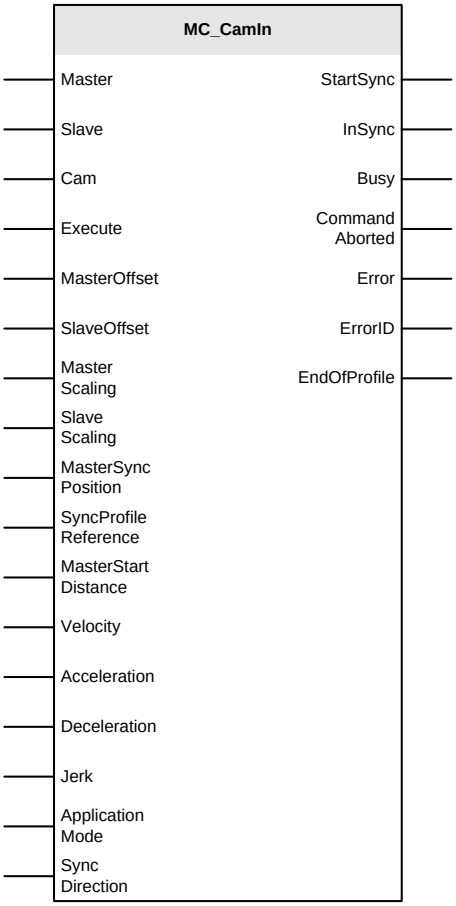
	Behavior	Comment
ApplicationMode = 0 (single)		The cam is executed only once. If the master axis exits the definition range of the cam, camming is stopped.
ApplicationMode = 1 (cyclic)		The cam is appended in the master value direction. The master axis can therefore no longer exit the definition range of the cam. Pay attention to any setpoint jumps on the slave axis!
ApplicationMode = 2 (cyclically appending)		The cam is appended in the master axis direction and in the slave axis direction. Ensure that the slope of the curve continues steadily at the start and end point!

4.4.3 Synchronization using a cam

The MC_CamIn technology function starts camming between a master axis and a slave axis with a cam as the transfer function.

The following explanations apply to master value-related synchronization of the slave axis with the master axis as can be achieved by the SyncProfileReference = 1 parameter setting on the MC_CamIn technology function.

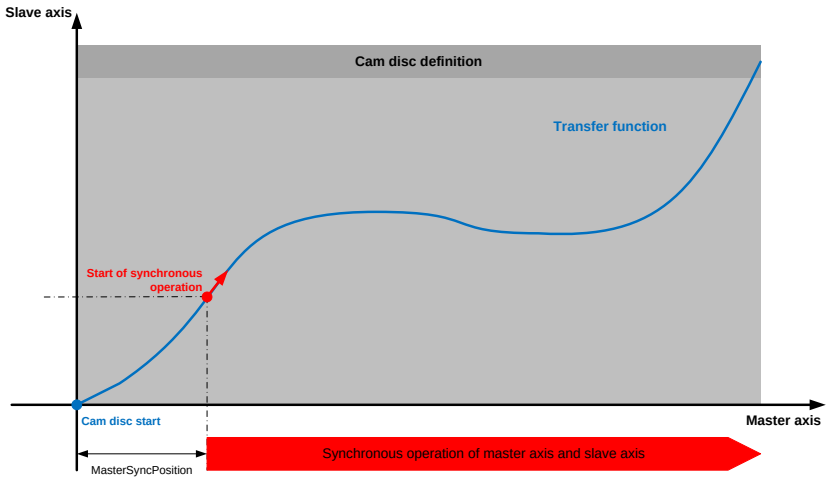
Figure 4-17 MC_CamIn technology function



The MasterSyncPosition parameter defines the position within the cam, relative to the cam start, starting from which the master axis and the slave axis of camming are to move synchronously with one another.
If the cam is to be run starting with the cam start, set the value 0.000 for the MasterSyncPosition parameter.

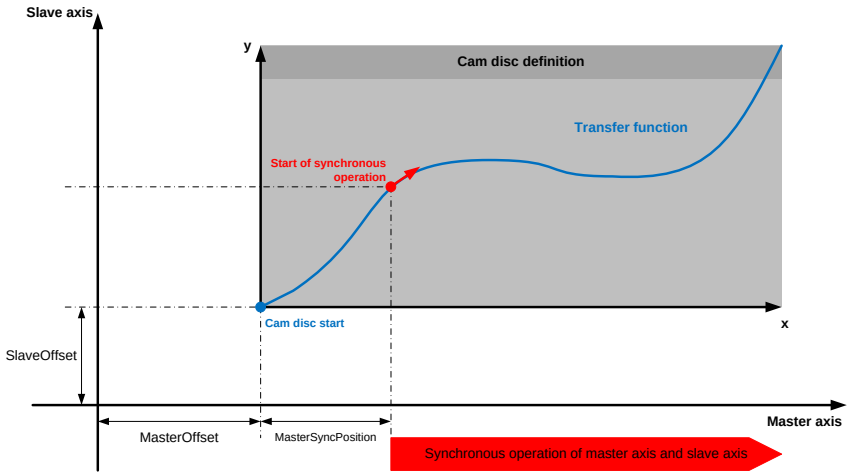
Leading synchronization of the slave axis with the master axis with the synchronous position, relative to the master axis, specified in the MasterSyncPosition parameter. When the synchronous position is reached, the two axes move synchronously with one another.

Figure 4-18 Synchronization using a cam



Using the MasterOffset parameter, you can offset the master values (x-values) of the cam in the coordinate system of the master axis. Using the SlaveOffset parameter, you can offset the slave values (y-values) of the cam in the coordinate system of the slave axis.
This moves the cam defined on an absolute basis to the position range required for your application.

Figure 4-19 Offsetting the cam



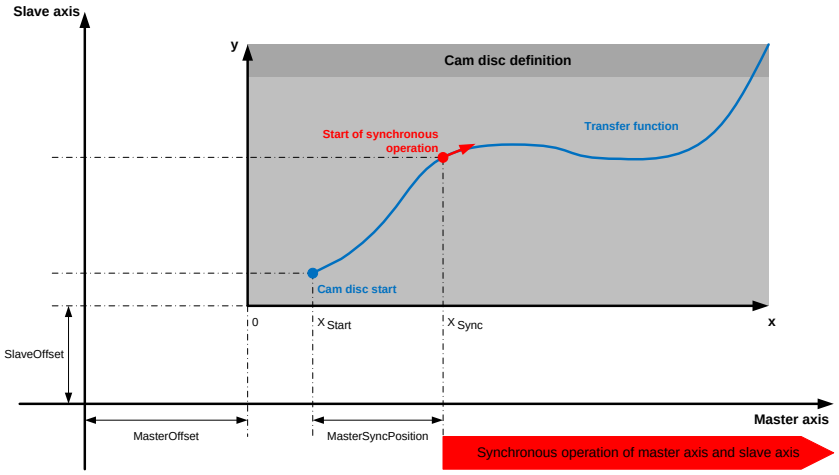
For cams that do not start at position 0.000 in the definition range of the cam, there is a special aspect regarding the calculation of the MasterSyncPosition parameter. Here, too, the value of the MasterSyncPosition parameter represents the distance between the cam start and the synchronous position on the cam. In simple terms, the MasterSyncPosition parameter can always be calculated using the following formula:

$$MasterSyncPosition = X_{Sync} - X_{Start}$$

Where:

- X_{Sync} is the master axis position in the cam definition range, starting from which master axis and slave axis are to move synchronously with one another.
- X_{Start} is the start position of the cam in the cam definition range.

Figure 4-20 Synchronizing with a cam that does not start at 0



Finally, the following section presents a brief calculation example of the above topic:

1. The following parameters are preset in the application:
   - Start of cam in definition range X_{Start}: 30.000
   - Synchronous position within cam X_{Sync}: 100.000
   - MasterOffset of cam in master axis system: 50.00
2. Using the above formula, the MasterSyncPosition parameter can then be calculated as follows:
   - MasterSyncPosition = X_{Sync} - X_{Start} = 100.000 – 30.000 = 70.000
3. This results in the following correlations on the master axis:
   - Due to the cam offset using MasterOffset, the cam start in the coordinate system of the master axis is at the following master axis position: (MasterOffset) + X_{Start} = 50.000 + 30.000 = 80.000
   - Due to the cam offset using MasterOffset, the slave axis moves synchronously with the master axis starting from the following master axis position: (MasterOffset) + X_{Sync} = 50.000 + 100.000 = 150.000

4.4.4 Synchronization of gearing at a standstill

If the two axes, master axis and slave axis, to be synchronized via gearing are at a standstill, some specifics need to be considered when synchronizing the two axes.

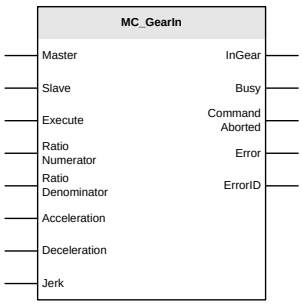
Basically, the aim is to synchronize the axes via their setpoints or actual values. The setpoints represent values of the control loop of the two axes that were calculated in the controller and, at a standstill of the two axes, therefore represent a constant value. The actual values depend on the actual axis motion of the

respective axis and represent the encoder feedback of the axis to the control loop. Therefore, the position values provided to the system by the actual values for the synchronization are a little less stable than the ones provided by setpoint coupling. This means that additional position value filter options provided by the system should be used for this coupling type.

MC_GearIn technology function

Due to the relative synchronization of master axis and slave axis using the MC_GearIn technology function, the position of the two axes is not relevant to the synchronization. Therefore, synchronous operation of the two axes starts directly when starting the MC_GearIn technology function. Via the InGear output, the block returns this state to the user program.

Figure 4-21 MC_GearIn technology function



Note The MC_GearIn technology function is the easiest way to synchronize two axes in gearing.

Note In the synchronized state, the behavior of the two technology functions for relative synchronous operation, MC_GearIn, and absolute synchronous operation, MC_GearInPos, is completely identical.

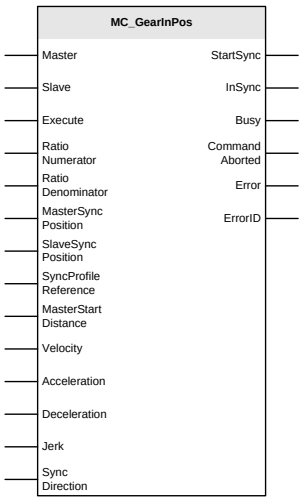
MC_GearInPos technology function

Alternatively, absolute synchronization of master axis and slave axis via the MC_GearInPos technology function can be used to synchronize the two axes. Before triggering the synchronization process, the following requirements must be met:

- Specify the current master axis position at the MasterSyncPosition parameter. If necessary, move the axis to the appropriate position using single-axis functions.
- Specify the current slave axis position at the SlaveSyncPosition parameter. If necessary, move the axis to the appropriate position using single-axis functions.

If these requirements are met and the synchronization process is started by a rising edge at the Execute input, absolute gearing between master axis and slave axis is immediately synchronized.

Figure 4-22 MC_GearInPos technology function



Note If the slave axis is not at the exact synchronous position specified at the SlaveSyncPosition parameter, the master axis must first pass the position defined at the MasterSyncPosition parameter to start the synchronization process and therefore a motion of the slave axis.

Note In the synchronized state, the behavior of the two technology functions for relative synchronous operation, MC_GearIn, and absolute synchronous operation, MC_GearInPos, is completely identical.

4.4.5 Synchronization of camming at a standstill

If the two axes, master axis and slave axis, to be synchronized via camming are at a standstill, some specifics need to be considered when synchronizing the two axes.

If synchronous operation is to be active immediately, the following requirements must be met before triggering the synchronization process via a cam:

- The master axis is at a standstill. If necessary, use single-axis commands to move the master axis to the desired position before triggering the synchronization process. The master axis position must be in the cam definition range.
- With the aid of the MC_GetCamFollowingValue technology function, the cam position of the slave axis can be determined based on the current master axis position.
- Then use single-axis commands to move the slave axis to the determined position. The slave axis, too, must then be at a standstill at this position.

- Relative to the current positions of the master axis and the slave axis at a standstill, the parameters of the MC_CamIn technology function now have to be calculated as follows:
  - MasterOffset = (current position of master axis) – (cam position of master axis)
  - SlaveOffset = (current position of slave axis) – (determined cam position of slave axis)
  - MasterSyncPosition = (current position of master axis) – (MasterOffset) – (start position of cam)
- A rising edge at the Execute input of the MC_CamIn technology function starts the cam synchronization of the two axes and immediately synchronizes the master axis with the slave axis.

Note When performing actual value coupling of camming, a system-internal hysteresis is also available here. In addition, you can parameterize a position filter, a velocity filter and a hysteresis for actual value extrapolation.

4.4.6 Changing a cam on-the-fly

If you want to change the active cam in an existing camming process, this can be achieved by once again calling the MC_CamIn technology function.

The load position of the new cam is defined at the MC_CamIn technology function using the MasterSyncPosition parameter. The transition between the two cams should be as continuous as possible. In this case, the synchronization parameters of the technology function should be set as follows:

- Synchronization over a master value distance:
  The master value distance set at the MasterStartDistance parameter should be as short as possible to achieve a direct transition from one cam to the next.
- Synchronization based on dynamic response parameters:
  The selected dynamic response parameters, Velocity, Acceleration, Deceleration and Jerk, should be as high as possible to complete the synchronization process with the new cam as quickly as possible.

4.4.7 Canceling a synchronous operation

An independent technology function for canceling a synchronous operation function is no longer available. Therefore, an active synchronous operation has to be canceled in a different manner.

If master axis and slave axis are in synchronous operation, i.e. both axes have already been synchronized, the following options are available for creating the active synchronous operation:

- Disable the slave axis using the MC_Power technology function.
- Start a motion job on the slave axis.
  For example, if the synchronous operation is to be canceled at a standstill without a motion of the slave axis, a motion job with the current slave axis position as the target position or an MC_Halt command can be started.

If the technology function to synchronize a slave axis with a master axis has already been issued while the two axes are not yet in synchronous operation, this synchronization job can only be canceled by another synchronous operation command, for example by transmitting a relative gearing command to a stopped master axis and then sending an MC_Halt command to the slave axis in the synchronized state.

Note Influencing the master axis does normally not affect active synchronous coupling to the slave axis.

Note The MC_SynchronizedMotionSimulation technology function allows you to maintain an active synchronous operation between a master axis and a slave axis even beyond disabling or moving the slave axis. The synchronous operation remains active for the controller as long as the MC_SynchronizedMotionSimulation technology function remains enabled.

To avoid a jump of the slave axis when disabling the MC_SynchronizedMotionSimulation technology function, the slave axis, at this time, should be at the same position it was when the MC_SynchronizedMotionSimulation technology function was started.

4.5 Programming functions for multiple axes

In the SIMATIC S7-31xT, the axes were represented by the number of the technology data blocks. It was possible to assign axes to technology functions simply by transferring the integer value of the technology data block number. However, access to technology object data such as the axis position was possible only by an offset in the technology data block.

In the SIMATIC S7-1500(T), by contrast, the technology objects are addressed symbolically. Symbolic programming considerably facilitates program generation when accessing technology object data and thus avoids programming errors by better readability of the program. Using technology functions generates an unambiguous assignment to the technology object by the symbolic name of the object.

However, when using large quantity frameworks on technology objects, this can easily involve considerable effort when generating the program. To avoid this problem, the following two TIA Portal functionalities can be used:

- Move or transfer the instance data of a function block
- Arrange the technology objects in an array

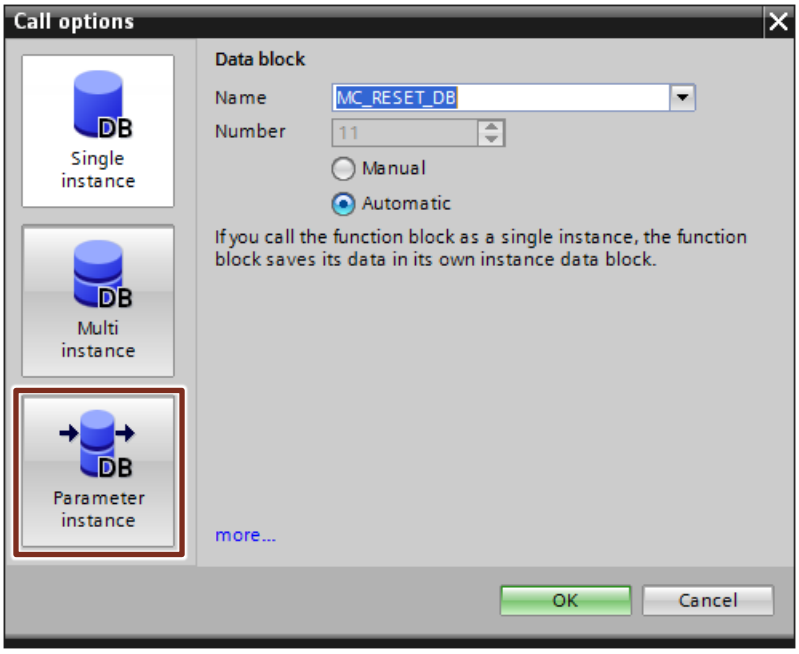
Combining these two functionalities can significantly reduce the programming effort for large quantity frameworks on technology objects.

4.5.1 Moving the instance data of a function block

If another function block or a technology function is called in a function block, the instance data of this function can be defined as follows:

- As an independent instance data block of the function in the user program
- As a multi-instance, i.e., as an instance data area in the instance data block of the calling function block
- As a parameter instance where the instance data of the function is moved to the higher-level function block via the interface of the calling function block.

Figure 4-23 Instance mapping when calling a function block



The advantage of moving the data to the higher-level instance data block via the parameter instance is that an array of this instance data can be created in this block. Due to this, many calls of a function block or technology function can also be made inside a program loop using the array index.

Note It is not possible to define the instance data of a function block or technology function in a global data block. The instance data can only be stored in an instance data block.

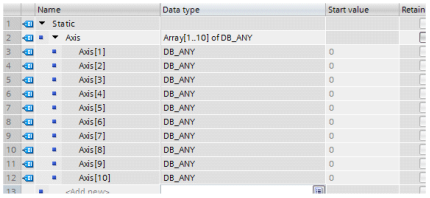
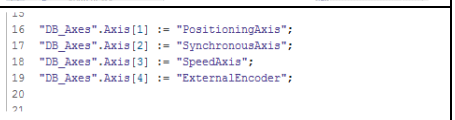
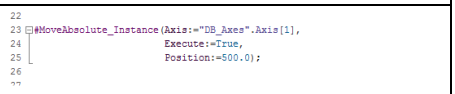
4.5.2 Arranging the technology objects in an array

In order to also access the technology objects in the project via the array index, they, too, must be arranged in an array. The DB_ANY data type is suitable for this purpose.

All types of data blocks can be stored in an array of the DB_ANY data type. This means the technology data blocks of the technology objects of a project can be stored as well. The unambiguous assignment of the technology objects to the array

elements must be made once in the user program, e.g. during CPU startup in OB 100. This requires the following program parts in the user program:

Table 4-5 Creating an array of the technology objects of a project

No.	Function	Note
1.	Create an array of the DB_ANY data type in a data block where you want to save the technology objects of the project.	
2.	During CPU startup (e.g., in OB 100), assign the appropriate technology objects of the project to the individual array elements.	
3.	Now the individual technology objects can be accessed in the program via the array index. This also allows you to perform a loop operation to the technology objects in the user program.	

However, one specific aspect needs to be considered when using the DB_ANY data type: The assignment of array elements at parameters of a function block or technology function must be made only when the data types are identical. Therefore, in the above example, only the Axis[1] element of the array that actually corresponds to a positioning axis may be assigned to the Axis input of the MC_MoveAbsolute technology function that requires the TO_PositioningAxis data type at this location.

Note

If the technology functions are directly interconnected with the technology objects in TIA Portal, for example, the layered concept of the axes described in Chapter 4.1.1 takes effect. This allows you to interconnect all equivalent or higher-level technology objects of the required data type at an input of a technology function.

If the DB_ANY data type is used for interconnection, the technology object contained in the interconnected parameter must exactly match the required data type of the technology function.

If using the DB_ANY data type requires type conversion, the call of the technology function must be encapsulated in an independent function block (FB) or a separate function (FC) for each data type to be converted that contains an input parameter for the technology object with the appropriate data type.

4.5.3 Example: Acknowledging alarms on many axes in a loop

The aim of the following example is to show acknowledgment of alarms on ten axes with the aid of a program loop.

First, this requires that the ten axes be provided in an array so that the individual axes can be accessed via the array index.
In practice, it is recommended to provide the array based on a PLC data type that contains the following information:

- Axis [DB_ANY]:
  This is where the technology object is later assigned.
- Type [Int]:
  By this type ID, the array indicates the specific axis type. The axis types used in the project can, for example, be defined throughout the project via global constants with meaningful names.
- Name [String]:
  This is where you can store a meaningful axis identifier, for example, for display on the HMI.

Figure 4-24 PLC data type for the array definition

UDT_AxisData

	Name	Data type	Default value	Accessible f...	Writa...	Visible in ...	Setpoint
1	Axis	DB_ANY	0	☑	☑	☑	☐
2	Type	Int	0	☑	☑	☑	☐
3	Name	String	''	☑	☑	☑	☐

Figure 4-25 Array definition for the technology objects

DB_Axes

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...
1	Static			☐	☐	☐	☐
2	Axis	Array[1..10] of "UDT_AxisData"		☐	☑	☑	☑
3	Axis[1]	"UDT_AxisData"		☐	☑	☑	☑
4	Axis	DB_ANY	0	☐	☑	☑	☑
5	Type	Int	0	☐	☑	☑	☑
6	Name	String	''	☐	☑	☑	☑
7	Axis[2]	"UDT_AxisData"		☐	☑	☑	☑
8	Axis[3]	"UDT_AxisData"		☐	☑	☑	☑
9	Axis[4]	"UDT_AxisData"		☐	☑	☑	☑
10	Axis[5]	"UDT_AxisData"		☐	☑	☑	☑
11	Axis[6]	"UDT_AxisData"		☐	☑	☑	☑
12	Axis[7]	"UDT_AxisData"		☐	☑	☑	☑
13	Axis[8]	"UDT_AxisData"		☐	☑	☑	☑
14	Axis[9]	"UDT_AxisData"		☐	☑	☑	☑
15	Axis[10]	"UDT_AxisData"		☐	☑	☑	☑

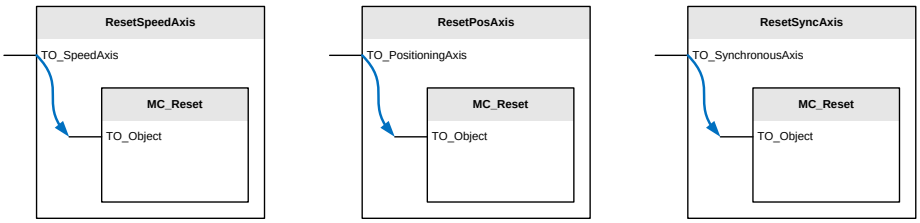
In the next step, the appropriate data of the technology objects must be unambiguously assigned to the array during CPU startup, i.e., in OB 100 of the user program.

Figure 4-26 Assigning the technology objects in OB 100 (using the example of Axis 1)

```
1  //Definition of Axis 1
2  "DB_Axes".Axis[1].Axis := "PositioningAxis";  //Select Technology object
3  "DB_Axes".Axis[1].Type := 2;                  //1: SpeedAxis, 2: PositioningAxis, 3: SynchronousAxis
4  "DB_Axes".Axis[1].Name := 'Axis 1';           //HMI name
5
6  //Definition of Axis 2
```

Now a function block or function that performs the type conversion must be created for each axis type for converting the individual axis types stored in the DB_ANY data type.

Figure 4-27 Type conversion of the individual axis types



Now all ten axes of the user program can be acknowledged inside a FOR loop by calling the appropriate type conversion functions that contain the MC_Reset technology function. For the type conversion functions, an array for the moved instances of the MC_Reset type was additionally created in the static data of the calling function block.

Figure 4-28 Acknowledging all axes in a loop

```
28
29 FOR #index := 1 TO 10 DO
30
31   CASE ("DB_Axes".Axis[#index].Type) OF
32     1:  //Axis type: SpeedAxis
33         "ResetSpeedAxis"(Axis:="DB_Axes".Axis[#index].Axis,
34                          Execute:=TRUE,
35                          MC_RESET_Instance:=#ResetInstance[#index]);
36
37     2:  //Axis type: PositioningAxis
38         "ResetPosAxis"(Axis := "DB_Axes".Axis[#index].Axis,
39                        Execute := TRUE,
40                        MC_RESET_Instance := #ResetInstance[#index]);
41
42     3:  //Axis type: SynchronousAxis
43         "ResetSyncAxis"(Axis := "DB_Axes".Axis[#index].Axis,
44                         Execute := TRUE,
45                         MC_RESET_Instance := #ResetInstance[#index]);
46   END_CASE;
47
48 END_FOR;
49
```

4.5.4 Example: Accessing axis data from the HMI

The aim of the second example is to show, for the HMI display, access to selected axis data such as position, velocity and acceleration via the axis index.

The array with the axis data of ten axes presented in the previous example is also the basis of this example. In order to access the specific data of an axis, a function (FC) to which the appropriate axis, of the same data type, from the array is transferred as a DB_ANY data type that accesses the axis data must be created for each axis type.

Figure 4-29 Block for a speed axis (without position information)

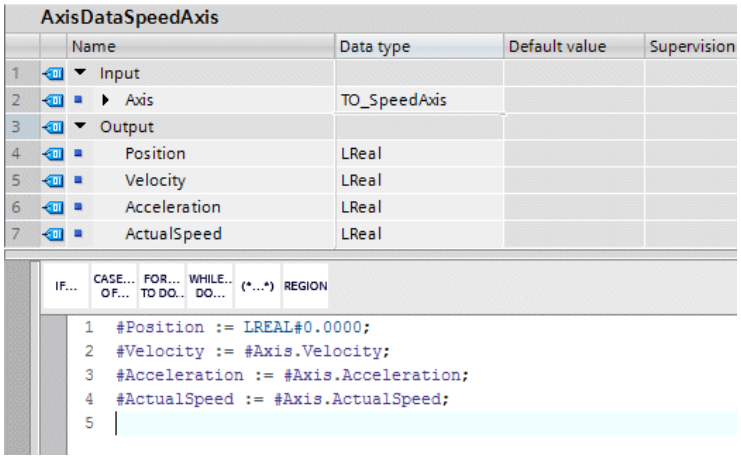


Figure 4-30 Block for a positioning axis

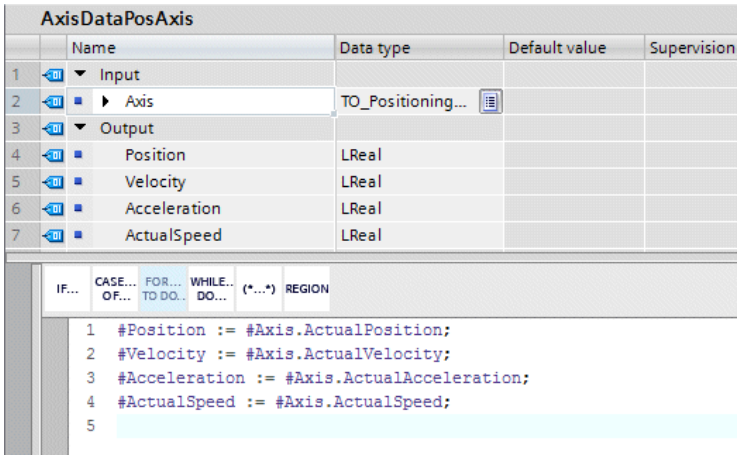
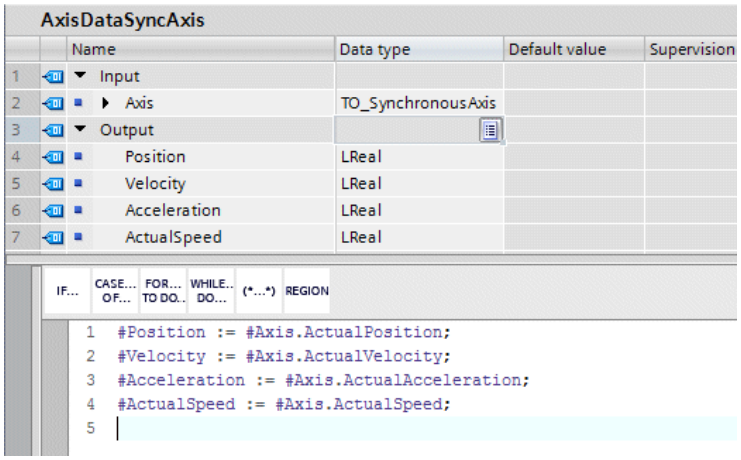


Figure 4-31 Block for a synchronous axis



Then the axis data can be read with the aid of a CASE statement that assigns the individual axis types.

Figure 4-32 Retrieving the axis data of all ten axes

```
28
29 FOR #index := 1 TO 10 DO
30
31   CASE ("DB_Axes".Axis[#index].Type) OF
32     1:  //Axis type: SpeedAxis
33         "AxisDataSpeedAxis"(Axis:="DB_Axes".Axis[#index].Axis,
34                             Position=>#HMI_Data[#index].Position,
35                             Velocity=>#HMI_Data[#index].Velocity,
36                             Acceleration=>#HMI_Data[#index].Acceleration,
37                             ActualSpeed=>#HMI_Data[#index].ActualSpeed);
38
39     2:  //Axis type: PositioningAxis
40         "AxisDataPosAxis"(Axis := "DB_Axes".Axis[#index].Axis,
41                           Position => #HMI_Data[#index].Position,
42                           Velocity => #HMI_Data[#index].Velocity,
43                           Acceleration => #HMI_Data[#index].Acceleration,
44                           ActualSpeed => #HMI_Data[#index].ActualSpeed);
45
46     3:  //Axis type: SynchronousAxis
47         "AxisDataSyncAxis"(Axis := "DB_Axes".Axis[#index].Axis,
48                            Position => #HMI_Data[#index].Position,
49                            Velocity => #HMI_Data[#index].Velocity,
50                            Acceleration => #HMI_Data[#index].Acceleration,
51                            ActualSpeed => #HMI_Data[#index].ActualSpeed);
52   END_CASE;
53
54 END_FOR;
55
56
```


4.6 Providing axis data in the user program

Current data of an axis or technology object is provided to the user program via the technology data blocks. This is where, for example, the current position and the setpoint position of an axis or its current velocity can be retrieved for further use in the user program or display on the HMI.

However, access to the technology data blocks does not act in the same way as normal access to global data blocks from the user program; in this case, when requested by the CPU's operating system, the data of the technology data block must be read from the technology object and provided via the technology data block. And this process requires some time.

For performance reasons, it is not recommended to read the required parameters in the user program in the appropriate places directly from the technology data blocks of the axes; rather, it is recommended to copy these parameters, e.g., at the beginning of an organization block as a collection of parameters from the technology data blocks of the axes or technology objects to a global data block and then use only the data from the global data block within the user program, for example, for display on the HMI.

Note If the user program requires data that is very up-to-date, this data cannot be determined using the above procedure as the data, for example due an organization block interrupt, may already be obsolete. Then this data must be read in the user program directly from the technology data block.

4.7 Organization blocks for motion control

4.7.1 Basic information about motion control OBs

When you create a technology object, organization blocks are automatically created for processing the technology objects. The motion control functionality of the technology objects generates its own execution level and is called based on the motion control application cycle.

The following organization blocks are created:

- MC-Servo [OB91]
  Calculation of the position controller of the axes.
- MC-Interpolator [OB92]
  Motion control with evaluation of motion control instructions, setpoint generation and monitoring functionality for the axes.

The organization blocks are protected (know-how protection). The program code cannot be viewed or modified. The cycle ratio of the two organization blocks to one another is always 1:1. The MC-Servo OB is always executed before the MC-Interpolator OB.

Application cycle

The application cycle and the organization blocks' priority can be set to suit your requirements for control quality and system load.

In the organization block properties in "General > Cycle Time", you can set the application cycle in which MC-Servo is called as follows:

- Synchronous to the bus
  MC-Servo is called synchronously with (1:1) or at a reduced ratio to a bus system. In the properties of the selected bus system, you can set the send clock. From the "Distributed I/O" drop-down list, you can select one of the following bus systems:
  - Isochronous PROFIBUS DP
  - Isochronous PROFINET IO

  You cannot call MC-Servo synchronously with a bus system that is connected to the CPU via a communications processor/communications module (CP/CM).
- Cyclic
  MC-Servo is called cyclically with the specified application cycle.

The selected application cycle must be long enough to be able to process all technology objects in one cycle. If the processing time of the technology objects is longer than the application cycle, overflows occur that cause the CPU to stop.

Determining the OB runtime

You can check the runtime of MC-Servo and MC-Interpolator using the extended instruction "RT_INFO".

Priority of the organization blocks

If necessary, you can set the priority of the organization blocks in their properties in "General > Attributes > Priority number":

- MC-Servo
  Priority 17 to 31 (default value: 26)
- MC-Interpolator
  Priority 16 to 30 (default value: 25)

Note The priority of MC Servo must be at least one higher than the priority of MC-Interpolator.

Note When changing the priority of MC-Servo, make sure that the MC-Servo organization block is not interrupted by other organization blocks.

4.7.2 Clock reduction (CPU firmware V1.5 and higher)

You can reduce the application cycle of MC-Servo relative to the send clock of an isochronous PROFINET IO system. As the factor, you can set an integer multiple of the send clock. Values up to 14 times the send clock (max. 32 ms) are possible for the application cycle.

If you call an isochronous interrupt OB and MC-Servo synchronously with the same PROFINET IO system, set the same application cycle for both organization blocks.

4.7.3 Processing the motion control functionality

When processing the motion control functionality, the MC-Servo and MC-Interpolator organization blocks are called and processed in each application cycle. The remaining cycle time is available for processing your user program.

For error-free program execution, the following rules apply:

- In each application cycle, MC-Servo must be started and processed completely.
  Overflows of MC-Servo, i.e. processing the organization block cannot be completed within an application cycle, set the CPU to Stop mode.
- In each application cycle, the associated MC-Interpolator must be started. Completing MC-Interpolator may even extend into the next application cycle, which represents an overflow of the MC-Interpolator OB. The CPU tolerates a maximum of three successive overflows before it goes to Stop mode.
- When using the DSC (Dynamic Servo Control) functionality, the runtime of MC-Servo, including MC-PreServo and MC-PostServo, must be significantly shorter than the PROFINET cycle so that data output to the drive can always take place in the next PROFINET cycle.

4.7.4 Using the MC-PreServo and MC-PostServo OBs

When using the motion control functions of the SIMATIC S7-1500, the following organization blocks are automatically created:

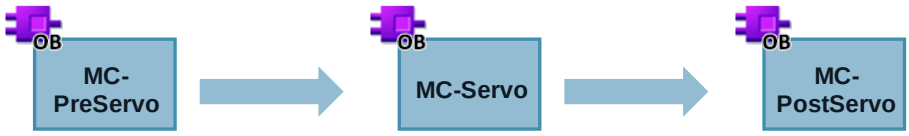
- MC-Servo [OB91]
  Calculation of the position controller of the axes.
- MC-Interpolator [OB92]
  Motion control with evaluation of motion control instructions, setpoint generation and monitoring functionality for the axes.

If you have to make changes to the drive's position control loop in the MC-Servo OB, for example when using an alternative or non-standardized encoder system or

when using a special drive such as a hydraulic axis, these changes cannot be included in the protected MC-Servo OB. The following two editable organization blocks are available for these purposes:

- MC-PreServo [OB67]
  Execution in the task system between the input process image update and before the calculation of the position controller of the axes by the MC-Servo OB.
  The input values of the position controller can be changed in this block.
- MC-PostServo [OB95]
  Execution in the task system after the calculation of the position controller of the axes by the MC-Servo OB and before process image output to the I/O.
  The output values of the position controller can be changed in this block.

Figure 4-33 OB processing sequence in the task system for the MC-Servo OB



The input and output values of the position controller are transferred in the control loop of the axes using the PROFIdrive telegram. The data can be broken down as follows:

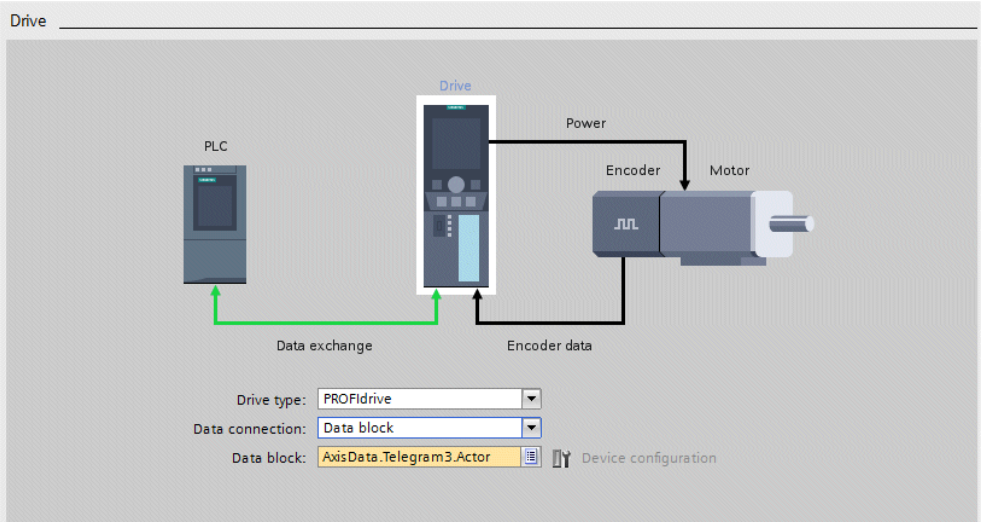
- Input values of the position controller:
  For example, the current encoder values of the drive such as the current position that are transferred to the position controller in compliance with encoder telegram 81.
- Output values of the position controller:
  For example, the speed setpoints output by the position controller that are forwarded to the drive using the PROFIdrive telegram.

Changing the input and output values of the position controller may be necessary, for example, in the following cases:

- Use of an analog value (potentiometer) as an encoder replacement.
- Control of a hydraulic axis using the motion control functionality of the SIMATIC CPU that may require linearizing the valve characteristic of the hydraulic axis with the aid of a compensation characteristic.

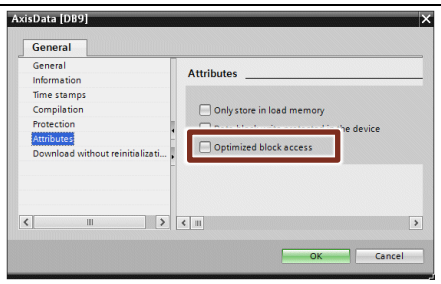
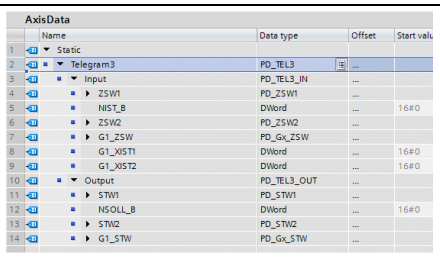
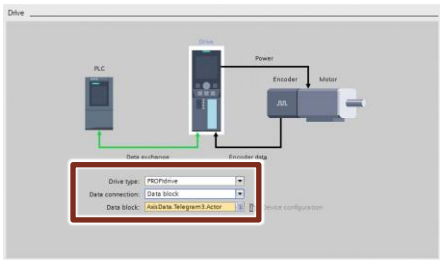
To be able to change the input and output values of the position controller, the technology object used may be directly connected to the drive I/O (I/O image) or a data block (DB). Then these data areas can be manipulated in the two organization blocks, MC-PreServo and MC-PostServo, which enables you to make the desired changes in the control loop before or after the calculation of the position controller.

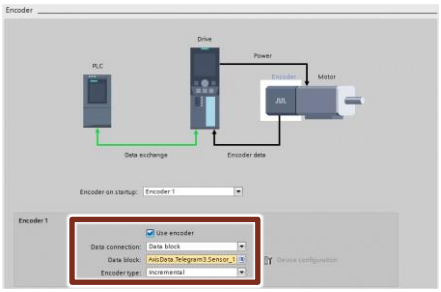
Figure 4-34 Connection of the technology object to a data block



To generate a data block usable for this purpose in the user program, proceed as follows:

Table 4-6 Creating a data block for connecting a technology object

No.	Instruction	Note
1.	In the user program, create a global data block.	
2.	In the data block properties, set the block as not optimized. Only non-optimized data blocks have unique offset addresses of the data.	
3.	Insert a data area for the input and output values of the position controller into the data block. The data area can be defined with the aid of a PLC data type, for example for telegram 3 (PD_TEL3), that has already been stored in TIA Portal. However, this data type must be entered manually; it cannot be selected from the drop-down list in the data block.	
4.	Compile the newly created data block.	
5.	Now you can connect the appropriate data areas of the technology object to the data areas of the data block in the technology object's configuration dialog.	

No.	Instruction	Note
		

Note The PLC data types stored in TIA Portal for the individual telegrams are each named PD_TELxxx and the telegram number must be entered for "xxx", for example PD_TEL105, PD_TEL81, etc.

4.8 Motion control resources

Each Technology CPU 31xT has a fixed number of usable technology objects for each object type, for example a maximum of 8 axes and a maximum of 16 output cams. Not using the maximum number of an object type does not affect another object type. For example, when using only 6 axes, not more than 16 output cams can be used in the CPU. Therefore, from the user's perspective, the usage of the CPU's memory and computing power is not always optimal.

A flexible concept for using technology objects was introduced with the SIMATIC S7-1500(T). Each CPU has a fixed number of motion control resources that can be divided among the technology objects. An exception to this are cams that represent pure data objects and therefore do not consume motion control resources.

Note Each technology object also requires computing time on the CPU. This means that each technology object extends the CPU's application cycle. Therefore, when determining the CPU type necessary for the technological requirements, keep an eye on the resulting application cycle length, including any reduction that may be set. Also consider potential effects on processes at other lower-priority execution levels of the CPU, e.g. safety.

4.8.1 Motion control resources provided

The following table lists the provided motion control resources for some selected CPUs. The provided resources depend on the CPU hardware version used and the firmware version used.

Table 4-7 Provided motion control resources for selected CPU types

CPU type	Order no.	Firmware	MC resources
S7-1511	6ES7 511-1AK01-0AB0	V 2.0	800
S7-1513	6ES7 513-1AL01-0AB0	V 2.0	800
S7-1515	6ES7 515-2AM01-0AB0	V 2.0	2400
S7-1516	6ES7 516-3AN01-0AB0	V 2.0	2400
S7-1517	6ES7 517-3AP00-0AB0	V 2.0	10240
S7-1518	6ES7 518-4AP00-0AB0	V 2.0	10240
S7-1511T	6ES7 511-1TK01-0AB0	V 2.0	800
S7-1515T	6ES7 515-2TM01-0AB0	V 2.0	2400
S7-1517T(F)	6ES7 517-3TP00-0AB0 6ES7 517-3UP00-0AB0	V 2.0	10240

Note For the exact values for the provided motion control resources of a CPU, please refer to the specific CPU's technical specifications or manual.

4.8.2 Number of cams provided

The following table lists the maximum number of cams for some selected CPUs.

Table 4-8 Provided number of cams for selected CPU types

CPU type	Order no.	Firmware	Maximum number
S7-1511T	6ES7 511-1TK01-0AB0	V 2.0	20
S7-1515T	6ES7 515-2TM01-0AB0	V 2.0	60
S7-1517T(F)	6ES7 517-3TP00-0AB0 6ES7 517-3UP00-0AB0	V 2.0	128

Note A Cam technology object does not consume motion control resources. The number of cams that can be created in a CPU is limited by the CPU's available work memory (data memory).

For detailed information, please refer to the specific CPU's technical specifications or manual.

4.8.3 Motion control resources consumed

The following table lists the motion control resources consumed by the technology objects.

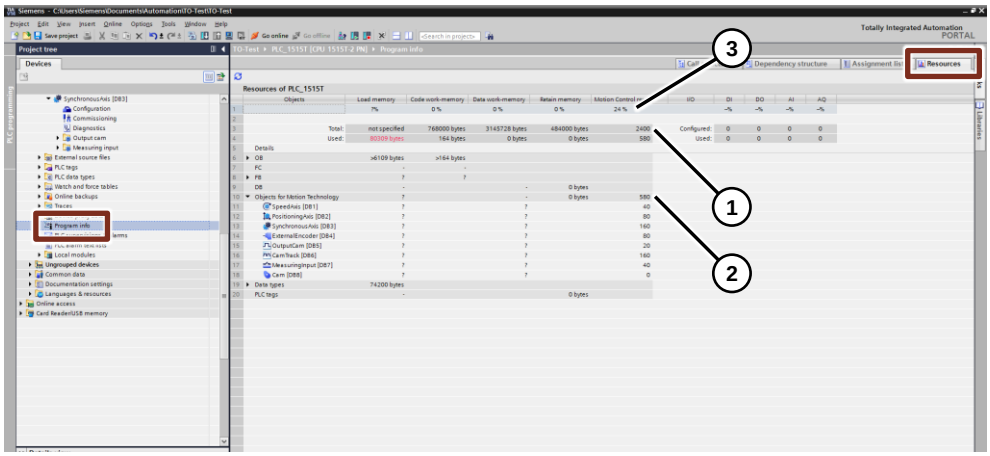
Table 4-9 Motion control resource consumption of the technology objects

Technology object	Version	MC resources	Comment
Speed Axis	V3.0	40	
Positioning Axis	V3.0	80	
Synchronous Axis	V3.0	160	
External Encoder	V3.0	80	
Measuring Input	V3.0	40	
Output Cam	V3.0	20	
Cam Track	V3.0	160	

4.8.4 Checking the motion control resource consumption

In TIA Portal, selecting Program info > Resources allows you to easily check the actual motion control resource consumption of a user program. The Resources tab compares the motion control resources provided by the CPU (1) with the motion control resources consumed by the technology objects used (2) and displays the resource consumption as a percentage (3).

Figure 4-35 Motion control resource consumption display



If, when developing the user program, it turns out that the selected CPU type does not provide a sufficient number of required motion control resources, the CPU's context menu allows you to quickly and easily replace the CPU with a suitable CPU type in the hardware configuration.

Figure 4-36 Change CPU – context menu for calling the function

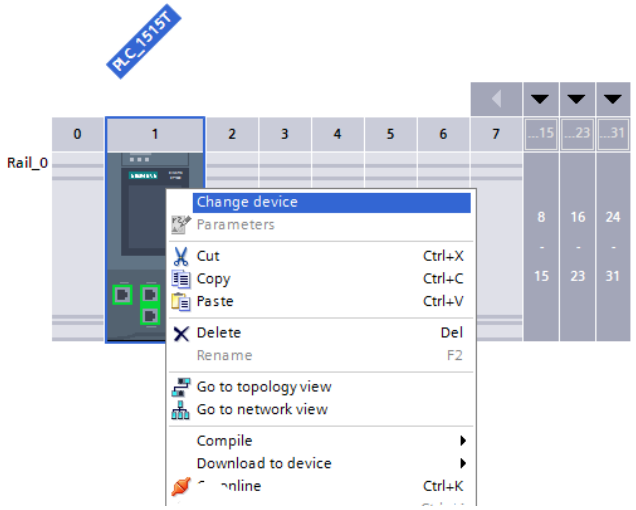
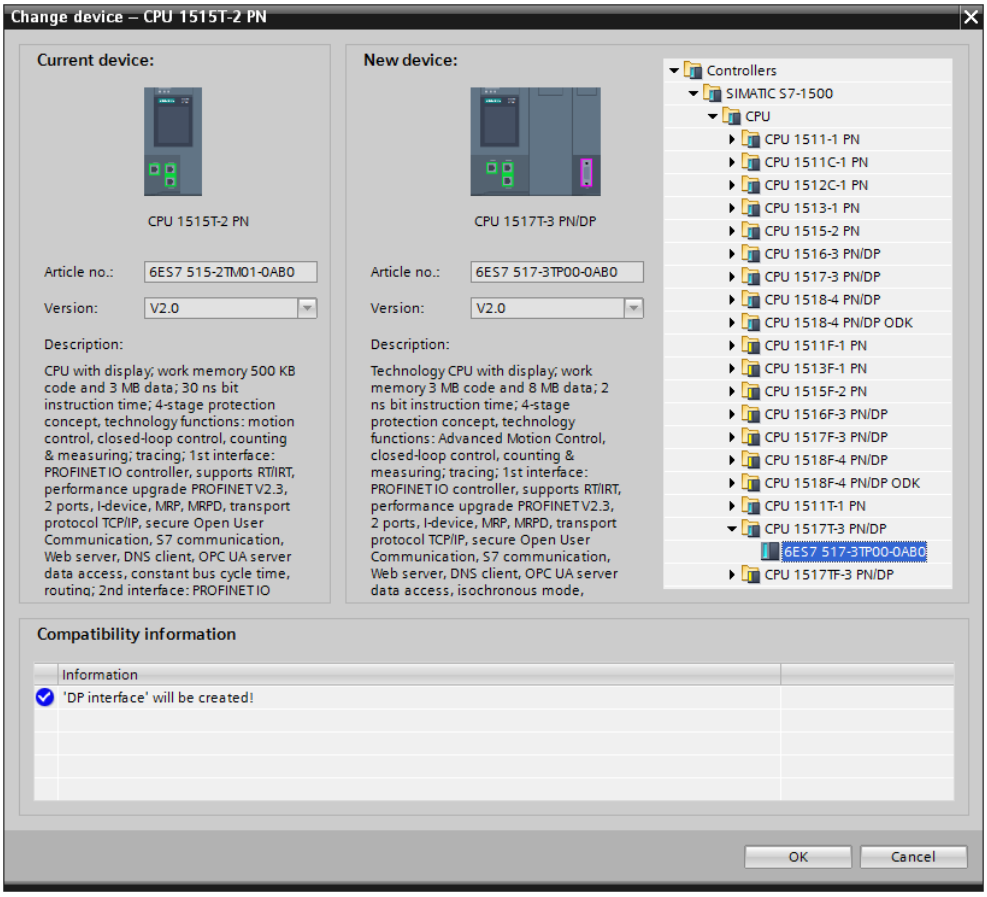


Figure 4-37 Change CPU – selection dialog of the new CPU



5 Appendix

5.1 Information about the technologies

The following table once again shows all technologies in the different CPUs.

Table 5-1 Technologies compared

Technology	S7-31xT	S7-1500	S7-1500T
Single-axis motion functions	Yes	Yes	Yes
Gearing	Yes	Limited	Yes
Camming	Yes	No	Yes
Output cams and cam tracks	Yes	Yes	Yes
Measuring inputs	Yes	Yes	Yes
External encoders	Yes	Yes	Yes
Path objects	Yes	No	No
Pressure control	Yes	No	No
Hydraulic axes	Yes	Specific to application	Specific to application
Stepper drives	Yes (IM174)	S7-1500C / TM PTO4	TM PTO4

The following chapters provide additional information about the above technologies and discuss whether and how they can be implemented in the CPUs of the S7-1500 and S7-1500T series.

5.1.1 Single-axis motion functions

All CPU types of the S7-1500 and S7-1500T series feature single-axis motion functions that can be used directly without additional software.

5.1.2 Gearing and camming

The relative gearing functionality is integrated in all CPUs of the S7-1500 series and can be used directly. If the following functions are required, a CPU of the S7-1500T series must be used:

- Absolute gearing to create a defined offset between master axis and slave axis already during the synchronization process.
- Camming.
- Actual value coupling between master axis and slave axis or use of external encoders as a master axis.
- Use of MC_Phasing for phase shift between master axis and slave axis.

5.1.3 Output cams and cam tracks, measuring inputs and external encoders

Single output cams and cam tracks for the use of multiple output cams in one track, measuring inputs and external encoders are available in each CPU of the S7-1500 and S7-1500T series.

5.1.4 Path objects

Path objects and machine kinematics and motion commands for trajectories are currently not yet available in TIA Portal and in the CPUs of the S7-1500 and S7-1500T series.

5.1.5 Pressure control

Pressure control is currently not available via the technology functions of the CPUs of the S7-1500 and S7-1500T series.

Specific to an application, pressure control can nevertheless be implemented using the PID and PID Compact controllers and suitable input and output modules.

Alternatively, pressure control can be implemented by controlling a hydraulic valve with the aid of the TM Pulse technology module and a suitable user program.

5.1.6 Hydraulic axes

Control of hydraulic axes using the technology functions of the CPUs of the S7-1500 and S7-1500T series is currently not available.

Specific to an application, control of hydraulic axes and consideration of a hydraulics characteristic can be implemented in the MC-PostServo organization block in the CPUs of the S7-1500 and S7-1500T series.

5.2 PLCopen blocks compared

This chapter compares the principle of operation of the individual PLCopen blocks of the technology functions. Its aim is to provide information and support for migrating a user program from the CPU31xT to the S7-1500 / S7-1500T.

If technology functions are not available on the S7-1500 / S7-1500T, the appropriate chapter shows, where possible, an application solution to transferring the functionality.

Note

At present, all technology functions that are not listed in the following chapters cannot be migrated to the S7-1500 / S7-1500T.

If your user program uses one of these functions, a migration is currently not possible or you have to find an application solution to implement this functionality.

Basic differences between the PLCopen blocks

The individual PLCopen blocks of the technology functions of the CPU31xT and the S7-1500 / S7-1500T basically differ in the following aspects that require special attention when migrating the technology functions:

- Different data types at the Axis input parameter
  In the CPU31xT, an integer value was available for axis assignment that was checked during CPU runtime and could therefore be changed during CPU runtime. In the S7-1500 / S7-1500T, axis assignment requires that the symbolic name of a technology object be entered, whose validity is already checked during programming by data type mapping.
  For self-created user function blocks that are to encapsulate PLCopen blocks, an Axis input can be created with the TO_Axis, TO_SpeedAxis, TO_PositioningAxis or TO_SynchronousAxis data types or a parameter of the DB_ANY type can be assigned to the Axis parameter of the included PLCopen function blocks that then refers to the technology object or the technology data block of the technology object of the correct axis type (example in Chapter 4.1).
- Different data types for floating-point values
  In the CPU31xT, max. the REAL data type was available for floating-point values. In the S7-1500 / S7-1500T, the precision of floating-point values was increased by adding the LREAL data type that is now used in the technology functions instead of REAL. After the migration, this may result in messages in TIA Portal regarding the precision of the transferred values at the parameters of the PLCopen blocks.
- Differences in the available modes
  Some technology functions of the CPU31xT have an extended scope of modes and a different assignment of the individual PLCopen block modes than the functions of the S7-1500 / S7-1500T; this is due to the currently available range of functions or the additional technologies in the CPU31xT such as hydraulics. Therefore, after the migration, carefully check mode control of the technology functions from the user program to avoid an unexpected response of the machine or axes and any resulting potential risks for humans and the machine.
- Different error concepts
  The CPU31xT and the S7-1500 / S7-1500T have different error concepts. Consequently, the error codes at the ErrorID output of the technology functions do not match. Therefore, error handling in the user program must be specifically checked once again after the migration.
- DoneFlag input parameter
  In the CPU31xT, the DoneFlag input parameter was used to quickly display the completion of a job issued via a PLCopen function block in the control section of the CPU. Due to the architecture of the S7-1500 / S7-1500T, this function is now no longer required. For this reason, the DoneFlag input parameter is no longer available at the PLCopen function blocks of the S7-1500 / S7-1500T.

NOTICE	**After a migration, check the user program specifically in terms of the mode specified at the PLCopen blocks of the technology functions to avoid an unexpected response of the machine or drives.**

Meaning of the icons

The icons at the beginning of a chapter indicate the availability of the respective technology function in the CPU types:

Table 5-2 Meaning of the icons

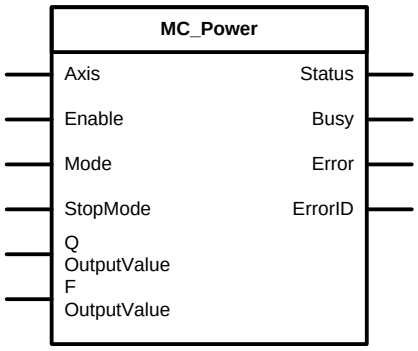
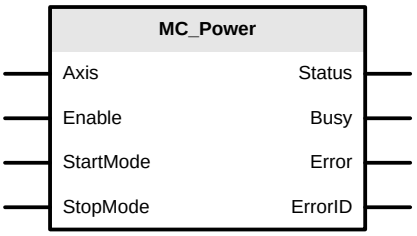
Icon	CPU type
	Technology CPU 31xT The technology functions for motion control marked with this icon are standard features in all controllers of the S7-300T series.
	SIMATIC S7-1500 The technology functions for motion control marked with this icon are standard features in all controllers of the S7-1500 series.
	SIMATIC S7-1500 The technology functions for motion control marked with this icon are not available as an independent technology function in the S7-1500 technology controller. However, they can be implemented in the user program by simple program functions such as direct access to a data block.
	SIMATIC S7-1500T The technology functions for motion control marked with this icon are only available in the S7-1500T technology controller of the S7-1500 series.
	SIMATIC S7-1500T The technology functions for motion control marked with this icon are not available as an independent technology function in the S7-1500T technology controller. However, they can be implemented in the user program by simple program functions such as direct access to a data block.

5.2.1 MC_Power

This technology function is available in ...

This technology function enables or disables a technology object.

Table 5-3 MC_Power

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object	TO_Axis	Axis
Enable	BOOL	Enable or disable technology object	BOOL	Enable
Mode	INT	Axis mode Enable positioning axis / synchronous axis not on a position-controlled basis / on a position-controlled basis	DINT	StartMode
StopMode	INT	Stop mode when disabling the technology object	INT	StopMode
Q OutputValue	REAL	Control signal replacement value for Q valve		
F OutputValue	REAL	Control signal replacement value for P valve / F output		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Status	BOOL	Enable status of the technology object	BOOL	Status
Busy	BOOL	The job is being processed	BOOL	Busy
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

The following parameters involve functional differences or differences in the parameterization:

Table 5-4

Mode / StartMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	**Speed axis** A speed axis with encoder is enabled on a closed-loop speed-controlled basis. A speed axis without encoder is enabled on an open-loop speed-controlled basis. **Positioning axis and synchronous axis** The axis is enabled on a position-controlled basis. **Default setting**	Do not enable positioning axis / synchronous axis on a position-controlled basis
1	**Speed axis** A speed axis with encoder is enabled on a closed-loop speed-controlled basis. A speed axis without encoder is enabled on an open-loop speed-controlled basis. **Positioning axis and synchronous axis** The axis is enabled on a closed-loop speed-controlled basis. The enable is saved (see mode = 0).	Enable positioning axis / synchronous axis on a position-controlled basis **Default setting** The parameter is initially effective when enabling the positioning axis (enable changes from FALSE to TRUE) and during the enable after successful acknowledgment of an alarm that caused the axis disable. When using a speed axis or an external encoder, this parameter will be ignored.
2	Following mode with enabled power unit	[Not available]
3	Following mode without enabled power unit	[Not available]
4	Simulation mode	[Not available]
5	Enable axis on a position controlled basis (positioning axes and synchronous axes only). The enable is saved (see mode = 0).	[Not available]
6	Enable axis on a force/pressure-controlled basis. Force/pressure control is enabled with pressure setpoint = current pressure actual value. If force/pressure control is terminated by a motion job, the axis goes to position-controlled mode.	[Not available]

Table 5-5

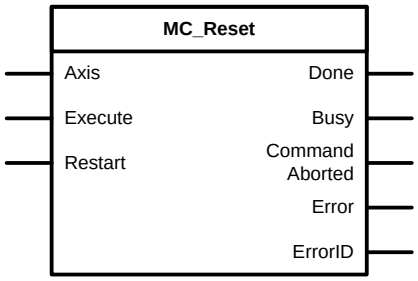
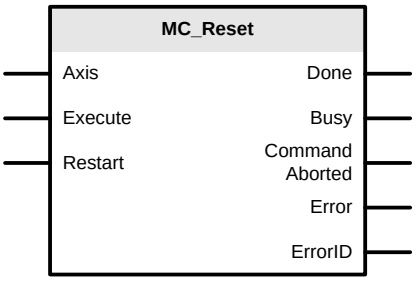
StopMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	DefaultStop **Default setting**	Emergency stop **Default setting**
1	FastStop	Immediate stop
2	TimeStop	Stop with maximum dynamic response values
3	OFF2Stop	[Not available]
4	GearStop	[Not available]
5	Ramp stop (real electrical axes only)	[Not available]
6	Fast stop (real electrical axes only)	[Not available]

5.2.2 MC_Reset

This technology function is available in ...

With this function, you can acknowledge all technology alarms that can be acknowledged in the user program.

Table 5-6 MC_Reset

CPU31xT	S7-1500(T)
	

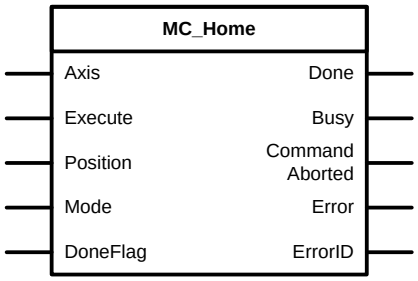
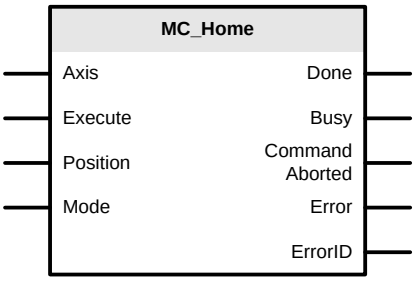
Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object	TO_Object	Axis
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Restart	BOOL	Reinitialization The technology object is reinitialized with the configured start values.	BOOL	Restart
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

5.2.3 MC_Home

This technology function is available in ...

With this technology function, you can create the relationship between the current position of the technology object and the mechanical position.

Table 5-7 MC_Home

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object	TO_Axis	Axis
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Position	REAL	Homing position The parameter is used based on the mode selection.	LREAL	Position
Mode	INT	Mode selection	INT	Mode
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

The following parameters involve functional differences or differences in the parameterization:

Table 5-8

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	**Active homing** Home position approach according to the axis configuration **Default setting**	**Direct homing (absolute)** The current position of the technology object is set to the value of the "Position" parameter. **Default setting**
1	**Active homing** Home position approach according to the axis configuration. The value of the Position input parameter is used as the home position coordinate.	**Direct homing (relative)** The current position of the technology object is shifted by the value of the "Position" parameter.
2	**Passive homing** Homing according to the configuration in S7T Config. The value of the Position input parameter is used as the home position coordinate.	**Passive homing (without reset)** Same function as "mode" 8, the difference being that enabling the function does not reset the "homed" status.
3	**Direct homing** The value of the Position input parameter is assigned to the current position.	**Active homing** The Positioning Axis / Synchronous Axis TO performs a homing motion according to the configuration. After completing the motion, the axis is at the value of the "Position" parameter.
4	**Correction of the actual position** (New actual position = current actual position – Position parameter)	[Not available]
5	**Absolute encoder adjustment** The value of the Position input parameter is assigned to the current position. The position shift due to absolute encoder adjustment is retained after a power failure. The axis must be at a standstill to allow precise adjustment. Any correction of the position by mode = 3, 4, 6 or 7 is retained after absolute encoder adjustment. For absolute encoder adjustment, the axis must be in position-controlled mode.	**Active homing** The Positioning Axis / Synchronous Axis TO performs a homing motion according to the configuration. After completing the motion, the axis is at the home position configured in "Technology object > Configuration > Extended parameters > Homing > Active homing". (<TO>.Homing.HomePosition)
6	**Setpoint position correction in the base coordinate system** Any superimposed motion is based on the corrected setpoint position. (New setpoint position = current base position - Position parameter)	**Absolute encoder adjustment (relative)** The current position is shifted by the value of the "Position" parameter. The calculated absolute value offset is stored retentively in the CPU. (<TO>.StatusSensor[n].AbsEncoder Offset)

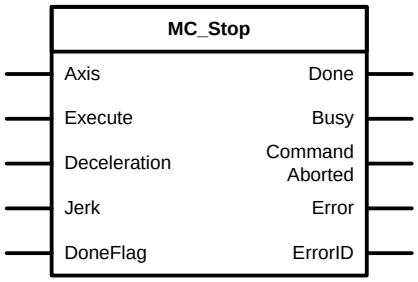
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
7	**Setpoint position correction in the superimposed coordinate system** (New setpoint position = current superimposed position - Position parameter) Please note: Position corrections (mode = 3, 4, 6 or 7) are active in addition to the absolute encoder adjustment. After a POWER OFF or restart, only the absolute encoder adjustment is active.	**Absolute encoder adjustment (absolute)** The current position is set to the value of the "Position" parameter. The calculated absolute value offset is stored retentively in the CPU. (<TO>.StatusSensor[n].AbsEncoder Offset)
8	**Canceling passive homing** A started passive homing job is canceled.	**Passive homing** When the homing mark is detected, the actual value is set to the value of the "Position" parameter.
9	[Not available]	**Canceling passive homing** A running passive homing job is canceled.
10	[Not available]	**Passive homing** When the homing mark is detected, the actual value is set to the home position configured in "Technology object > Configuration > Extended parameters > Homing > Passive homing". (<TO>.Homing.HomePosition)

5.2.4 MC_Stop

This technology function is available in ...

This technology function stops all motion processes and prevents the axes from restarting.

Table 5-9 MC_Stop

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object		
Execute	BOOL	Start technology function with rising edge		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Deceleration parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

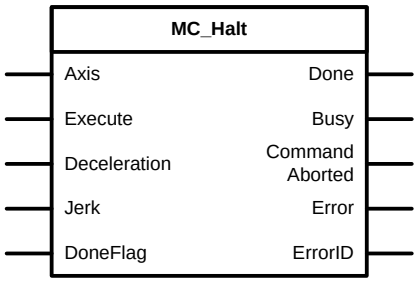
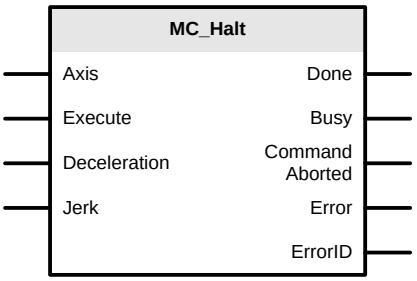
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.5 MC_Halt

This technology function is available in ...

This technology function decelerates a motion of an axis to a standstill.

Table 5-10 MC_Halt

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Axis	INT	Technology object	TO_ SpeedAxis	Axis
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
		Use jerk limit when stopping	BOOL	Abort Acceleration
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

Output parameters (OUT)

Parameter	Data type	Function	Data type	Parameter
Done	BOOL	Zero velocity reached	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

For the Deceleration parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed (exception: the axis is at a standstill)
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

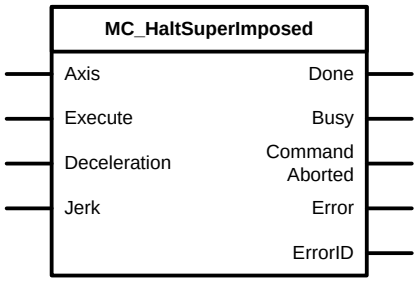
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.6 MC_HaltSuperImposed

This technology function is available in ...

This technology function decelerates a superimposed motion of an axis to a standstill.

Table 5-11 MC_HaltSuperImposed

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object		
Execute	BOOL	Start technology function with rising edge		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	Superimposed motion terminated		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Deceleration parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed (exception: the axis is at a standstill)
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

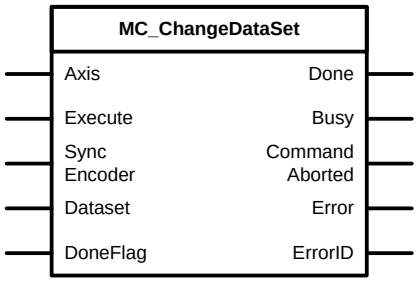
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.7 MC_ChangeDataSet

This technology function is available in ...

This technology function allows you to toggle between multiple data sets of a technology object.

Table 5-12 MC_ChangeDataSet

CPU31xT	S7-1500(T)
	

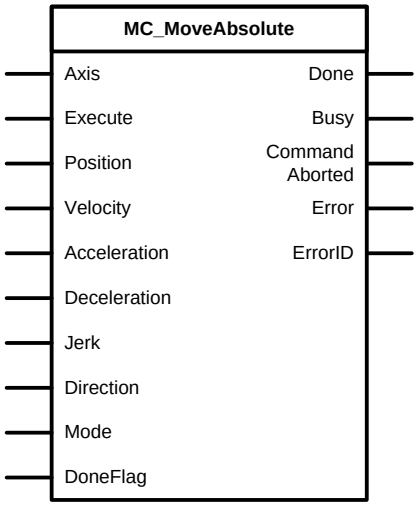
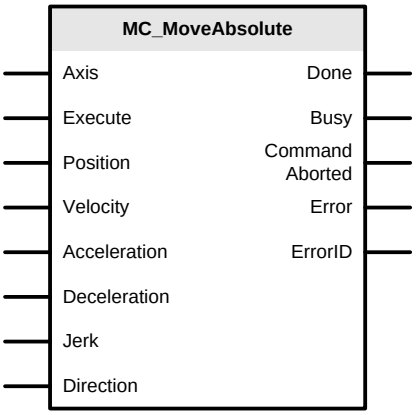
Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object		
Execute	BOOL	Start technology function with rising edge		
Sync Encoder	BOOL	With/without encoder synchronization		
Dataset	INT	Data set number		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	Data set was changed		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

5.2.8 MC_MoveAbsolute

This technology function is available in ...

This technology function starts a positioning motion of an axis to an absolute position.

Table 5-13 MC_MoveAbsolute

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object	TO_Positio-ningAxis	Axis
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Position	REAL	Target position	LREAL	Position
Velocity	REAL	Velocity	LREAL	Velocity
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
Direction	INT	Direction command for modulo axes	INT	Direction
Mode	INT	Override mode		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	Target position reached	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy

CPU31xT			S7-1500(T)	
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-14

Direction input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use default of modulo axis **Default setting**	[Not available]
1	Positive direction of rotation	Positive direction of rotation **Default setting**
2	Shortest path	Negative direction of rotation
3	Negative direction of rotation	Shortest path
4	Keep last used direction of rotation	[Not available]

Table 5-15

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	**Override motion** The current motion is canceled **Default setting**	

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
1	**Append motion** The motion job is written to the job buffer, the axis stops at the motion transition	
2	**Overlay motion** The motion job is written to the job buffer, the motion transition is overlaid	

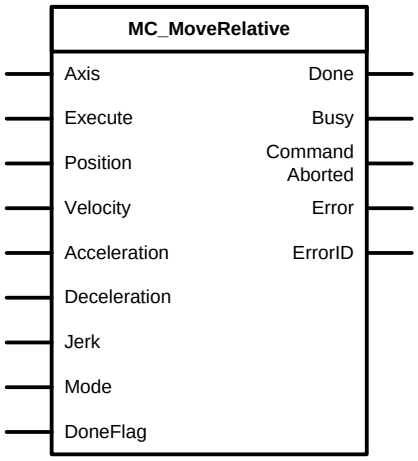
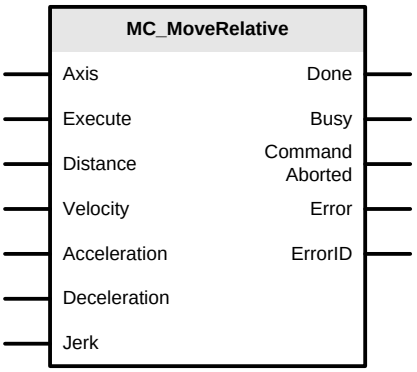
Note All technology functions for motion functions in the S7-1500 / S7-1500T currently have no mode input for defining the override response of the technology function. Regarding their response, these technology functions override one another, i.e., a currently active motion function is overridden and thus stopped by starting another motion function for the same technology object.

5.2.9 MC_MoveRelative

This technology function is available in ...

This technology function starts a positioning motion of an axis to a position relative to the start position.

Table 5-16 MC_MoveRelative

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Axis	INT	Technology object	TO_Positio-ningAxis	Axis
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Distance	REAL	Distance for the positioning operation	LREAL	Distance
Velocity	REAL	Velocity	LREAL	Velocity
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
Mode	INT	Override mode		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

Output parameters (OUT)

Parameter	Data type	Function	Data type	Parameter
Done	BOOL	Target position reached	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error

CPU31xT			S7-1500(T)	
ErrorID	WORD	Error ID	WORD	ErrorID

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-17

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	**Override motion** The current motion is canceled **Default setting**	
1	**Append motion** The motion job is written to the job buffer, the axis stops at the motion transition	
2	**Overlay motion** The motion job is written to the job buffer, the motion transition is overlaid	

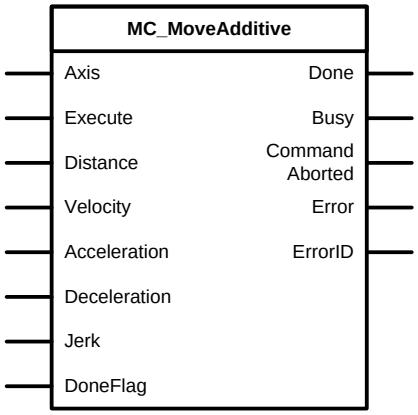
Note All technology functions for motion functions in the S7-1500 / S7-1500T currently have no mode input for defining the override response of the technology function. Regarding their response, these technology functions override one another, i.e., a currently active motion function is overridden and thus stopped by starting another motion function for the same technology object.

5.2.10 MC_MoveAdditive

This technology function is available in ...

This technology function starts axis positioning with user-definable dynamic values to a position that is relative to the target position of a current positioning job.

Table 5-18 MC_MoveAdditive

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object		
Execute	BOOL	Start technology function with rising edge		
Distance	REAL	Distance for the positioning operation		
Velocity	REAL	Velocity		
Acceleration	REAL	Acceleration		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	Target position reached		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

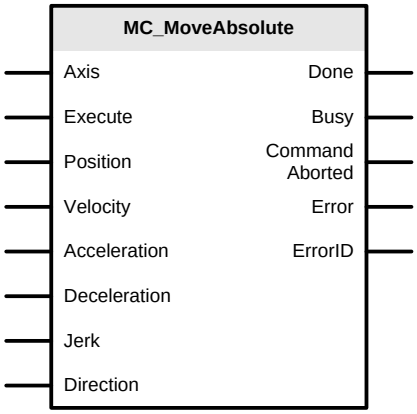
For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

Application-specific implementation in the S7-1500 / S7-1500T

To implement this function specific to an application in the S7-1500 / S7-1500T, please use the MC_MoveAbsolute technology function.

Figure 5-1 MC_MoveAbsolute technology function of the S7-1500(T)



At the Position parameter, specify the following value as the target position:

Position = Axis.StatusPositioning.TargetPosition + Distance

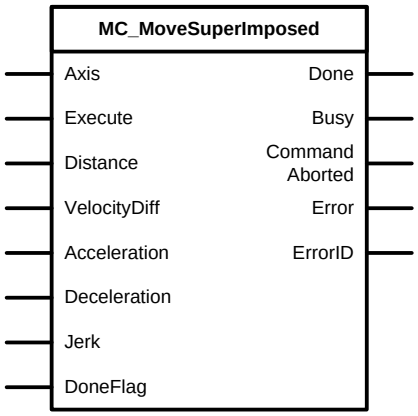
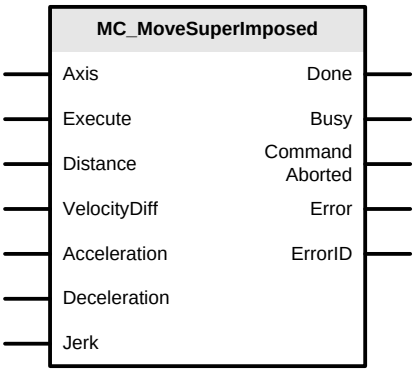
The value Distance corresponds to the distance for the additive positioning operation from the user program of the CPU31xT.

5.2.11 MC_MoveSuperImposed

This technology function is available in ...

This technology function starts superimposed positioning of an axis for an active motion job.

Table 5-19 MC_MoveSuperImposed

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object	TO_Positio-ningAxis	Axis
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Distance	REAL	Additional distance for superimposed motion	LREAL	Distance
VelocityDiff	REAL	Maximum velocity deviation compared to current motion	LREAL	VelocityDiff
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	Target position reached	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error

CPU31xT			S7-1500(T)	
ErrorID	WORD	Error ID	WORD	ErrorID

For the VelocityDiff, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

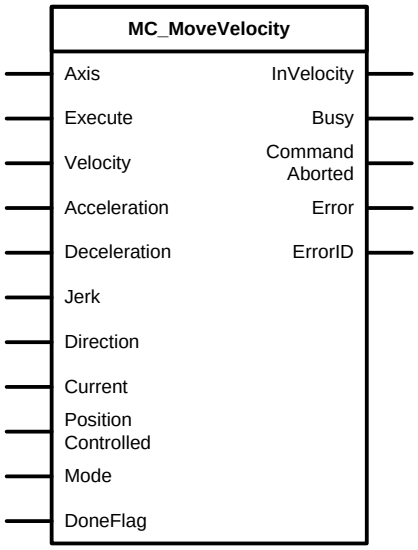
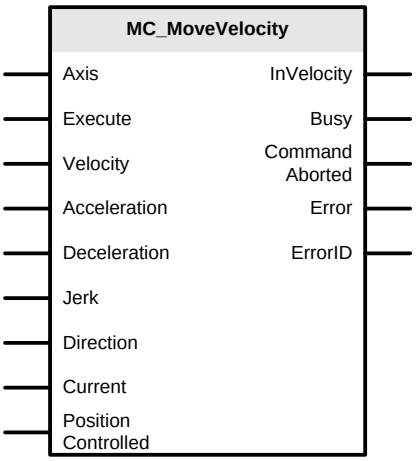
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.12 MC_MoveVelocity

This technology function is available in ...

This technology function causes the axis to move at a specified speed.

Table 5-20 MC_MoveVelocity

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object	TO_Speed Axis	Axis
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Velocity	REAL	Maximum velocity	LREAL	Velocity
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
Direction	INT	Direction command	INT	Direction
Current	BOOL	Maintain current velocity	BOOL	Current
Position Controlled	BOOL	Speed-controlled or position-controlled mode	BOOL	Position Controlled
Mode	INT	Override mode		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

CPU31xT			S7-1500(T)	
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
InVelocity	BOOL	Velocity reached	BOOL	InVelocity
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-21

Direction input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use default of axis **Default setting**	The sign of the Velocity parameter defines the direction of rotation. **Default setting**
1	Positive direction of rotation	Positive direction of rotation
2	Setting not allowed	Negative direction of rotation
3	Negative direction of rotation	[Not available]
4	Keep last used direction of rotation	[Not available]

Table 5-22

Current input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
False	The "maintain current velocity" function is disabled. **Default setting**	The "maintain current velocity" function is disabled. **Default setting**
True	The current velocity and direction are retained.	The current velocity and direction are retained.

Table 5-23

PositionControlled input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
False	Speed-controlled mode	Non-position-controlled mode
True	Position-controlled mode **Default setting**	Position-controlled mode **Default setting**

Table 5-24

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	**Override motion** The current motion is canceled. **Default setting**	
1	**Append motion** The motion job is written to the job buffer, the axis stops at the motion transition	
2	**Overlay motion** The motion job is written to the job buffer, the motion transition is overlaid.	

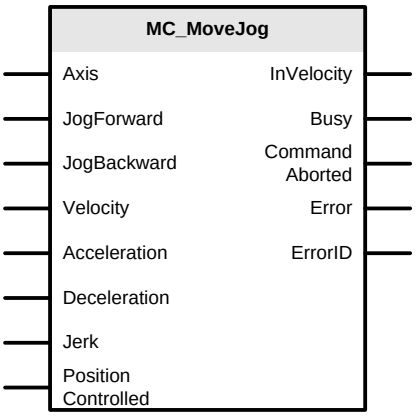
Note All technology functions for motion functions in the S7-1500 / S7-1500T currently have no mode input for defining the override response of the technology function. Regarding their response, these technology functions override one another, i.e., a currently active motion function is overridden and thus stopped by starting another motion function for the same technology object.

5.2.13 MC_MoveJog

This technology function is available in ...

This technology function moves an axis in jog mode.

Table 5-25 MC_MoveJog

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Technology object	TO_Speed Axis	Axis
		Move axis in positive direction	BOOL	Jog Forward
		Move axis in negative direction	BOOL	Jog Backward
		Velocity	LREAL	Velocity
		Acceleration	LREAL	Acceleration
		Deceleration	LREAL	Deceleration
		Jerk	LREAL	Jerk
		Speed-controlled or position-controlled mode	BOOL	Position Controlled

Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Velocity reached	BOOL	InVelocity
		The job is being processed	BOOL	Busy
		The job was aborted by another job	BOOL	Command Aborted
		An error occurred while processing the job	BOOL	Error
		Error ID	WORD	ErrorID

For the Velocity parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : A moving axis is stopped with the configured deceleration. When the setpoint velocity zero is reached, the value True is displayed at the "InVelocity" block output.
- Value < 0 : Use the default setting of the technology object

For the Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

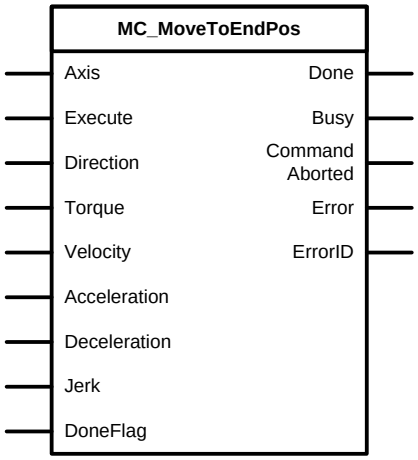
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.14 MC_MoveToEndPos

This technology function is available in ...

This technology function moves the axis against an obstacle and holds it in this position.

Table 5-26 MC_MoveToEndPos

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object		
Execute	BOOL	Start technology function with rising edge		
Direction	INT	Direction command		
Torque	REAL	Torque limit on drive		
Velocity	REAL	Maximum velocity		
Acceleration	REAL	Acceleration		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
InClamping	BOOL	Clamping reached		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		

CPU31xT			S7-1500(T)	
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Torque parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value <= 0 : This setting is not allowed

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-27

Direction input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use default of axis **Default setting**	
1	Positive direction of rotation	
2	Setting not allowed	
3	Negative direction of rotation	
4	Keep last used direction of rotation	

Implementation of this functionality in the S7-1500 / S7-1500T

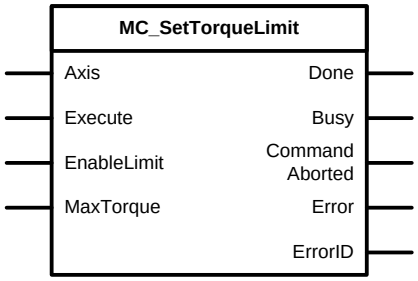
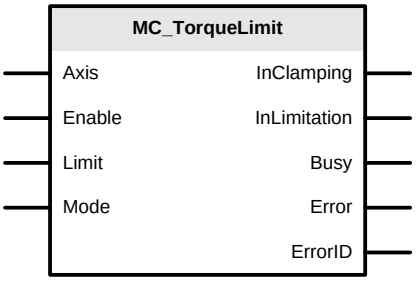
In the S7-1500 / S7-1500T, this function can be implemented with the aid of the MC_TorqueLimit technology function.

5.2.15 MC_SetTorqueLimit / MC_TorqueLimit

This technology function is available in ...

This technology function enables / disables force or torque limiting.

Table 5-28 MC_SetTorqueLimit / MC_TorqueLimit

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Technology object	TO_Speed Axis	Axis
Execute	BOOL	Start technology function with rising edge		
		Enable technology function according to parameters	BOOL	Enable
EnableLimit	BOOL	Enable and disable limit		
MaxTorque	REAL	Effective max. torque on motor		
		Force or torque limit value	LREAL	Limit
		Mode selection	DINT	Mode
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		The drive is kept at the fixed stop	BOOL	InClamping
		The drive force or torque is limited	BOOL	InLimitation
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

For the Limit parameter, please note the following:

- Value ≥ 0 : Use the value specified at the parameter
- Value < 0 : Use the default torque limit

The following parameters involve functional differences or differences in the parameterization:

Table 5-29

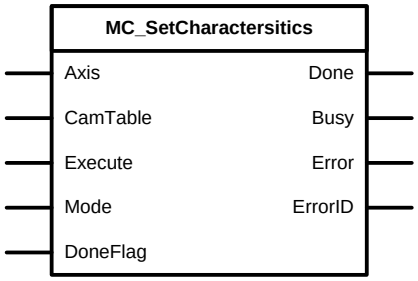
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0		Run as force / torque limiting. **Default setting**
1		Run as fixed stop detection.

5.2.16 MC_SetCharacteristics

This technology function is available in ...

This technology function enables a valve characteristic for a Q valve or P valve.

Table 5-30 MC_SetCharacteristics

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Axis technology object		
CamTable	INT	Cam technology object		
Execute	BOOL	Start technology function with rising edge		
Mode	INT	Mode selection		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

The following parameters involve functional differences or differences in the parameterization:

Table 5-31

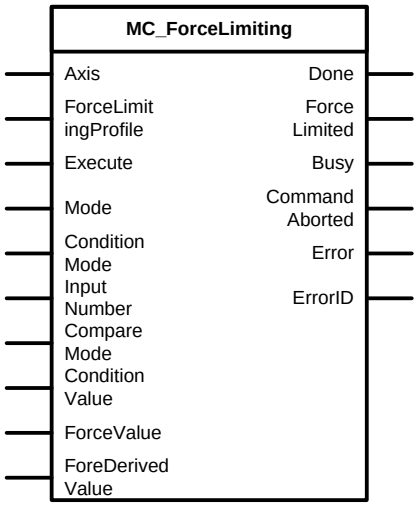
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Characteristic for Q valve stored in cam. **Default setting**	
1	Characteristic for P valve stored in cam.	

5.2.17 MC_ForceLimiting

This technology function is available in ...

This technology function enables / disables force or pressure limiting for an axis.

Table 5-32 MC_ForceLimiting

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Axis technology object		
ForceLimitingProfile	INT	Cam technology object		
Execute	BOOL	Start technology function with rising edge		
Mode	INT	Mode selection		
Condition Mode	INT	Start condition for pressure limiting job		
Input Number	INT	Number of the digital input of the comparison pressure sensor		
Compare Mode	INT	Comparison condition		
Condition Value	REAL	Comparison value for switching condition		
ForceValue	REAL	Force / pressure limiting value		
ForceDerivedValue	REAL	Rate-of-change limiting for pressure limiting signal		

CPU31xT			S7-1500(T)	
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Force Limited	BOOL	Pressure limiting condition met or job issued		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the ForceDerivedValue parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value ≤ 0 : Use the default setting on the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-33

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Enable/disable force / pressure limiting. **Default setting**	
1	Immediately enable force / pressure limiting.	
2	Immediately enable force / pressure limiting as a function of time.	
3	Enable force / pressure limiting starting from the time a condition is met.	
4	Enable force / pressure limiting as a function of time starting from the time a condition is met.	
5	Enable force / pressure limiting as a function of position starting from the time a condition is met.	

Table 5-34

ConditionMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use default of technology object **Default setting**	
1	Enable force / pressure limiting after a delay time.	
2	Enable force / pressure limiting based on the axis position.	
3	Enable force / pressure limiting based on the pressure sensor value.	
4	Enable force / pressure limiting following an edge at the digital input.	

Table 5-35

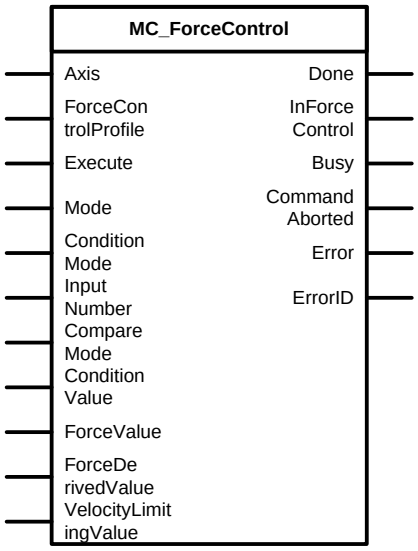
CompareMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	
1	Value greater than or equal to ConditionValue or logic state 1 at the digital input	
2	Value less than ConditionValue or logic state 0 at the digital input	

5.2.18 MC_ForceControl

This technology function is available in ...

This technology function enables and disables the pressure control mode on an axis.

Table 5-36 MC_ForceControl

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	Axis technology object		
ForceControlProfile	INT	Cam technology object		
Execute	BOOL	Start technology function with rising edge		
Mode	INT	Mode selection		
Condition Mode	INT	Start condition for pressure control job		
Input Number	INT	Number of the digital input of the comparison pressure sensor		
Compare Mode	INT	Comparison condition		
Condition Value	REAL	Comparison value for switching condition		
ForceValue	REAL	Force / pressure limiting value		
ForceDerivedValue	REAL	Rate-of-change limiting for pressure limiting signal		

CPU31xT			S7-1500(T)	
VelocityLimitingValue	REAL	Velocity limiting value		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
InForce Control	BOOL	Pressure control was enabled		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the ForceDerivedValue parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value ≤ 0 : Use the default setting on the technology object

For the VelocityLimitingValue parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : without limiting
- Value < 0 : Use the default setting on the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-37

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	
1	Immediately enable force / pressure control. **Default setting**	
2	Immediately enable force / pressure control as a function of time.	
3	Enable force / pressure control starting from the time a condition is met.	
4	Enable force / pressure control as a function of time starting from the time a condition is met.	
5	Enable force / pressure control as a function of position starting from the time a condition is met.	

Table 5-38

ConditionMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use default of technology object **Default setting**	
1	Enable force / pressure control after a delay time.	
2	Enable force / pressure control based on the axis position.	
3	Enable force / pressure control based on the pressure sensor value.	
4	Enable force / pressure control following an edge at the digital input.	

Table 5-39

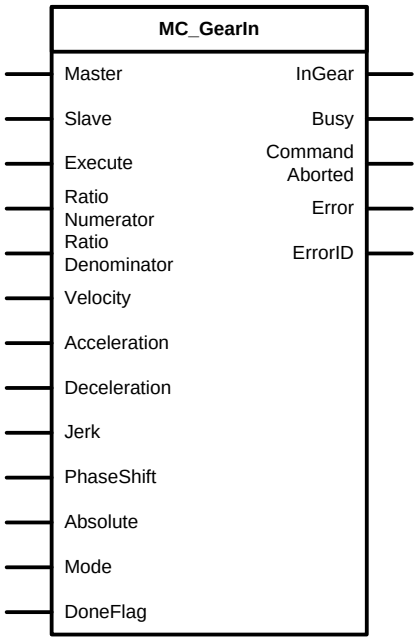
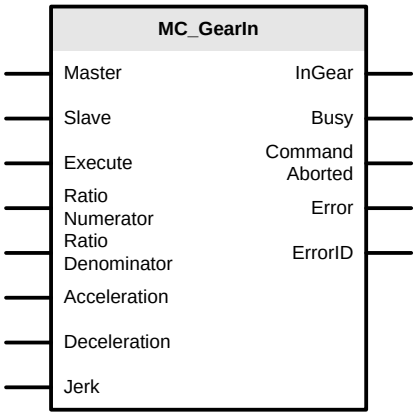
CompareMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	
1	Value greater than or equal to ConditionValue or logic state 1 at the digital input	
2	Value less than ConditionValue or logic state 0 at the digital input	

5.2.19 MC_GearIn / MC_GearIn

This technology function is available in ...

This technology function starts relative gearing between a master axis and a slave axis.

Table 5-40 MC_GearIn

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Master	INT	Master Axis technology object	TO_Axis	Master
Slave	INT	Slave Axis technology object	TO_SynchronousAxis	Slave
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Ratio Numerator	DINT	Gear ratio numerator	DINT	Ratio Numerator
RatioDenominator	DINT	Gear ratio denominator	DINT	RatioDenominator
Velocity	REAL	Maximum velocity for synchronizing		
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk

CPU31xT			S7-1500(T)	
PhaseShift	REAL	Phase offset after reaching synchronous operation		
Absolute	BOOL	Absolute gearing		
Mode	INT	Synchronization mode selection		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
InGear	BOOL	Synchronous operation of the two axes reached	BOOL	InGear
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

Note For the SIMATIC S7-1500(T), only relative gearing can be established using the GearIn function.

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-41

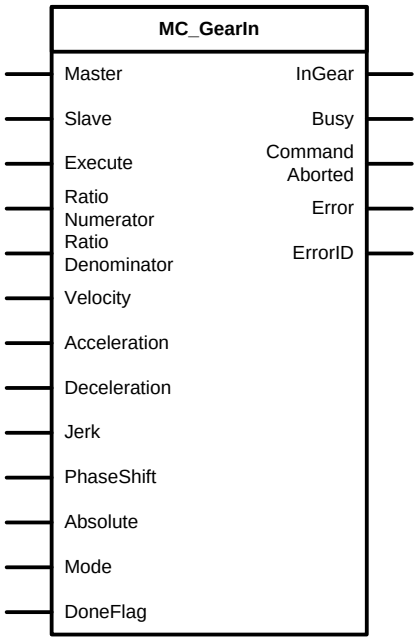
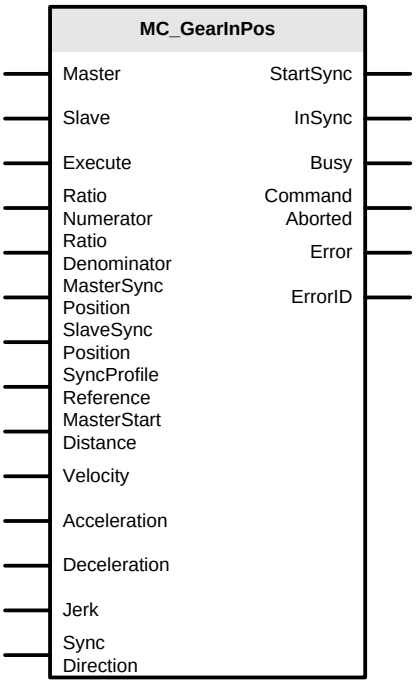
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use the default setting of the synchronous object. **Default setting**	
1	Time-based synchronization	
2	Time-based synchronization considering the direction	

5.2.20 MC_GearIn / MC_GearInPos


This technology function is available in ...

This technology function starts absolute gearing between a master axis and a slave axis.

Table 5-42 MC_GearIn

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Master	INT	Master Axis technology object	TO_Axis	Master
Slave	INT	Slave Axis technology object	TO_SynchronousAxis	Slave
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Ratio Numerator	DINT	Gear ratio numerator	DINT	Ratio Numerator
RatioDenominator	DINT	Gear ratio denominator	DINT	RatioDenominator
		Master axis position starting from which both axes move synchronously	LREAL	MasterSync Position

CPU31xT			S7-1500(T)	
		Slave axis position starting from which the two axes move synchronously	LREAL	SlaveSync Position
		Synchronization type	DINT	SyncProfile Reference
		Master value distance	LREAL	MasterStart Distance
Velocity	REAL	Maximum velocity for synchronizing	LREAL	Velocity
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
		Synchronization direction	DINT	Sync Direction
PhaseShift	REAL	Phase offset after reaching synchronous operation		
Absolute	BOOL	Absolute gearing		
Mode	INT	Synchronization mode selection		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Start of synchronization	BOOL	StartSync
InGear	BOOL	Synchronous operation of the two axes reached	BOOL	InSync
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

Note For the SIMATIC S7-1500T, only absolute gearing can be established using the GearInPos function.

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-43

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use the default setting of the synchronous object. **Default setting**	
1	Time-based synchronization	
2	Time-based synchronization considering the direction	

Table 5-44

SyncProfileReference input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0		Synchronization via dynamic response parameters **Default setting**
1		Synchronization via master value distance

Table 5-45

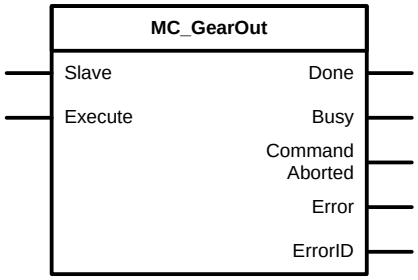
SyncDirection input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0		[Not available]
1		Positive direction
2		Negative direction
3		Shortest path **Default setting**

5.2.21 MC_GearOut

This technology function is available in ...

This technology function stops active gearing.

Table 5-46 MC_GearOut

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Slave	INT	Slave Axis technology object		
Execute	BOOL	Start technology function with rising edge		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

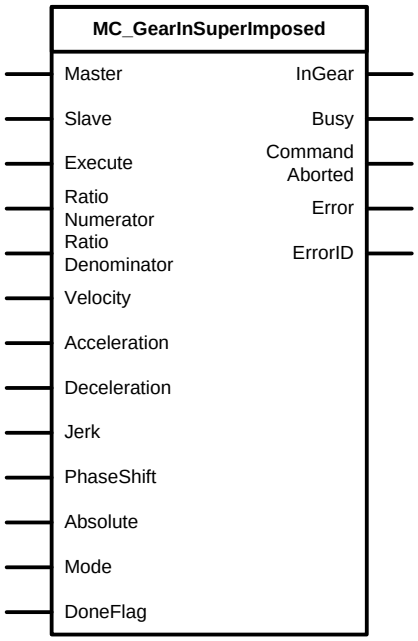
Note For the SIMATIC S7-1500(T), active gearing can only be stopped by running a suitable technology function for the slave axis.

5.2.22 MC_GearInSuperImposed

This technology function is available in ...

This technology function starts superimposed gearing between a master axis and a slave axis.

Table 5-47 MC_GearInSuperImposed

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Master	INT	Master Axis technology object		
Slave	INT	Slave Axis technology object		
Execute	BOOL	Start technology function with rising edge		
Ratio Numerator	DINT	Gear ratio numerator		
RatioDe nominator	DINT	Gear ratio denominator		
Velocity	REAL	Maximum velocity for synchronizing		
Acceleration	REAL	Acceleration		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		

CPU31xT			S7-1500(T)	
PhaseShift	REAL	Phase offset after reaching synchronous operation		
Absolute	BOOL	Absolute gearing		
Mode	INT	Synchronization mode selection		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
InGear	BOOL	Synchronous operation of the two axes reached		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-48

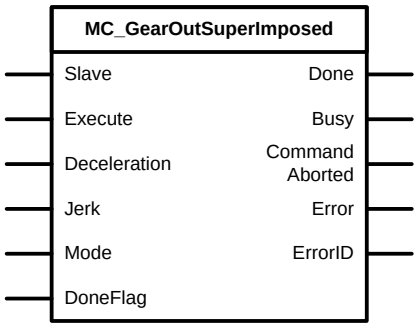
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use the default setting of the synchronous object. **Default setting**	
1	Time-based synchronization	
2	Time-based synchronization considering the direction	

5.2.23 MC_GearOutSuperImposed

This technology function is available in ...

This technology function stops superimposed gearing.

Table 5-49 MC_GearOutSuperImposed

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Slave	INT	Slave Axis technology object		
Execute	BOOL	Start technology function with rising edge		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		
Mode	INT	Desynchronization mode selection		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Deceleration parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-50

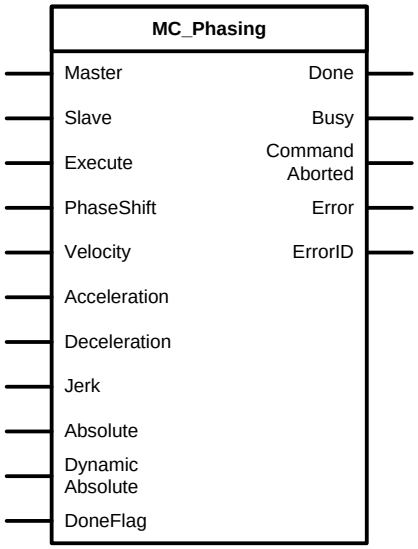
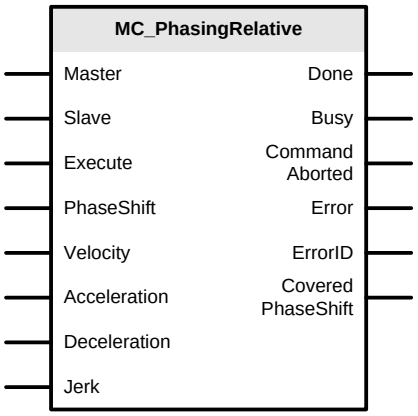
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use the default setting of the synchronous object. **Default setting**	
1	Desynchronize immediately according to dynamic response parameters	

5.2.24 MC_Phasing / MC_PhasingRelative


This technology function is available in ...

This technology function is used for relative shifting of the master value to absolute gearing.

Table 5-51 MC_Phasing

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Master	INT	Master Axis technology object	TO_Axis	Master
Slave	INT	Slave Axis technology object	TO_SynchronousAxis	Slave
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
PhaseShift	REAL	Phase shift	LREAL	PhaseShift
Velocity	REAL	Maximum velocity for setting the phase angle	LREAL	Velocity
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
Absolute	BOOL	Absolute phase shift to current phase		
Absolute Dynamic	BOOL	Absolute specification of the dynamic values for the axis		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				

CPU31xT			S7-1500(T)	
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
		Master value offset implemented so far	LREAL	Covered PhaseShift

Note For the SIMATIC S7-1500T, only a relative phase shift can be implemented in active absolute gearing using the technology function listed here.

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

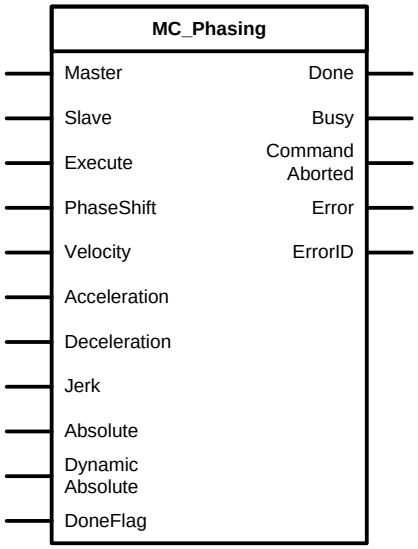
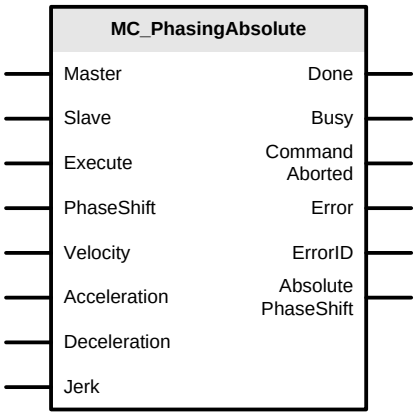
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.25 MC_Phasing / MC_PhasingAbsolute


This technology function is available in ...

This technology function is used for absolute shifting of the master value to absolute gearing.

Table 5-52 MC_Phasing

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Master	INT	Master Axis technology object	TO_Axis	Master
Slave	INT	Slave Axis technology object	TO_SynchronousAxis	Slave
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
PhaseShift	REAL	Phase shift	LREAL	PhaseShift
Velocity	REAL	Maximum velocity for setting the phase angle	LREAL	Velocity
Acceleration	REAL	Acceleration	LREAL	Acceleration
Deceleration	REAL	Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
Absolute	BOOL	Absolute phase shift to current phase		
Absolute Dynamic	BOOL	Absolute specification of the dynamic values for the axis		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

CPU31xT			S7-1500(T)	
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
		Master value offset implemented so far	LREAL	Absolute PhaseShift

Note For the SIMATIC S7-1500T, only an absolute phase shift can be implemented in active absolute gearing using the technology function listed here.

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

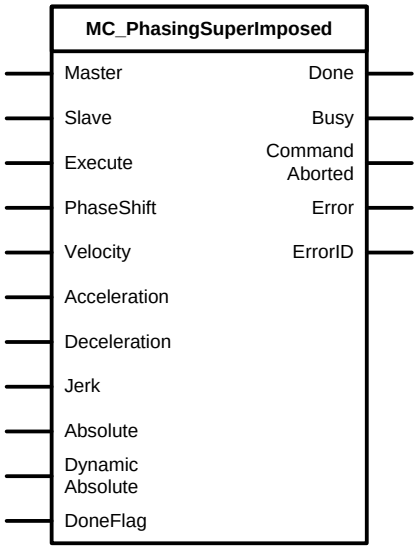
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.26 MC_PhasingSuperImposed

This technology function is available in ...

This technology function is used for a superimposed phase shift between a master axis and a slave axis.

Table 5-53 MC_PhasingSuperImposed

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Master	INT	Master Axis technology object		
Slave	INT	Slave Axis technology object		
Execute	BOOL	Start technology function with rising edge		
PhaseShift	REAL	Phase shift		
Velocity	REAL	Maximum velocity for setting the phase angle		
Acceleration	REAL	Acceleration		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		
Absolute	BOOL	Absolute phase shift to current phase		
Dynamic Absolute	BOOL	Absolute specification of the dynamic values for the axis		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

CPU31xT			S7-1500(T)	
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

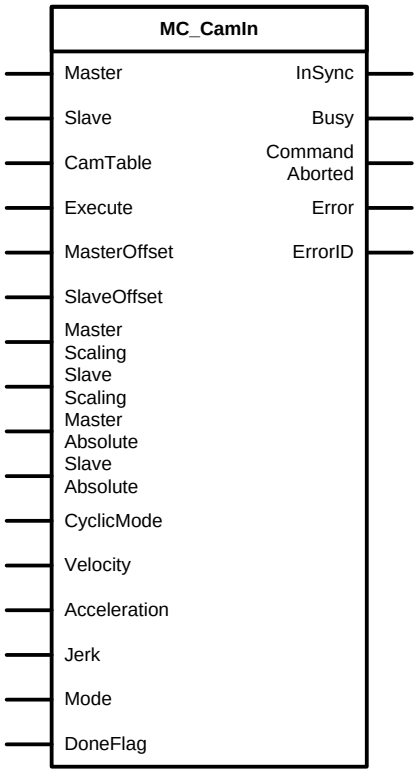
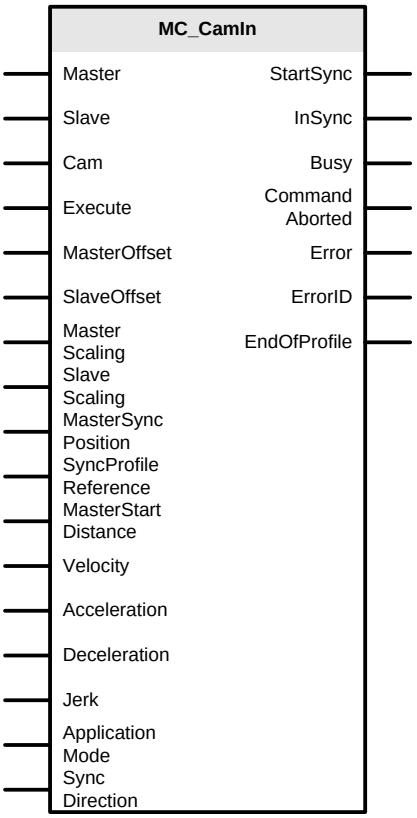
- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

5.2.27 MC_CamIn


This technology function is available in ...

This technology function starts camming between a master axis and a slave axis.

Table 5-54 MC_CamIn

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Master	INT	Master Axis technology object	TO_Axis	Master
Slave	INT	Slave Axis technology object	TO_SynchronousAxis	Slave
CamTable	INT	Cam technology object	TO_Cam	Cam
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Master Offset	REAL	Shift cam in master axis coordinates	LREAL	Master Offset
Slave Offset	REAL	Shift cam in slave axis coordinates	LREAL	Slave Offset
Master Scaling	REAL	Scale cam in master axis coordinates	LREAL	Master Scaling

CPU31xT			S7-1500(T)	
Slave Scaling	REAL	Scale cam in slave axis coordinates	LREAL	Slave Scaling
Master Absolute	BOOL	Cam interpreted absolute to master axis		
Slave Absolute	BOOL	Cam interpreted absolute to slave axis		
CyclicMode	BOOL	Cyclic execution of cam		
		Master axis position where synchronization is complete.	LREAL	MasterSync Position
		Synchronization profile	DINT	SyncProfile Reference
		Master value distance	LREAL	MasterStart Distance
Velocity	REAL	Maximum velocity for synchronizing	LREAL	Velocity
Acceleration	REAL	Acceleration	LREAL	Acceleration
		Deceleration	LREAL	Deceleration
Jerk	REAL	Jerk	LREAL	Jerk
		Use of cam	DINT	Application Mode
		Synchronization direction	DINT	Sync Direction
Mode	INT	Synchronization mode		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Start of synchronization	BOOL	StartSync
InSync	BOOL	Camming reached	BOOL	InSync
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
		End of cam reached	BOOL	EndOfProfile

For the Velocity, Acceleration and Deceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-55

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use the default setting of the synchronous object. **Default setting**	
1	Time-based synchronization	
2	Time-based synchronization considering the direction	

Table 5-56

ApplicationMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0		Single non-cyclic use of cam. **Default setting**
1		Cyclic use of cam absolute to slave axis.
2		Cyclic use of cam continually appending to slave axis.

Table 5-57

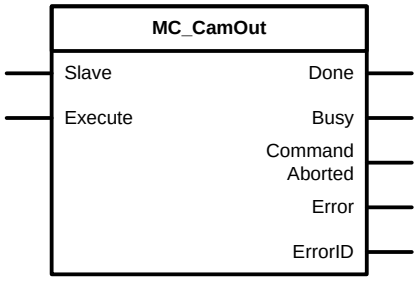
SyncDirection input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0		[Not available]
1		Positive synchronization direction
2		Negative synchronization direction
3		Shortest path **Default setting**

5.2.28 MC_CamOut

This technology function is available in ...

This technology function stops camming between a master axis and a slave axis.

Table 5-58 MC_CamOut

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Slave	INT	Slave Axis technology object		
Execute	BOOL	Start technology function with rising edge		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

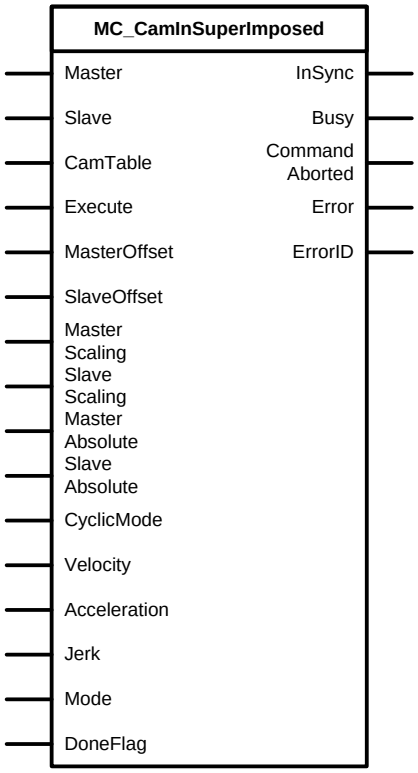
Note For the SIMATIC S7-1500T, active camming can only be stopped by running a suitable technology function for the slave axis.

5.2.29 MC_CamInSuperImposed

This technology function is available in ...

This technology function starts superimposed camming between a master axis and a slave axis.

Table 5-59 MC_CamInSuperImposed

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Master	INT	Master Axis technology object		
Slave	INT	Slave Axis technology object		
CamTable	INT	Cam technology object		
Execute	BOOL	Start technology function with rising edge		
Master Offset	REAL	Shift cam in master axis coordinates		
Slave Offset	REAL	Shift cam in slave axis coordinates		
Master Scaling	REAL	Scale cam in master axis coordinates		

CPU31xT			S7-1500(T)	
Slave Scaling	REAL	Scale cam in slave axis coordinates		
Master Absolute	BOOL	Cam interpreted absolute to master axis		
Slave Absolute	BOOL	Cam interpreted absolute to slave axis		
CyclicMode	BOOL	Cyclic execution of cam		
Velocity	REAL	Maximum velocity for synchronizing		
Acceleration	REAL	Acceleration		
Jerk	REAL	Jerk		
Mode	INT	Synchronization mode		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
InSync	BOOL	Camming reached		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Velocity and Acceleration parameters, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-60

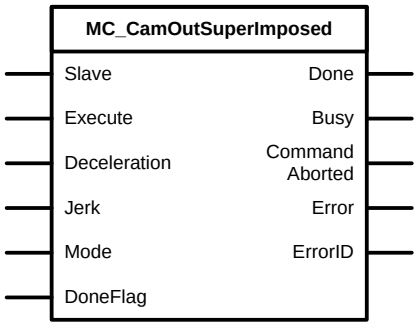
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use the default setting of the synchronous object. **Default setting**	
1	Time-based synchronization	
2	Time-based synchronization considering the direction	

5.2.30 MC_CamOutSuperImposed

This technology function is available in ...

This technology function stops superimposed camming between a master axis and a slave axis.

Table 5-61 MC_CamOutSuperImposed

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Slave	INT	Slave Axis technology object		
Execute	BOOL	Start technology function with rising edge		
Deceleration	REAL	Deceleration		
Jerk	REAL	Jerk		
Mode	INT	Decoupling mode		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

For the Deceleration parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : This setting is not allowed
- Value < 0 : Use the default setting of the technology object

For the Jerk parameter, please note the following:

- Value > 0 : Use the value specified at the parameter
- Value = 0 : Use the trapezoidal velocity profile
- Value < 0 : Use the default setting of the technology object

The following parameters involve functional differences or differences in the parameterization:

Table 5-62

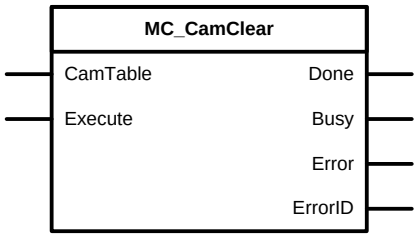
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Use the default setting of the synchronous object. **Default setting**	
1	Decouple according to dynamic response setting on the block.	

5.2.31 MC_CamClear


This technology function is available in ...

This technology function clears all interpolation points of a cam and sets the cam to edit mode.

Table 5-63 MC_CamClear

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTable	INT	Cam technology object		
Execute	BOOL	Start technology function with rising edge		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

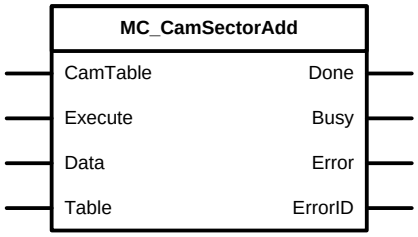
Note For the SIMATIC S7-1500T, clearing the defined interpolation points of a cam can be achieved by directly writing to the cam's data block.

5.2.32 MC_CamSectorAdd


This technology function is available in ...

This technology function adds a new cam sector to a cam.

Table 5-64 MC_CamSectorAdd

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTable	INT	Cam technology object		
Execute	BOOL	Start technology function with rising edge		
Data	ANY	Data area of the cam segments to be inserted		
Table	INT	Specification of the data area		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

The following parameters involve functional differences or differences in the parameterization:

Table 5-65

Table input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	The Data data area specifies the data as a mathematical polynomial. **Default setting**	
1	The Data data area specifies the data as an interpolation point table.	

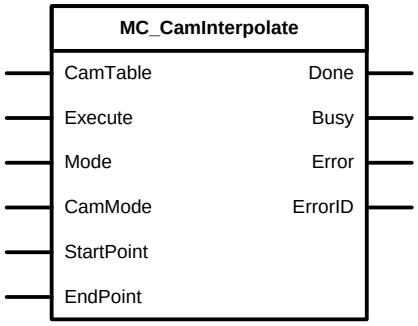
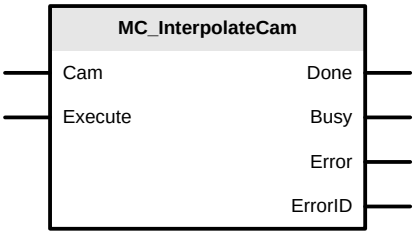
Note For the SIMATIC S7-1500T, inserting interpolation points or polynomials of a cam can be achieved by directly writing to the cam's data block.

5.2.33 MC_CamInterpolate / MC_InterpolateCam


This technology function is available in ...

This technology function interpolates a cam and prepares it for use.

Table 5-66 MC_CamInterpolate / MC_InterpolateCam

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTable	INT	Cam technology object	TO_Cam	Cam
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Mode	INT	Interpolation mode		
CamMode	INT	Cam type		
StartPoint	REAL	Start point of definition range		
EndPoint	REAL	End point of definition range		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

The following parameters involve functional differences or differences in the parameterization:

Table 5-67

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Linear interpolation **Default setting**	
1	Cubic splines	
2	Bezier splines	

Table 5-68

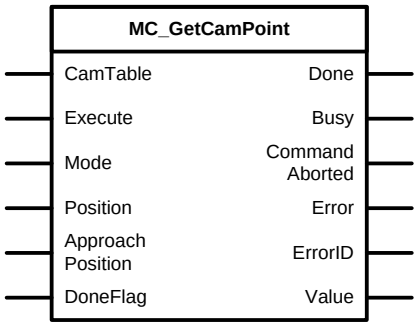
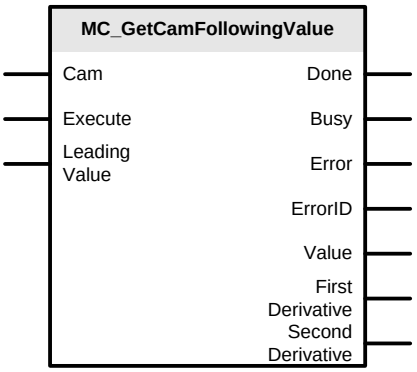
CamMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Cyclic relative – constant velocity. **Default setting**	
1	Cyclic absolute – constant position and velocity	
2	Non-cyclic – not constant at the edge points	

5.2.34 MC_GetCamPoint / MC_GetCamFollowingValue


This technology function is available in ...

This technology function reads the slave value defined for a master value and the associated first and second derivative from a cam.

Table 5-69 MC_GetCamPoint / MC_GetCamFollowingValue

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTable	INT	Cam technology object	TO_Cam	Cam
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Mode	INT	Mode selection		
Position	REAL	Position for which the value is to be determined	LREAL	Leading Value
Approach Position	REAL	Assumed master axis position		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
Value	REAL	Read position value	LREAL	Value

CPU31xT			S7-1500(T)	
		Function value of the first derivative (velocity)	LREAL	First Derivative
		Function value of the second derivative (acceleration)	LREAL	Second Derivative

The following parameters involve functional differences or differences in the parameterization:

Table 5-70

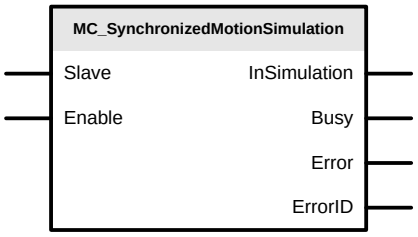
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	The position of the slave axis is determined relative to a master axis position. **Default setting**	
1	The position of the master axis is determined relative to a slave axis position.	
2	The first derivative of the cam point is calculated relative to a master axis position (velocity)	
3	The second derivative of the cam point is calculated relative to a master axis position (acceleration)	

5.2.35 MC_SynchronizedMotionSimulation

This technology function is available in ...

This technology function sets a synchronous operation active on a slave axis to simulation, i.e., the synchronous operation remains active when disabling the slave axis via MC_Power and it is not necessary to re-synchronize it after a re-enable.

Table 5-71 MC_SynchronizedMotionSimulation

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Technology object of slave axis	TO_SynchronousAxis	Slave
		Set synchronous operation to simulation	BOOL	Enable
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		The synchronous operation is in simulation and remains active	BOOL	InSimulation
		The job is being processed	BOOL	Busy
		An error occurred while processing the job	BOOL	Error
		Error ID	WORD	ErrorID

This function suppresses only the setpoint from the active synchronous operation for the slave axis. However, setpoints from superimposed motions continue to be transferred to the slave axis. Likewise, due to them, single-axis motion jobs for the slave axis are executed during the simulation.

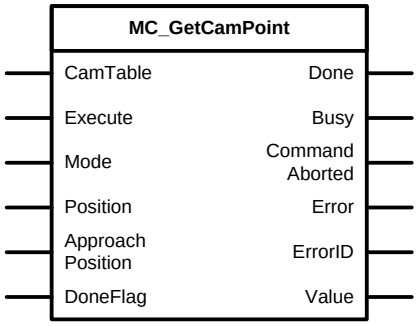
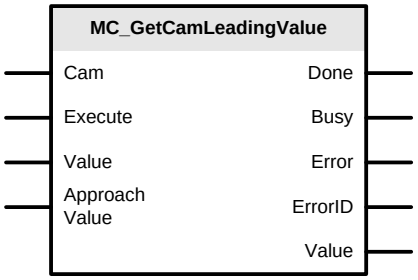
If the slave axis was moved during an active synchronous operation simulation, canceling the simulation results in a setpoint jump on the slave axis. Therefore, when canceling the simulation, the slave axis should preferably be at the same position as when the simulation was started.

5.2.36 MC_GetCamPoint / MC_GetCamLeadingValue


This technology function is available in ...

This technology value reads the master value defined for a slave value from a cam.

Table 5-72 MC_GetCamPoint / MC_GetCamLeadingValue

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTable	INT	Cam technology object	TO_Cam	Cam
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Mode	INT	Mode selection		
Position	REAL	Position for which the value is to be determined	LREAL	Following Value
Approach Position	REAL	Assumed master axis position	LREAL	Approach Leading Value
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
Value	REAL	Read position value	LREAL	Value

The following parameters involve functional differences or differences in the parameterization:

Table 5-73

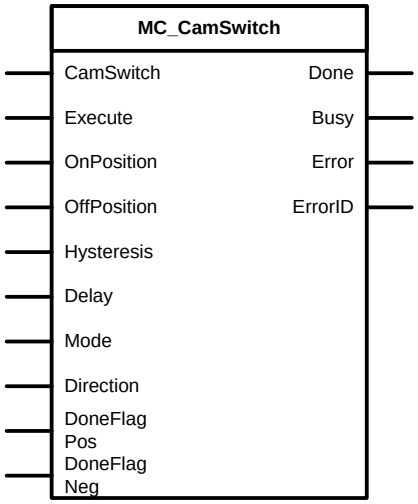
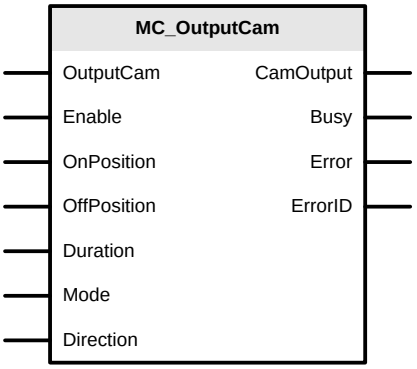
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	The position of the slave axis is determined relative to a master axis position. **Default setting**	
1	The position of the master axis is determined relative to a slave axis position.	
2	The first derivative of the cam point is calculated relative to a master axis position (velocity)	
3	The second derivative of the cam point is calculated relative to a master axis position (acceleration)	

5.2.37 MC_CamSwitch / MC_OutputCam

This technology function is available in ...

This technology function enables a set distance output cam.

Table 5-74 MC_CamSwitch / MC_OutputCam

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamSwitch	INT	Output Cam technology object	TO_Output Cam	OutputCam
Execute	BOOL	Start technology function with rising edge		
		Enable output cam output	BOOL	Enable
OnPosition	REAL	Start position	LREAL	OnPosition
OffPosition	REAL	End position	LREAL	OffPosition
		Activation duration of time-based output cam	LREAL	Duration
Hysteresis	REAL	Hysteresis	**Note:** These settings can be changed directly on the output cam.	
Delay	REAL	Time-based offset of output cam switching point		
Mode	INT	Mode selection	DINT	Mode
Direction	INT	Effective direction of output cam	DINT	Direction
DoneFlag Pos	INT	Flag for central evaluation of the status of the technology functions		
DoneFlag Neg	INT	Flag for central evaluation of the status of the technology functions		

CPU31xT			S7-1500(T)	
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Status display of the last call of the technology function	BOOL	CamOutput
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed	BOOL	Busy
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

For the Delay parameter, please note the following:

- Value < 0 : Premature activation of output cam
- Value > 0 : Delayed activation of output cam

The following parameters involve functional differences or differences in the parameterization:

Table 5-75

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	[Not available]
1	Disable output cam	Enable output cam **Default setting**
2	Enable output cam **Default setting**	Enable output cam with inverted output
3	Enable output cam with inverted output	Output cam permanently on
4	Output cam permanently on	[Not available]

Table 5-76

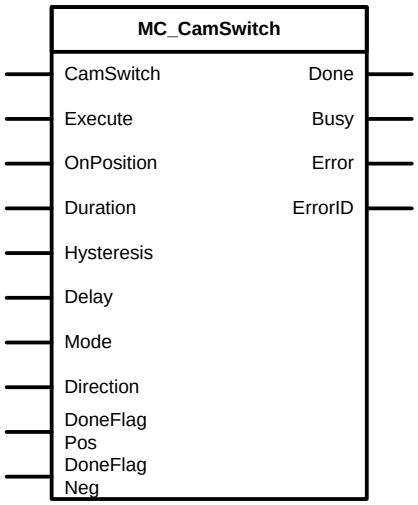
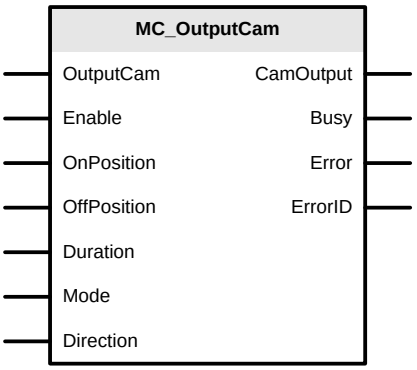
Direction input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	[Not available]
1	Positive effective direction. **Default setting**	Positive effective direction. **Default setting**
2	Positive and negative effective direction.	Negative effective direction.
3	Negative effective direction.	Positive and negative effective direction.
4	Use last active effective direction.	[Not available]

5.2.38 MC_CamSwitchTime / MC_OutputCam

This technology function is available in ...

This technology function enables a set time-based output cam.

Table 5-77 MC_CamSwitchTime / MC_OutputCam

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamSwitch	INT	Output Cam technology object	TO_Output Cam	OutputCam
Execute	BOOL	Start technology function with rising edge		
		Enable output cam output	BOOL	Enable
OnPosition	REAL	Start position	LREAL	OnPosition
		End position	LREAL	OffPosition
Duration	REAL	Activation duration of time-based output cam	LREAL	Duration
Hysteresis	REAL	Hysteresis		
Delay	REAL	Time-based offset of output cam switching point		
Mode	INT	Mode selection	DINT	Mode
Direction	INT	Effective direction of output cam	DINT	Direction
DoneFlag Pos	INT	Flag for central evaluation of the status of the technology functions		
DoneFlag Neg	INT	Flag for central evaluation of the status of the technology functions		

CPU31xT			S7-1500(T)	
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Status display of the last call of the technology function	BOOL	CamOutput
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed	BOOL	Busy
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

For the Delay parameter, please note the following:

- Value < 0 : Premature activation of output cam
- Value > 0 : Delayed activation of output cam

The following parameters involve functional differences or differences in the parameterization:

Table 5-78

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	[Not available]
1	Disable output cam	Enable output cam **Default setting**
2	Enable output cam **Default setting**	Enable output cam with inverted output
3	Enable output cam with inverted output	Output cam permanently on
4	Output cam permanently on	[Not available]

Table 5-79

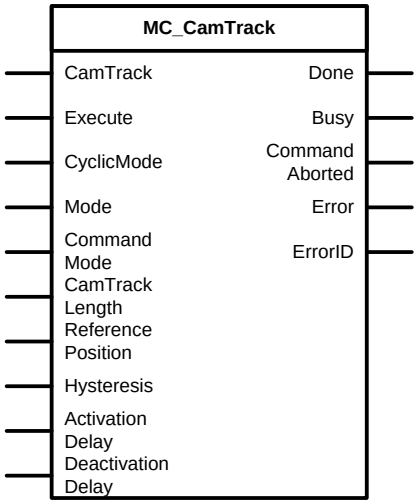
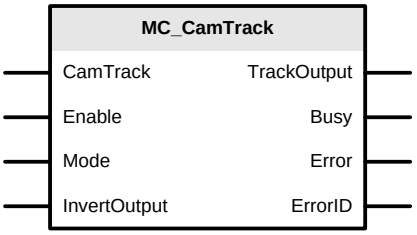
Direction input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	[Not available]
1	Positive effective direction. **Default setting**	Positive effective direction. **Default setting**
2	Positive and negative effective direction.	Negative effective direction.
3	Negative effective direction.	Positive and negative effective direction.
4	Use last active effective direction.	[Not available]

5.2.39 MC_CamTrack


This technology function is available in ...

This technology function enables a set cam track.

Table 5-80 MC_CamTrack

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTrack	INT	Cam Track technology object	TO_Cam Track	CamTrack
Execute	BOOL	Start technology function with rising edge		
		Enable cam track	BOOL	Enable
CyclicMode	BOOL	Creation of cam track at axis reference position		
Mode	INT	Mode selection	INT	Mode
		Inverted output of cam track	BOOL	InvertOutput
Command Mode	INT	Cam track activation mode		
CamTrack Length	REAL	Cam track length		
Reference Position	REAL	Axis reference position		
Hysteresis	REAL	Hysteresis		
Activation Delay	REAL	Time-based offset of output cam activation point		
Deactivation Delay	REAL	Time-based offset of output cam deactivation point		

CPU31xT			S7-1500(T)	
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
		Status display of the cam track's switching state	BOOL	TrackOutput
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed	BOOL	Busy
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID

For the ActivationDelay and DeactivationDelay parameters, please note the following:

- Value < 0 : Premature activation of output cam
- Value = 0 : No influence on cam
- Value > 0 : Delayed activation of output cam

The following parameters involve functional differences or differences in the parameterization:

Table 5-81

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	Immediately enable cam track execution.
1	Disable cam track	Enable cam track execution with the next cam track cycle. **Default setting**
2	Enable cam track **Default setting**	Cam track permanently on or off.
3	[Not allowed]	[Not available]
4	Cam track permanently on	[Not available]

Table 5-82

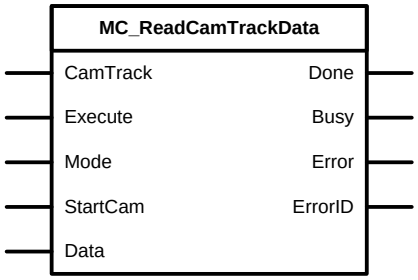
CommandMode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	
1	Immediately disable/enable cam track. **Default setting**	
2	Disable/enable cam track after exiting cam track or during transition to next cam track cycle.	
3	Disable active cam track job only after all output cams have been deactivated.	

5.2.40 MC_ReadCamTrackData

This technology function is available in ...

This technology function reads output cam data from a cam track.

Table 5-83 MC_ReadCamTrackData

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTrack	INT	Cam Track technology object		
Execute	BOOL	Start technology function with rising edge		
Mode	INT	Mode selection		
StartCam	INT	Number of the single output cam where reading is to start		
Data	ANY	Destination area for the read data		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

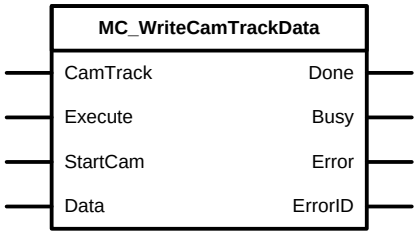
Note For the SIMATIC S7-1500T, changing a cam track can be achieved by directly writing to the cam track's data block.

5.2.41 MC_WriteCamTrackData

This technology function is available in ...

This technology function writes output cam data to a cam track.

Table 5-84 MC_WriteCamTrackData

CPU31xT	S7-1500(T)
	

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
CamTrack	INT	Cam Track technology object		
Execute	BOOL	Start technology function with rising edge		
StartCam	INT	Number of the single output cam where writing is to start		
Data	ANY	Source area of the data to be written		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

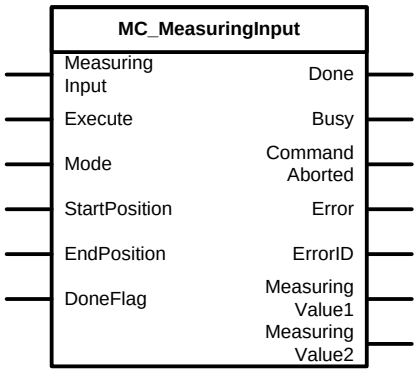
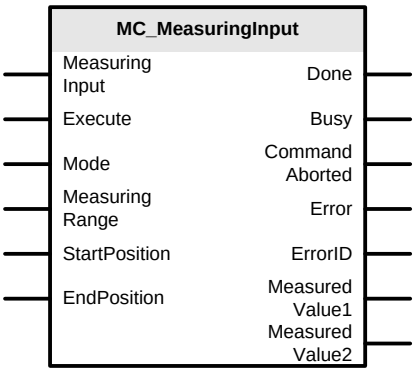
Note For the SIMATIC S7-1500T, changing a cam track can be achieved by directly writing to the cam track's data block.

5.2.42 MC_MeasuringInput / MC_MeasuringInput

This technology function is available in ...

This technology function starts a single measurement job for a measuring input.

Table 5-85 MC_MeasuringInput / MC_MeasuringInput

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Measuring Input	INT	Cam Track technology object	TO_ Measuring Input	Measuring Input
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Mode	INT	Mode selection	DINT	Mode
		Acquisition of measured values	BOOL	Measuring Range
StartPosition	REAL	Start position of measuring range	LREAL	StartPosition
EndPosition	REAL	End position of measuring range	LREAL	EndPosition
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

Output parameters (OUT)

Parameter	Data type	Function	Data type	Parameter
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
Measuring Value1	REAL	Measured value 1	LREAL	Measured Value1

CPU31xT			S7-1500(T)	
Measuring Value2	REAL	Measured value 2	LREAL	Measured Value2

The following parameters involve functional differences or differences in the parameterization:

Table 5-86

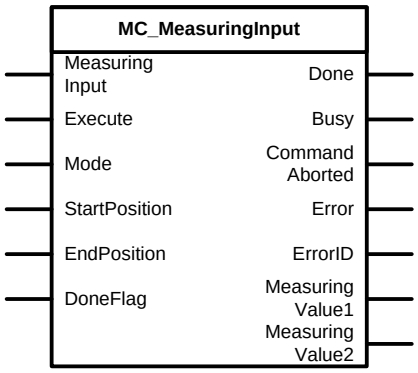
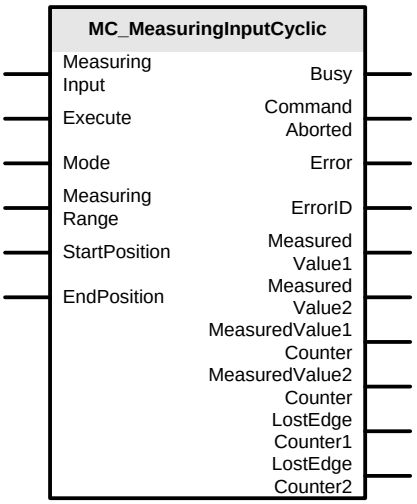
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	Measurement of the next rising edge. **Default setting**
1	Measurement of the next rising edge. **Default setting**	Measurement of the next falling edge.
2	Measurement of the next falling edge.	Measurement of the next two edges.
3	Measurement of the next two edges, starting with the rising edge.	Measurement of the next two edges, starting with the rising edge.
4	Measurement of the next two edges, starting with the falling edge.	Measurement of the next two edges, starting with the falling edge.
5	Cancel current measurement.	[Not available]

5.2.43 MC_MeasuringInput / MC_MeasuringInputCyclic

This technology function is available in ...

This technology function starts a cyclic measurement job for a measuring input.

Table 5-87 MC_MeasuringInput / MC_MeasuringInputCyclic

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Measuring Input	INT	Cam Track technology object	TO_ Measuring Input	Measuring Input
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Mode	INT	Mode selection	DINT	Mode
		Acquisition of measured values	BOOL	Measuring Range
StartPosition	REAL	Start position of measuring range	LREAL	StartPosition
EndPosition	REAL	End position of measuring range	LREAL	EndPosition
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

Output parameters (OUT)

Parameter	Data type	Function	Data type	Parameter
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted

CPU31xT			S7-1500(T)	
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
Measuring Value1	REAL	Measured value 1	LREAL	Measured Value1
Measuring Value2	REAL	Measured value 2	LREAL	Measured Value2
		Count for measured value 1	UDINT	Measured Value1 Counter
		Count for measured value 2	UDINT	Measured Value2 Counter
		Count for lost edges for measured value 1	UDINT	LostEdge Counter1
		Count for lost edges for measured value 2	UDINT	LostEdge Counter2

The following parameters involve functional differences or differences in the parameterization:

Table 5-88

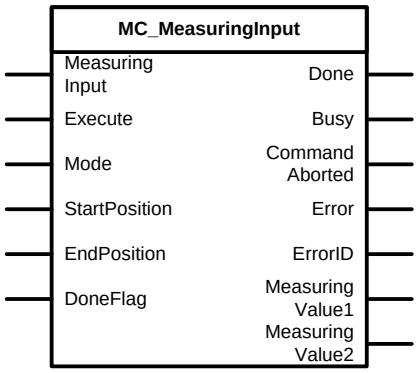
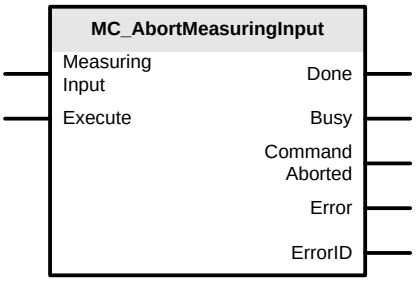
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	Measurement of rising edges. **Default setting**
1	Measurement of the next rising edge. **Default setting**	Measurement of falling edges.
2	Measurement of the next falling edge.	Measurement of both edges.
3	Measurement of the next two edges, starting with the rising edge.	[Not available]
4	Measurement of the next two edges, starting with the falling edge.	[Not available]
5	Cancel current measurement.	[Not available]

5.2.44 MC_MeasuringInput / MC_AbortMeasuringInput

This technology function is available in ...

This technology function stops a current measurement job for a measuring input.

Table 5-89 MC_MeasuringInput / MC_AbortMeasuringInput

CPU31xT	S7-1500(T)
	

Input parameters (IN)

Parameter	Data type	Function	Data type	Parameter
Measuring Input	INT	Cam Track technology object	TO_ Measuring Input	Measuring Input
Execute	BOOL	Start technology function with rising edge	BOOL	Execute
Mode	INT	Mode selection		
StartPosition	REAL	Start position of measuring range		
EndPosition	REAL	End position of measuring range		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		

Output parameters (OUT)

Parameter	Data type	Function	Data type	Parameter
Done	BOOL	The job is complete	BOOL	Done
Busy	BOOL	The job is being processed	BOOL	Busy
Command Aborted	BOOL	The job was aborted by another job	BOOL	Command Aborted
Error	BOOL	An error occurred while processing the job	BOOL	Error
ErrorID	WORD	Error ID	WORD	ErrorID
Measuring Value1	REAL	Measured value 1		
Measuring Value2	REAL	Measured value 2		

The following parameters involve functional differences or differences in the parameterization:

Table 5-90

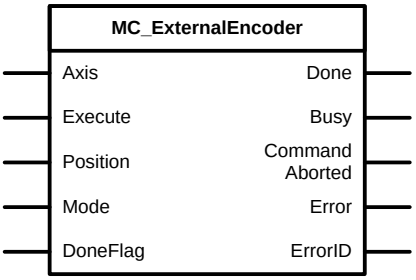
Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	[Not available]	
1	Measurement of the next rising edge. **Default setting**	
2	Measurement of the next falling edge.	
3	Measurement of the next two edges, starting with the rising edge.	
4	Measurement of the next two edges, starting with the falling edge.	
5	Cancel current measurement.	

5.2.45 MC_ExternalEncoder

This technology function is available in ...

This technology function enables / disables an external encoder.

Table 5-91 MC_ExternalEncoder

CPU31xT	S7-1500(T)
	**Note:** For the SIMATIC S7-1500T, the MC_Power and MC_Home technology functions must be applied to the external encoder to influence an external encoder.

Input parameters (IN)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Axis	INT	External Encoder technology object		
Execute	BOOL	Start technology function with rising edge		
Position	REAL	Setting value for external encoder position		
Mode	INT	Mode selection		
DoneFlag	INT	Flag for central evaluation of the status of the technology functions		
Output parameters (OUT)				
Parameter	**Data type**	**Function**	**Data type**	**Parameter**
Done	BOOL	The job is complete		
Busy	BOOL	The job is being processed		
Command Aborted	BOOL	The job was aborted by another job		
Error	BOOL	An error occurred while processing the job		
ErrorID	WORD	Error ID		

The following parameters involve functional differences or differences in the parameterization:

Table 5-92

Mode input parameter		
Value	**Meaning in the CPU31xT**	**Meaning in the S7-1500(T)**
0	Disable external encoder.	
1	Enable external encoder. **Default setting**	
2	Direct homing of the external encoder using the Position parameter.	
3	Passive homing of the external encoder according to the default settings of the technology object. The setting of the Position parameter is used as the home position.	
4	Direct homing of the external encoder using the settings of the technology object.	
5	Passive homing of the external encoder according to the default settings of the technology object.	
6	Absolute encoder adjustment. The setting of the Position parameter is used as the home position.	
7	Actual value correction considering the Position parameter.	

Note For the SIMATIC S7-1500T, the MC_Power and MC_Home technology functions must be applied to the external encoder to influence an external encoder.

6 Links & Literature

Table 6-1

	Topic	Title
\1\	Siemens Industry Online Support	http://support.industry.siemens.com
\2\	Download page of the entry	https://support.industry.siemens.com/cs/ww/en/view/109743136
\3\	Block library	Blocks for controlling the SINAMICS infeed modules (ALM) from the SIMATIC S7-1500(T) "DriveLib" block library https://support.industry.siemens.com/cs/en/en/view/109475044
\4\	Block library	More blocks for connecting a drive to the SIMATIC S7-1500 / S7-1500T. "DriveLib" block library https://support.industry.siemens.com/cs/en/en/view/109475044
\5\	Block library	Blocks for parameter access to SINAMICS drives that are also suitable for multi-axis drives. "LAcycCom" block library https://support.industry.siemens.com/cs/en/en/109479553
\6\	Block library	Creating cams at runtime in the user program of the SIMATIC S7-1500T "LCamHdl" block library https://support.industry.siemens.com/cs/ww/en/view/105644659
\7\	Application example	"Verwendung der Bausteine MC_PreServo und MC_PostServo" https://support.industry.siemens.com/cs/ww/en/view/109741575
\8\	Application example	"Circular Motion on the Basis of Cam Disks "MoveCircle2D"" https://support.industry.siemens.com/cs/ww/en/view/109742306
\9\	Application example	"Die Technologieobjekte (TOs) der SIMATIC S7-1500(T)" https://support.industry.siemens.com/cs/ww/en/view/Entry ID
\10\	Application example	"Time-based IO with the SIMATIC S7-1500 and TM Timer" https://support.industry.siemens.com/cs/ww/en/view/109738186
\11\	FAQ	"Isochronous mode with PROFINET – an example with SIMATIC S7-1500" https://support.industry.siemens.com/cs/ww/en/view/109480489
\12\	FAQ	"How Can You Integrate a Drive into the TIA Portal via the Device Master File (GSD)?" https://support.industry.siemens.com/cs/ww/en/view/73257075
\13\	Tools	TIA Selection Tool Selection assistance and configurator for automation technology http://w3.siemens.com/mcms/topics/en/simatic/tia-selection-tool/pages/tab.aspx
\14\	Tools	SIZER for Siemens Drives Configuration tool for Siemens drives https://support.industry.siemens.com/cs/ww/en/view/54992004

7 History

Table 7-1

Version	Date	Modifications
V1.0	01/2017	First version
V1.1	10/2017	Correction of Table 2-2