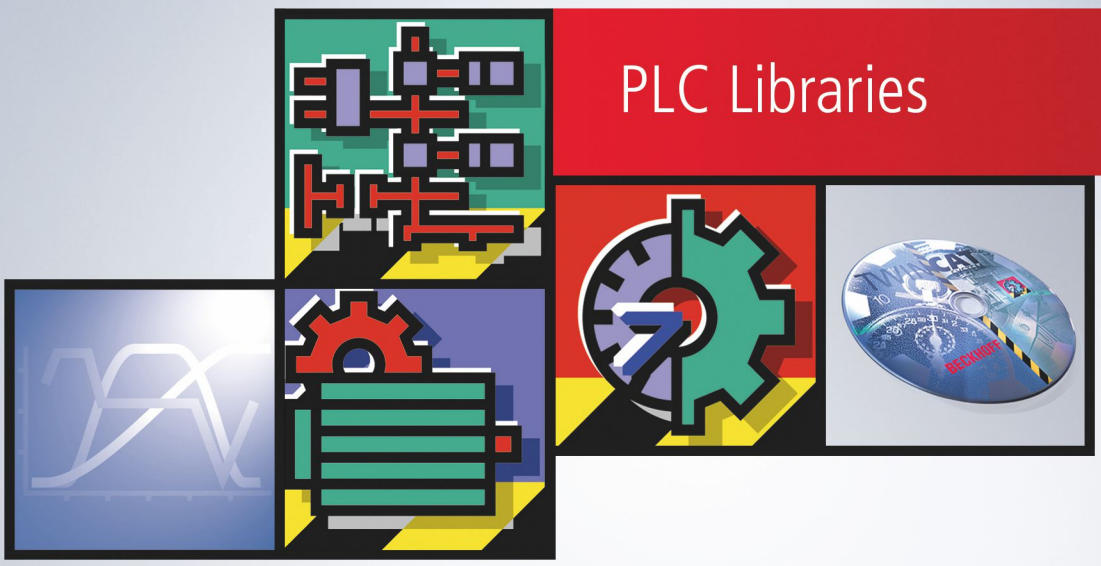
BECKHOFF New Automation Technology

Manual | EN

PLC Library: TcNC

TwinCAT 2 | TX1200, PlcNc, TcNcUtilities



2025-07-01 | Version: 1.1

Table of contents

1 Foreword

1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.
The documentation and the following notes and explanations must be complied with when installing and commissioning the components.
The trained specialists must always use the current valid documentation.

The trained specialists must ensure that the application and use of the products described is in line with all safety requirements, including all relevant laws, regulations, guidelines, and standards.

Disclaimer

The documentation has been compiled with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without notice.
Claims to modify products that have already been supplied may not be made on the basis of the data, diagrams, and descriptions in this documentation.

Trademarks

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar®, and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.
If third parties make use of the designations or trademarks contained in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

Copyright

Third-party trademarks

Trademarks of third parties may be used in this documentation. You can find the trademark notices here: https://www.beckhoff.com/trademarks.

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation, and drive technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER

Hazard with high risk of death or serious injury.

⚠ WARNING

Hazard with medium risk of death or serious injury.

⚠ CAUTION

There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE

The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Overview

The PLC and the NC communicate in two ways:

- Cyclical interface from the PLC to NC [▶ 63] (enables, overrides etc.) NC to the PLC [▶ 54] (actual values, statuses etc.): Exchange of the necessary information in each PLC cycle by way of the cyclical process image.
- Function blocks Function blocks (referred to below as NC-FBs) in the sense of IEC1131 are provided, each of which contains several related functions. The NC-FBs are implemented as firmware blocks. In other words, they are part of the controller software, and their behavior has a fixed definition. The blocks have inputs and outputs whose data types are all elementary IEC1131 data types (i.e., no derived types).

The NC-FBs are utilized through the formation of instances:

The PLC programmer creates a variable (an instance) of the desired block whenever required and can then supply parameters to it when it is called.

Communication is provided between the PLC and NC axes, and between the PLC and the controller channels. In both cases there is a direct exchange of data on the level of the process images. This data must always be available. Functions are called by associated blocks which implement data exchange through ADS services.

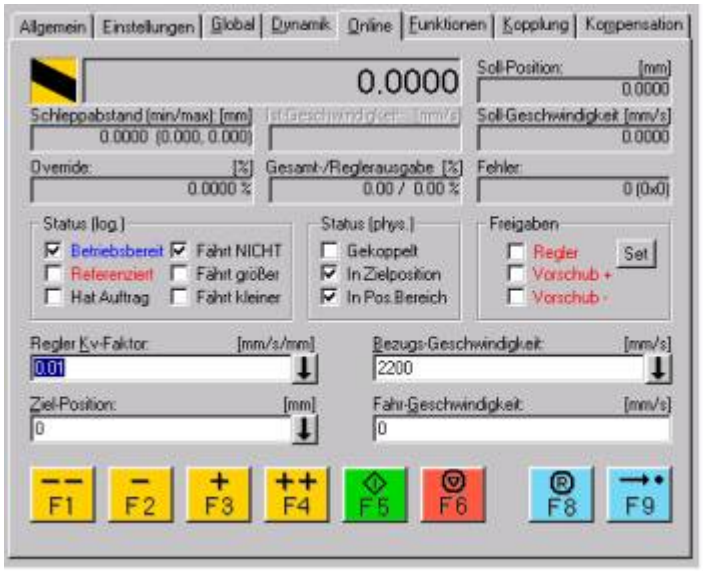
All the available PLC languages can be used to access these blocks.

The NC architecture of the equipment is represented in the System Manager: entries are made for the NC channels and axes, and, optionally, other special groups (interpolation). These elements must then be parameterized. This makes it possible to add to the tree in order to insert a new NC axis or a new NC channel into an existing configuration (the NC is freely scalable NC). It is possible to drive all the existing axes individually, or to place them into groups of two or three to permit interpolating processes. It is also, for example, possible to couple individual axes as slaves to any other axis (the master).

The maximum number of NC elements (axes, groups or channels) that one computer can support depends on the available computing power. The structure of the software limits the number of axes to a maximum of 255.

Axes are commissioned with the aid of the appropriate dialogues in the NC configuration area of the System Managers.

Example



You will find further information either in the documentation or in the System Manager's online help.

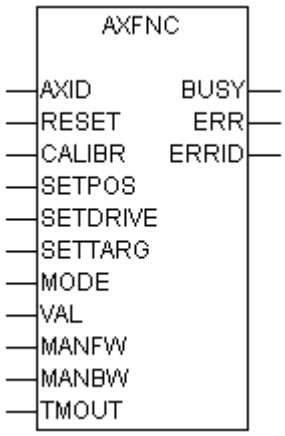
Library functions and function blocks

The PLC library collects useful functions and function blocks for programming axis controllers. A library contains blocks that can be used universally. Blocks for special NC functions are found in additional specialized libraries.

3 Function blocks

3.1 NC Axis interface

3.1.1 AXFNC (axis functions)



Functions that are required when the equipment is started up after errors, to prepare an axis for normal operation, are assembled in this block. Only one function from this block may be active at any one time!

If a rising or falling edge is presented to more than one input the command with highest priority is sent to the axis.

The prorities of the function block inputs:

- Falling edge to SETDRIVE (highest priority);
- Falling edge to MANFW;
- Falling edge to MANBW;
- Rising edge to SETDRIVE;
- Rising edge to MANFW;
- Rising edge to MANBW;
- Rising edge to RESET;
- Rising edge to CALIBR;
- Rising edge to SETPOS;
- Rising edge to SETTARG (lowest priority);

Inputs

The block has the following inputs:

Input	Data type	Description
AXID	INT	ID of the axis (see System Manager)
RESET	BOOL	Places the axis into a basic error-free state, as far as that may be logically and physically possible. It must not be triggered during movement, as this would abruptly halt the axis.
CALIBR	BOOL	A rising edge at this input will initiate referencing of the selected axis.
SETPOS	BOOL	Sets the actual value of an axis to the value supplied in 'Val', in accordance with the setting type given in 'Mode'.
SETDRIVE	BOOL	A rising edge will result in the output value specified in 'Val' (usually a voltage) being sent to the controller assigned to the axis, in accordance with the specifications of the mode parameter. This means that the axis

Input	Data type	Description
		no longer operates under the influence of the controller. The controller enable must be set continuously during this function is active. The output is maintained until a falling edge occurs at 'SetDrive'.
SETTARG	BOOL	Changes the destination of an axis while it is in operation. The new target position must be specified in 'Val'. The target position will be interpreted in accordance with the specification in the 'Mode' parameter.
MODE	DWORD	Mode – see below
VAL	LREAL	Contains the value required by the function being carried out: **SetPos:** new set value (consider mode!) **SetDrive:** value for direct drive output (consider mode!) **SetTargP:** value for new target position (consider mode!) **ManFw:** value for the forwards manual movement speed **ManBw:**: value for the reverse manual movement speed
MANFW	BOOL	If the axis starts continuous movement at a rising edge in the direction of positive encoder counts, it will be stopped by a falling edge. The software limit switches are effective unless they have been deselected by the axis configuration. The desired velocity must be provided in 'Val'. Alternatively, the axis could be stopped by an instance of the AXACT or AXACTEX blocks.
MANBW	BOOL	If the axis starts continuous movement at a rising edge in the direction of negative encoder counts, it will be stopped by a falling edge. The software limit switches are effective unless they have been deselected by the axis configuration. The desired velocity must be provided in 'Val'. Alternatively, the axis could be stopped by an instance of the AXACT or AXACTEX blocks.
TMOUT	TIME	ADS Timeout-Delay

Output	Data type	Description
BUSY	BOOL	This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the ‘Timeout’ input. While Busy = TRUE, no new command will be accepted at the inputs. Please note that it is not the execution of the service but its acceptance whose time is monitored.
ERR	BOOL	This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in ‘ErrorId’. If the block has a timeout error, ‘Error’ is TRUE and ‘ErrorId’ is 1861 (hexadecimal 0x745). Is reset to FALSE by the execution of a command at the inputs.
ERRID	UDINT	Contains the command-specific error code of the most recently executed command. Is reset to 0 by the execution of an instruction at the inputs. The error numbers in ErrId can be consulted in the ADS error documentation

MODE: depending on the function being carried out, mode specifies the effect of the function more precisely:

Define	Set types for the actual value
1	Absolute
2	Relative (± travel region)
3	Reserved
4	Reserved
5	Modulo (can also be larger than the modulo factor)

Define	Drive output
1	Drive output in % in the range [-100%, 100%] of the maximum range that can be set
2	Drive output as an absolute output speed (e.g. mm/s)

Define	New target position
1	Absolute
2	Relative (± travel region)
3	Reserved
4	Reserved
5	Modulo (can also be larger than the modulo factor)

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.2 AXACT (axis actions)

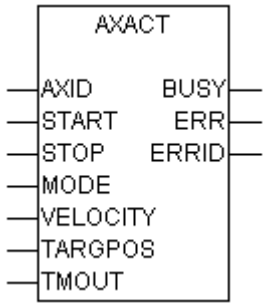


Fig. 1: axact

The STOP input has highier priority as the START input. If a rising edge is presented to both the 'START' and 'STOP' inputs at the same time, the Stop command will be sent to the axis.

The block has the following inputs:

Input	Data type	Description
AXID	INT	Axis ID
START	BOOL	A rising edge at this boolean input sends a start command to the axis. As soon as the block's BUSY output goes FALSE, the bit in the cyclical process image for 'axis has transport instruction' is set, and the NC has accepted the start command for the axis; however, this means neither that the axis has already started physical movement, nor that it has arrived at its target. Only when the bit for 'axis has transport instruction' has returned to zero, without any axis error having occurred in the meantime, has the logical positioning of the axis been completed. If a target position window has been set, it is possible for reaching the target position window to be evaluated as reaching the physical position.
STOP	BOOL	A rising edge at this boolean input sends a stop command to the axis. Only when the 'axis has transport instruction' bit in the cyclical process image is set to zero has the logical positioning of the axis been completed.

Input	Data type	Description
MODE	DWORD	Start mode: absolute, continuous, modulo – see below. **Remark**: Not all of the start modes are supported with all axis types !
VELOCITY	LREAL	The parameter contains the required transport speed for a following transport instruction, e.g. mm/s.
TARGPOS	LREAL	Target position in physical magnitudes, e.g. mm, degrees
TMOUT	TIME	ADS Timeout-Delay

Output	Data type	Description
BUSY	BOOL	This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the 'Timeout' input. While Busy = TRUE, no new command will be accepted at the inputs. Please note that it is not the execution of the service but its acceptance whose time is monitored.
ERR	BOOL	This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in 'ErrorId'. If the block has a timeout error, 'Error' is TRUE and 'ErrorId' is 1861 (hexadecimal 0x745). Is reset to FALSE by the execution of a command at the inputs.
ERRID	UDINT	Contains the command-specific error code of the most recently executed command. Is reset to 0 by the execution of an instruction at the inputs. The error numbers in ErrId can be consulted in the ADS error documentation

Mode: start types (1D)

supported Axistypes	Define	Set types for the actual value
all	1	Absolute (± travel region)
all	2	Relative (± travel region)
all	3	Continuous positive
all	4	Continuous negative
all	5	Modulo (± travel region) (can also be larger than the modulo factor)
only Servo Axis	272	Continuous positive with slow manual speed
only Servo Axis	528	Continuous positive with fast manual speed
only Servo Axis	784	Continuous positive with rapid speed
only Servo Axis	288	Relative positive by pulse width
only Servo Axis	544	Relative positive at 1/1000
only Servo Axis	800	Relative positive at 1/100
only Servo Axis	1056	Relative positive at 1/10
only Servo Axis	1312	Relative positive at 1/1
only Servo Axis	273	Continuous negative with slow manual speed
only Servo Axis	529	Continuous negative with fast manual speed
only Servo Axis	785	Continuous negative with rapid speed
only Servo Axis	289	Relative negative by pulse width
only Servo Axis	545	Relative negative at 1/1000
only Servo Axis	801	Relative negative at 1/100
only Servo Axis	1057	Relative negative at 1/10
only Servo Axis	1313	Relative negative at 1/1

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.3 AXACTEX (extended axis actions)

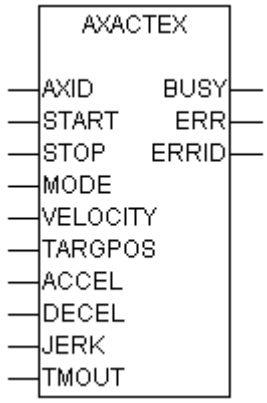


Fig. 2: axactex

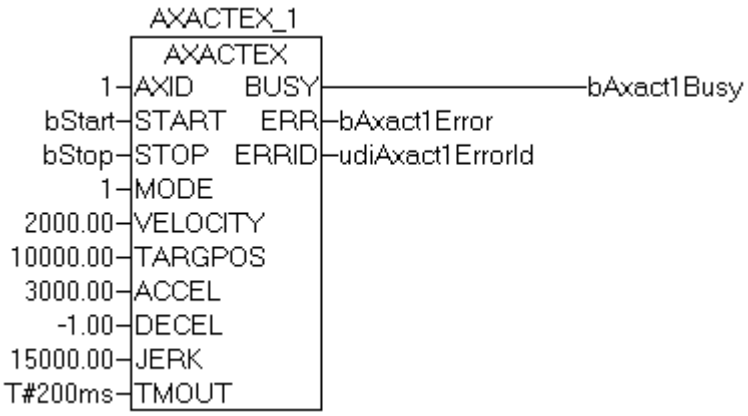
The STOP input has higher priority as the START input. If a rising edge is presented to both the 'START' and 'STOP' inputs at the same time, the Stop command will be sent to the axis.

The block has the following inputs:

Input	Data type	Description
AXID	INT	Axis ID
START	BOOL	A rising edge at this boolean input sends a start command to the axis. As soon as the block's BUSY output goes FALSE, the bit in the cyclical process image for 'axis has transport instruction' is set, and the NC has accepted the start command for the axis; however, this means neither that the axis has already started physical movement, nor that it has arrived at its target. Only when the bit for 'axis has transport instruction' has returned to zero, without any axis error having occurred in the meantime, has the logical positioning of the axis been completed. If a target position window has been set, it is possible for reaching the target position window to be evaluated as reaching the physical position.
STOP	BOOL	A rising edge at this boolean input sends a stop command to the axis. Only when the 'axis has transport instruction' bit in the cyclical process image is set to zero has the logical positioning of the axis been completed.
MODE	DWORD	Start mode: absolute, continuous, modulo – see below
VELOCITY	LREAL	The parameter contains the required transport speed for a following transport instruction, e.g. mm/s.
ACCEL	LREAL	Acceleration, e.g. mm/s²
DECEL	LREAL	Deceleration, e.g. mm/s²
JERK	LREAL	Jerk e.g. mm/s³
TARGPOS	LREAL	Target position in physical magnitudes, e.g. mm, degrees
TMOUT	TIME	ADS Timeout-Delay

Output	Data type	Description
BUSY	BOOL	This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the ‘Timeout’ input. While Busy = TRUE, no new command will be accepted at the inputs. Please note that it is not the execution of the service but its acceptance whose time is monitored.
ERR	BOOL	This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in ‘ErrorId’. If the block has a timeout error, ‘Error’ is TRUE and ‘ErrorId’ is 1861 (hexadecimal 0x745). Is reset to FALSE by the execution of a command at the inputs.
ERRID	UDINT	Contains the command-specific error code of the most recently executed command. Is reset to 0 by the execution of an instruction at the inputs. The error numbers in ErrId can be consulted in the ADS error documentation

Example

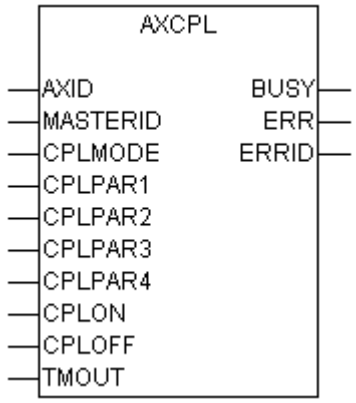


The diagram shows "AXACTEX_1" as an instance of the AXACTEX block, with which axis no. 1 can be started and stopped. The target position is at 10000.00 mm, and the speed is 2000 mm/s. Additionally, the acceleration (3000 mm/s2) and the jerk (15000 m/s3) are specified with the aid of this block, the standard value being taken from the axis configuration for the deceleration (special identifier "-1").

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.4 AXCPL (axis coupling)



This block is required in order to couple any servo-axis as a slave to any other servo axis. The coupling can take place while the axes are stationary, or while the master axis is moving, depending on the type of coupling. Once a slave axis has been coupled it must no longer be addressed by movement commands, since it has lost its independence until such time as it is again uncoupled. The simplest type of coupling is linear coupling with a fixed gear ratio (an electronic gear). This kind of coupling can also have a negative gear ratio. The COPLOFF input has higher priority as the COPLON input. If a rising edge is presented to both the 'COPLOFF' and 'COPLON' inputs at the same time, the deactivate coupling command will be sent to the axis.

NOTICE

Migrating to TwinCAT 2.11

If the Set Point Generator Type is set to '7 Phases (optimized)', the slave axis will reduce its acceleration to zero after it is being decoupled and it will then continue moving endless at constant velocity.
The decoupled axis will not be positioned to any target position. The behavior is comparable to a move commanded by MC_MoveVelocity.
With TwinCAT 2.10 the set point generator type is selectable.
From TwinCAT 2.11 the setting is fixed to '7 Phases (optimized)'.
If a project is upgraded from TwinCAT 2.10 to TwinCAT 2.11, the behavior will be as described here. After updating an existing application to TwinCAT 2.11, it might be necessary to adapt the PLC program.

The block has the following inputs:

Input	Data type	Description
AXID	INT	Axis ID
MASTERID	INT	ID of the master card
CPLMODE	INT	Coupling type
CPLPAR1	LREAL	Coupling parameter 1
CPLPAR2	LREAL	Coupling parameter 2
CPLPAR3	LREAL	Coupling parameter 3
CPLPAR4	LREAL	Coupling parameter 4
CPLON	BOOL	Activate coupling
CPLOFF	BOOL	Deactivate coupling
TMOUT	TIME	ADS Timeout-Delay

Output	Data type	Description
BUSY	BOOL	This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the ‘Timeout’ input. While Busy = TRUE, no new command will be accepted at the inputs. Please note that it is not the execution of the service but its acceptance whose time is monitored.
ERR	BOOL	This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in ‘ErrorId’. If the block has a timeout error, ‘Error’ is TRUE and ‘ErrorId’ is 1861 (hexadecimal 0x745). Is reset to FALSE by the execution of a command at the inputs.
ERRID	UDINT	Contains the command-specific error code of the most recently executed command. Is reset to 0 by the execution of an instruction at the inputs. The error numbers in ErrId can be consulted in the ADS error documentation

CPLMODE

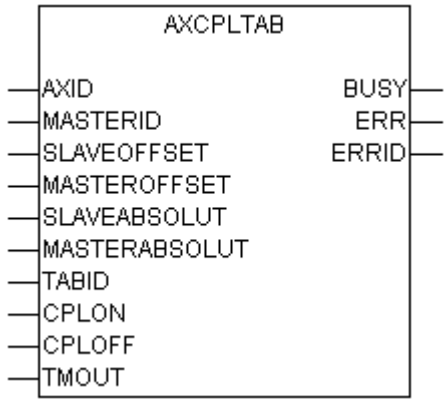
Define	Coupling type	
1	**"Linear coupling"** Linear coupling with a fixed gear ratio. The ratio can be positive or negative, but the value must not be zero. It is also possible for the gear ratio of an existing coupling to be changed online. However, since this change will take effect abruptly, the gear ratio must only be altered by very small amounts in the fractional part of its value (mistuning).	
	‘CplPara1’:	gearing factor
	‘CplPara2’:	reserved
	‘CplPara3’:	reserved
	‘CplPara4’:	reserved
2	**”Diagonal coupling: Linear"** Weak diagonal synchronised coupling (”flying saw") with linear velocity curve. Weak: master travel during the slave acceleration phase = 2*slave travel during the slave acceleration phase.	
	‘CplPara1’:	Abs. synchronous position of the master [mm]
	‘CplPara2’:	Abs. synchronous position of the slave [mm]
	‘CplPara3’:	Angle of inclination of the diagonal saw with reference to the orthogonal line to the master route [degrees]. The angle must be within the range greater than 0 degrees and smaller than or equal to 90 degrees.
	‘CplPara4’:	Gearing factor (a value of 1.0 will be used if 0 is specified)
3	**”Diagonal coupling: jerk-restricted"** Weak diagonal synchronised coupling (”flying saw") with jerk-restricted velocity curve. Weak: master travel during the slave acceleration phase = 2*slave travel during the slave acceleration phase.	
	CplPara1’:	Abs. synchronous position of the master [mm]
	CplPara2’:	Abs. synchronous position of the slave [mm]
	CplPara3’:	Angle of inclination of the diagonal saw with reference to the orthogonal line to the master route [degrees]. The angle must be within the range greater than 0 degrees and smaller than or equal to 90 degrees.
	CplPara4’:	Gearing factor (a value of 1.0 will be used if 0 is specified)
12	**"Flying saw modulo coupling (acceleration-limited)"** This coupling type was specially developed for modulo axes (periodic continuous motion, where the period may or may not be 360.0 degrees). The special feature of this slave type is that, for the first time, the slave axis has a variety of sub-operating modes. Operating modes can be changed at any time via external requests. The slave axis responds with an intermediate phase, which means synchronization to the requested operating mode. This synchronization is implemented as a linear path control, i.e. across a broad range it is unaffected by the alterations in the master dynamics. For this modulo coupling, synchronization means that, a fixed orientation is produced over the shortest path length (± 1/2 period), based on the modulo period.	
	‘CplPara1’:	Gearing factor (mandatory at the moment with 1.0 presupposed!)
	CplPara2':	Maximum slave acceleration in percent based on minimum the slave acceleration of acceleration and deceleration. Value range: >0.0 to ... 1.0
	CplPara3':	Master axis ID 2 for master coupling
	CplPara4':	Optional table-ID (table types: equidistant cyclic and not equidistant cyclic) This function is not approved yet !
15	**"Linear coupling with cyclically variable gearing factor"** For a cyclic slave axis, the gearing factor is specified by the PLC via the axis interface (see variable "fAxisModeLReal" in the structure PLCTONC_AXLESTRUCT) and can be changed during each PLC cycle. This change is then automatically transferred from the PLC to the NC via the cyclic data exchange of the axis interface. To avoid extreme step	

Define	Coupling type	
	changes of the gearing factor and thus step changes in acceleration, the acceleration of the slave resulting from a change in gearing factor can be limited via a limit parameter p_a. The limit parameter is specified during axis coupling via the first coupling parameter.	
	CplPara1':	corresponds indirectly, i.e. relative to a maximum master velocity, to a maximum permitted acceleration ($p_a = a_{SlaveMax} / v_{MasterMax}$). The limit parameter p_a corresponds to the reciprocal value of the run-up time $t_H = 1 / p_a$.
	CplPara2':	reserved
	CplPara3':	reserved
	CplPara4':	reserved
16	**"Bi-linear coupling"** Bi-linear coupling is comparable to linear coupling (coupling mode: 1), although two linear couplings are active simultaneously via gearing factor 1 and gearing factor 2 (for additional master axis ID 2). This mode may be used, for example, for tapping/thread cutting, where linear feed must be coordinated with rotary motion. *Remark:* If the main master axis 1 is not logically in motion, no slave target values are calculated, independent of the motion phase of master axis 2. In this context it should also be noted that an axis travelling with velocity override 0% is logically in motion!	
	‘CplPara1’:	Gearing factor 1 (gearing factor)
	CplPara2':	Gearing factor 2 (incidental gearing factor)
	CplPara3':	Master axis ID 2 for the gearing factor 2
	CplPara4':	reserved
18	**"Coupling with constant surface velocity and cyclically variable gearing factor"** This type contains the mathematical calculation for coupling between a rotary slave axis and a translatory master axis. Its purpose is to generate and re-adjust a constant surface velocity (peripheral speed), relative to the master axis, for the rotary calibrated and controlled slave axis, depending on its drum diameter. Via a second encoder (auxiliary encoder) that has to be configured for the slave axis in the system manager, the drum radius of this slave axis is automatically evaluated during each NC-SAF cycle (SAF = "Satzausführungstask", block execution task) and used for the calculation. This radius must never have the value 0.0 mm, since otherwise a calculation is no longer possible. Additionally, a gearing factor g(t) (see variable "fAxisModeLReal" in the structure PLCTONC_AXLESTRUCT) is specified via the cyclic axis interface (see "linear coupling with cyclical variable gearing factor"), which in the most trivial case may have the constant value of 1.0 (no further influencing). Therefore, if the slave axis is calibrated to degrees via its main encoder and to mm via its auxiliary encoder (radius detection r(t)), and the master as translatory axis in mm, the slave velocity can be calculated according to the following formula: $v_{Slave} = 360° / (2PI * r(t))* g(t)* v_{Master}$.	
	‘CplPara1’:	corresponds indirectly, i.e. relative to a maximum master velocity, to a maximum permitted acceleration ($p_a = a_{SlaveMax} / v_{MasterMax}$). The limit parameter p_a corresponds to the reciprocal value of the run-up time $t_H = 1 / p_a$.
	‘CplPara2’:	reserved
	'CplPara3':	reserved
	'CplPara4':	reserved

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.5 AXCPLTAB (table-based axis coupling)



This block is required in order to couple any servo-axis as a slave to any other servo axis. In this case the coupling is implemented in the form of a table, and this in turn consists of a specifiable number of support points (location curve).

Four different types of table are distinguished:

- equidistant linear tables,
- equidistant cyclical tables,
- monotonic linear tables and
- monotonic cyclical tables.

This kind of coupling can only be initiated while the axes are stationary. Once a slave axis has been coupled it must no longer be addressed by movement commands, since it has lost its independence until such time as it is again uncoupled. The COPLOFF input has highier priority as the COPLON input. If a rising edge is presented to both the 'COPLOFF' and 'COPLON' inputs at the same time, the deactivate coupling command will be sent to the axis.

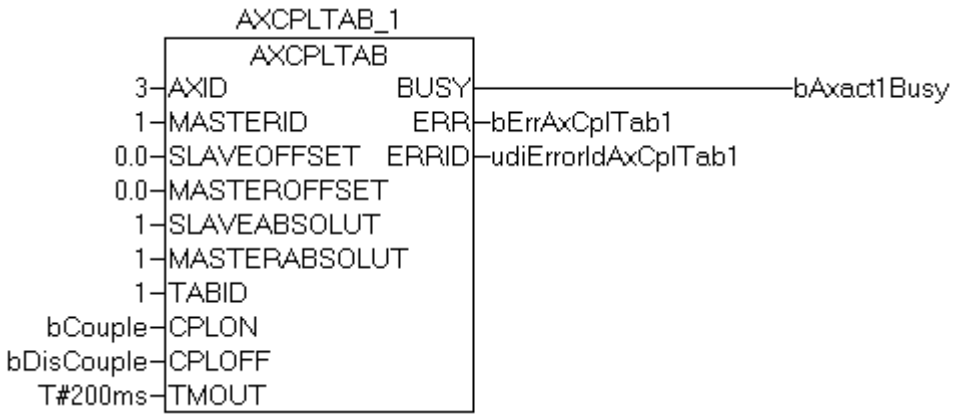
The block has the following inputs:

Input	Data type	Description
AXID	UINT	Axis ID
MASTERID	UINT	Id of the master axis
SLAVEOFFSET	LREAL	A positional offset, 0.0 in the standard case, can be given for the interpretation of the slave position in the coupling table. The physical unit corresponds to the basic length unit associated with the axis (e.g. mm).
MASTEROFFSET	LREAL	A positional offset, 0.0 in the standard case, can be given for the interpretation of the master position in the coupling table. The physical unit corresponds to the basic length unit associated with the axis (e.g. mm).
SLAVEABSOLUT	UDINT	Interpretation of the slave table positions as absolute or relative positions 1 : Absolute 0 : Relative
MASTERABSOLUT	UDINT	Interpretation of the master table positions as absolute or relative positions 1 : Absolute 0 : Relative
TABID	UDINT	The table ID which is valid across the whole system is given here. Coupling a table assumes that the table already exists, that it has the appropriate dimensions, and that it has been filled with valid values.

Input	Data type	Description
		A distinction is drawn here between four different types of table: • equidistant linear tables (Type 1) • equidistant cyclic tables (Type 2) • monotonic linear tables (Type 3) • monotonic cyclical tables (Type 4)
CPLON	BOOL	Coupling on
CPLOFF	BOOL	Coupling off
TMOUT	TIME	ADS Timeout-Delay

Output	Data type	Description
BUSY	BOOL	This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the ‘Timeout’ input. While Busy = TRUE, no new command will be accepted at the inputs. Please note that it is not the execution of the service but its acceptance whose time is monitored.
ERR	BOOL	This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in ‘ErrorId’. If the block has a timeout error, ‘Error’ is TRUE and ‘ErrorId’ is 1861 (hexadecimal 0x745). Is reset to FALSE by the execution of a command at the inputs.
ERRID	UDINT	Contains the command-specific error code of the most recently executed command. Is reset to 0 by the execution of an instruction at the inputs. The error numbers in ErrId can be consulted in the ADS error documentation

Example



The diagram shows "AXCPLTAB_1", an instance of the AXCPLTAB block, by means of which the axis with ID 3 can be coupled to or uncoupled from the axis with ID 1 (the master axis) (at a rising edge of the variables "bCouple" or "bDiscouple"). The set position values of the master and of the slave are taken from a binary table which is present. An equidistant linear table is used in this example (TabId=1). In this example the positional offsets of the master and slave are pre-defined as 0.0. An example follows of an equidistant linear table in ASCII format, with 81 lines (master position from 0.0 to 80.0 mm or degrees; slave position from 0.0 to 279.72 mm or degrees).

81	2
+0.00000	+0.00000
+1.00000	+0.10783
+2.00000	+0.43114
...	...

+76.00000	+277.99809
+77.00000	+278.75055
+78.00000	+279.28886
+79.00000	+279.61217
+80.00000	+279.72000

Table type:

Equidistant linear table with 81 pairs of values (support points)

First line:

Number of lines (n=81) and number of columns (m=2)

Following lines:

n lines with the master position (absolute or relative) and the associated slave position (absolute or relative)

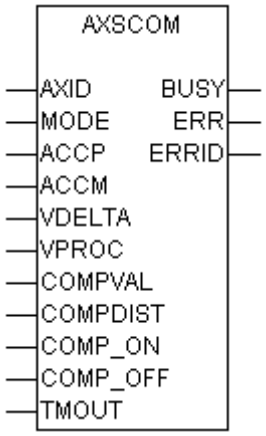
Remark:

Between the support points, a linear interpolation takes place in each SAF cycle of the axis. The table size and the way in which the discrete values are assigned to the position can be freely selected.

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.6 AXSCOM (axis section compensation)



This block is needed if a moving axis (master or slave) is to catch up or to fall back a specifiable distance over a given travel region. This is implemented by means of an increase or reduction of the speed for a limited period calculated by the axis positioning software. The COMP_OFF input has highier priority as the COMP_ON input. If a rising edge is presented to both the 'COMP_OFF' and 'COMP_ON' inputs at the same time, the compensation off command will be sent to the axis.

The block has the following inputs:

Input	Data type	Description
AXID	INT	Axis ID
MODE	UDINT	Compensation mode – see below
ACCP	LREAL	Max. acceleration
ACCM	LREAL	Max. deceleration
VDELTA	LREAL	Max. permitted change in speed

Input	Data type	Description
VPROC	LREAL	Base speed for the process
COMPVAL	LREAL	Compensation value to be caught up or dropped back
COMPDIST	LREAL	Compensation section available for the task
COMP_ON	BOOL	Compensation on
COMP_OFF	BOOL	Compensation off
TMOUT	TIME	ADS Timeout-Delay

Output	Data type	Description
BUSY	BOOL	This output remains TRUE until the block has executed a command, but at the longest for the duration supplied to the 'Timeout' input. While Busy = TRUE, no new command will be accepted at the inputs. Please note that it is not the execution of the service but its acceptance whose time is monitored.
ERR	BOOL	This output is switched to TRUE if an error occurs during the execution of a command. The command-specific error code is contained in 'ErrorId'. If the block has a timeout error, 'Error' is TRUE and 'ErrorId' is 1861 (hexadecimal 0x745). Is reset to FALSE by the execution of a command at the inputs.
ERRID	UDINT	Contains the command-specific error code of the most recently executed command. Is reset to 0 by the execution of an instruction at the inputs. The error numbers in ErrId can be consulted in the ADS error documentation

Mode

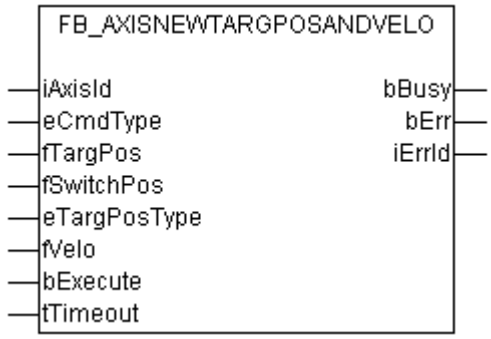
Define	Compensation profile
1	Trapezoidal speed profile

Warning: If the compensation cannot be fully executed within the required parameters, the start command is answered with the NC error code "0x4243". This response should be looked on as merely a warning since the compensation will still be carried out as far as may be possible within the specified limits.

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.7 FB_AxisNewTargPosAndVelo



The target position and velocity of the axis can be changed during the operation with the function block "FB_AixsNewTartAndVelo". With the Function AxisIsMoving [▶ 39] can be determined, if the axis is moving. If it does not move, this command creates an error. Furthermore the command should not be set very shortly before the positioning ends, because otherwise the axis could already be locked at the receiving of the command.

VAR_INPUT

```
VAR_INPUT
    iAxisId       : UINT;
    eCmdType      : E_CmdTypeNewTargPosAndVelo;
    fTargPos      : LREAL;
    fSwitchPos    : LREAL;
    eTargPosType  : E_TargPosType;
    fVelo         : LREAL;
    bExecute      : BOOL;
    tTimeout      : TIME;
END_VAR
```

iAxisId: Axis-ID of the axis

eCmdType: The type (E_CmdTypeNewTargPosAndVelo [▶ 65]) of the command defines, if target position, velocity or both values are changed during the current operation. Furthermore this parameter defines, if the change operates immediately or at the switch threshold fSwitchPos.

fTargPos: Target position of the current operation of the axis.

fSwitchPos: Optional switch threshold. When it is reached, the command acts.

eTargPosType: Type of the target position (E_TargPosType [▶ 66]). The relative positioning should not be used in this context. Otherwise, the target position would be dependent of the position that the command is activated at and therefore not exactly.

fVelo: New velocity, with it should be moved to the target position.

bExecute: The command is triggered by a rising edge at this input.

tTimeout: Timeout until the NC acknowledges the command.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy         : BOOL;
    bErr          : BOOL;
    iErrId        : UDINT;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until and acknowledgement is received.

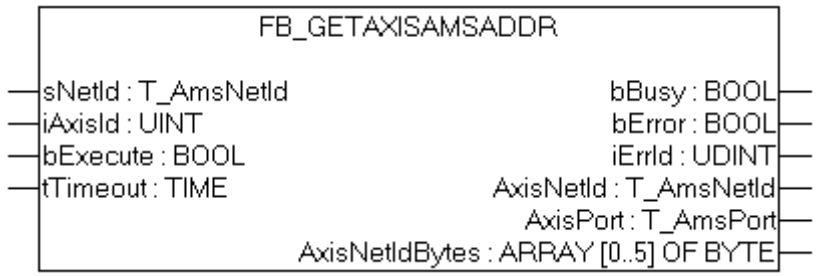
bErr: The output bErr is set if an error occurs as the command is being executed.

iErrId: Supplies the error number if the bErr output is set.

Requirements

Development Environment	Target System	PLC Libraries to include
TwinCAT v2.8	PC (i386)	TcNc.Lib

3.2 FB_GetAxisAmsAddr



This block is needed to read the ADS address (NetId and Port) of the IO hardware linked with the axis.

VAR_INPUT

```
VAR_INPUT
    sNetId    : T_AmsNetId;
    iAxisId   : UINT;
    bExecute  : BOOL;
    tTimeout  : TIME;
END_VAR
```

sNetId : : It is possible here to provide the AmsNetId of the TwinCAT computer on which the address information of the axis is to be read. If it is to be run on the local computer, an empty string can be entered.

iAxisId: Axis ID.

bExecute: The command is triggered by a rising edge at this input.

tTimeout: ADS Timeout-Delay

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy           : BOOL;
    bError          : BOOL;
    iErrId          : UDINT;
    AxisNetId       : T_AmsNetId;
    AxisPort        : T_AmsPort;
    AxisNetIdBytes  : ARRAY[0..5] OF BYTE;
    AxisChannel     : BYTE;
END_VAR
```

bBusy: When the function block is activated this output is set. It remains set until and acknowledgement is received.

bError: If an ADS error should occur during the execution of the command, then this output is set.

iErrId: Supplies the error number when the **bError** output is set.

AxisNetId: ADS NetId of the NC axis hardware.

AxisPort: ADS port number of the NC axis hardware.

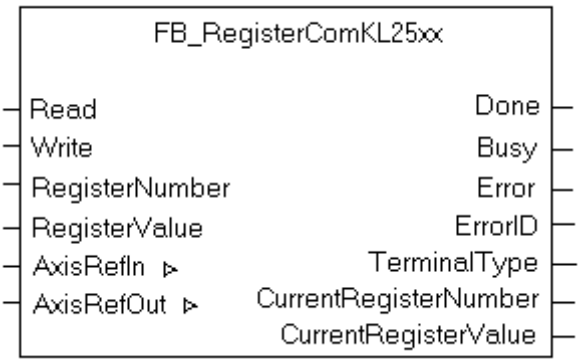
AxisNetIdBytes: ADS-NetId as byte array.

AxisChannel: Chanel number of the NC axis hardware (multi channel devices)

Requirements

Development environment	Target System	PLC Libraries to include
TwinCAT v2.8 Build > 746 TwinCAT v2.9 Build > 947	PC (i386)	TcNc.Lib

3.3 FB_RegisterComKL25xx



The function block FB_RegisterComKL25xx is used for register communication between PLC and the terminals KL2502, KL2521, KL2531, KL2541 and KL5001.

The function block before reading or writing from or into the register reads the AxisID, EncoderID, DriveID to which the terminal variables are mapped. Further the function block cancels or cuts the link between drive o/p and terminal so as inhibit the NC task access. Hence there is no communication between the drive output and terminal during the execution of this function block. This is then followed by register read or write. The NC task access is restored once the register communication is done (Drive ouput is activated).

Requirements:

It is necessary to map the terminal process data variables to the encoder and drive variables of the corresponding axis by selecting appropriate encoder and drive type. Based on mapping between IO terminals and NC variables, the terminals can be classified into two groups as KL2531 and KL2541 into one group and the other group consisting of the terminals KL2502 and KL2521.

The mapping between terminal and NC variables that is required for successful execution of this function block can be shown as under

KL2531/KL2541	
State	Axis_Drive.Inputs.Axis_Drive_In.nStatus1
Position	Axis_Enc.Inputs.Axis_Enc_In.nInData1.nInData1[0]
ExtStatus	Axis_Enc.Inputs.Axis_Enc_In.nStatus1
Control	Axis_Drive.Outputs.Axis_Drive_Out.nCtrl1
Velocity	Axis_Drive.Outputs.Axis_Drive_Out.nOutData2.nOutData2[0]
ExtCntrl	Axis_Enc.Outputs.Axis_Enc_Out.nCtrl1
KL2502/KL2521	
State	Axis_Enc.Inputs.Axis_Enc_In.nStatus1
DataIn	Axis_Enc.Inputs.Axis_Enc_In.nInData1.nInData1[0]
Control	Axis_Enc.Outputs.Axis_Enc_Out.nCtrl1
DataOut	Axis_Drive.Outputs.Axis_Drive_Out.nOutData1.nOutData1[0]

Note :

If a register communication cycle has been initiated by a rising edge on both *Read* and *Write* inputs, the function block first writes the given *RegisterValue* in the specified *RegisterNumber* and then reads the register value from the same *RegisterNumber* again. This situation is not considered as an error.

The register read and write is done from and to the EEPROM respectively. In case of writing into a register there is an exception that one cannot write into Manufacturer registers using this function block. This would lead to an error number 0x4B41.

VAR_INPUT

```
VAR_INPUT
    Read            : BOOL;  (* Indication to read a register *)
    Write           : BOOL;  (* Indication to write into a register *)
    RegisterNumber  : USINT; (* Register number to be communicated with *)
    RegisterValue   : UINT;  (* Register Value to be written, provided Write = TRUE *)
END_VAR
```

Read : The read command is executed with rising edge on this input.

Write : The write command is executed with rising edge on this input.

RegisterNumber : The register number to be read/written is specified in this variable.

RegisterValue : In case the input *Write* is TRUE then this variable specifies the register value to be written. In case the input *Read* is TRUE, the value in this variable is not considered.

VAR_OUTPUT

```
VAR_OUTPUT
    Done                  : BOOL;  (* move completed *)
    Busy                  : BOOL;  (* function block is currently busy *)
    Error                 : BOOL;  (* Signals that an error has occurred within Function Block *)
    ErrorID               : UDINT; (* Error identification *)
    TerminalType          : UINT;  (* Terminal type/number involved in register communication *)
    CurrentRegisterNumber : USINT; (* Register Number that was under process *)
    CurrentRegisterValue  : UINT;  (* Register Value of the register under process *)
END_VAR
```

Done : Becomes TRUE, if a register communication has been successfully completed. The register number read/written is specified in *CurrentRegisterNumber* and the corresponding value in *CurrentRegisterValue*.

Busy: Becomes TRUE as soon as the function block is active, and becomes FALSE when it has returned to its initial state.

Error : Becomes TRUE, as soon as an error occurs.

ErrorID : If the error output is set, this parameter supplies the error number.

TerminalType : When Done is TRUE this output holds the terminal type/number involved in register communication.

CurrentRegisterNumber : When Done is TRUE this output gives the register number that was under process.

CurrentRegisterValue : When Done is TRUE this output gives the register value of the register under process.

VAR_IN_OUT

```
VAR_IN_OUT
    AxisRefIn   : NCTOPLC_AXLESTRUCT;
    AxisRefOut  : PLCTONC_AXLESTRUCT;
END_VAR
```

AxisRefIn : Axis structure from NC.

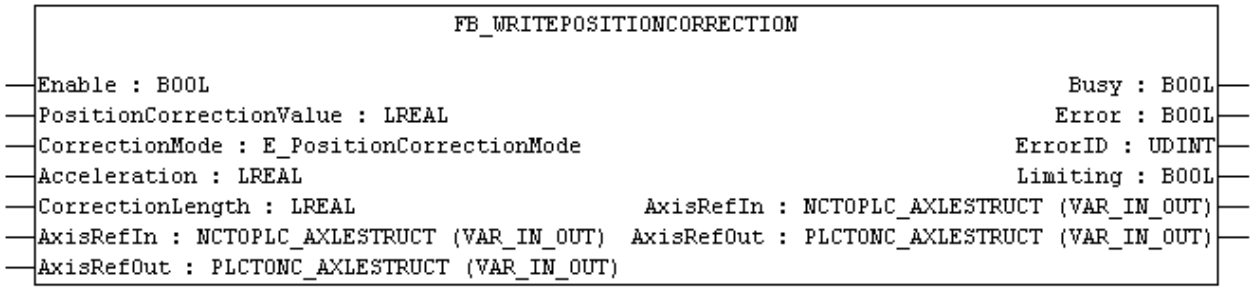
AxisRefOut : Axis structure from PLC.

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1314	PC (i386)	TcNc.Lib

3.4 FB_WritePositionCorrection

[This is preliminary documentation and subject to change.]



The function block FB_WritePositionCorrection writes an offset (PositionCorrectionValue) to the nominal position of an axis. Depending on the correction mode the data is written immediately or 'filtered' to the cyclic axis interface.

VAR_INPUT

```
VAR_INPUT
   Enable                : BOOL;
   PositionCorrectionValue : LREAL;
   CorrectionMode          : E_PositionCorrectionMode;
   Acceleration            : LREAL;
   CorrectionLenght        : LREAL;
END_VAR
```

Enable: Continuous writing of PositionCorrectionValue is enabled with a rising edge on this input. It must be true as long as new correction values should be accepted.

PositionCorrectionValue: Correction value that should be written to the cyclic axis interface

CorrectionMode: Depending on this mode the PositionCorrectionValue is written immediately or 'filtered' For a detailed description refer E_PosiitonCorrectionMode [▶ 66]

Acceleration: Depending on the CorrectionMode the maximum acceleration to achieve the new correction value is specified here. In case of PositionCorrectionMode_Fast [▶ 26] this value has an direct influence to the position delta per PLC-Tick.
max. accepted delta of position correction value = acceleration * (PLC-Cycletime)^2

CorrectionLength: If the CorrectionMode is equal to PositionCorrectionMode_FullLength this parameter becomes active. A change in PosiitonCorrectionValue is split up on this correction length

VAR_IN_OUT

```
VAR_IN_OUT
   AxisRefIn  : NCTOPLC_AXLESTRUCT;
   AxisRefOut : NCTOPLC_AXLESTRUCT;
END_VAR
```

AxisRefIn : Axis structure from NC.

AxisRefOut : Axis structure from PLC.

VAR_OUTPUT

```
VAR_OUTPUT
   Busy     : LREAL;
   Error    :   BOOL;
   ErrorId  : UDINT;
   Limiting :  BOOL;
ND_VAR
```

Busy: Becomes TRUE as soon as the function block is active, and becomes FALSE when it has returned to its initial state.

Error : Becomes TRUE, as soon as an error occurs.

ErrorId : If the error output is set, this parameter supplies the error number.

Limiting: Becomes TRUE, if the requested PositionCorrectionValue is not yet full accepted.

Hint:

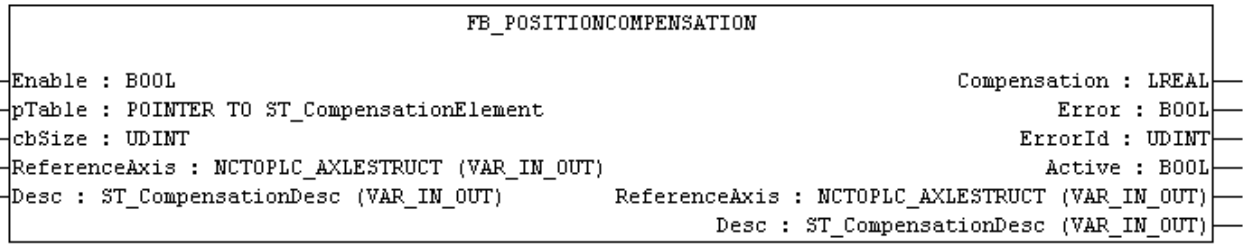
To use this function block successfully, Actual Position Correction in the System Manager must be enabled.

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1330	PC (i386)	TcNc.Lib (version >= 1.0.42)

3.5 FB_PositionCompensation

[This is preliminary documentation and subject to change.]



The function block FB_PositionCompensation returns the compensation value for e.g. pitch- and cross-compensation. If the FB is enabled, the compensation value is calculated by the position of the reference axis and a look-up in the correction table (pTable). In case of pitch-compensation the reference axis and compensated axis are the same. If this FB is used for cross-compensation they are different.

The compensation value that is returned by this FB is written to the axis by using FB_WritePositionCompensation.

VAR_INPUT

```
VAR_INPUT
   Enable: BOOL;
   pTable: POINTER TO ST_CompensationElement;
   cbSize: UDINT;
ND_VAR
```

Enable: Continuous calculation of the compensation value is enabled with a rising edge on this input. It must be true as long as compensation data should be calculated.

pTable: Pointer to the compensation table. This table is an array of type ST_CompensationElement

cbSize: Size in bytes of the compensation table

VAR_IN_OUT

```
VAR_IN_OUT
   ReferenceAxis : NCTOPLC_AXLESTRUCT;
   Desc          : ST_CompensationDesc;
END_VAR
```

ReferenceAxis: Axis structure from NC. This axis can be different to that axis that should be compensated. E.g. in case of a cross-compensation this will not be the compensated axis.

Desc: Description structure [▶ 67] for the compensation.

VAR_OUTPUT

```
VAR_OUTPUT
   Compensation : LREAL;
   Error        : BOOL;
   ErrorId      : UDINT;
   Active       : BOOL;
ND_VAR
```

Compensation: Calculated compensation value. Compensation will be 0.0 if Enable or Active is FALSE.

Error : Becomes TRUE, as soon as an error occurs.

ErrorId : If the error output is set, this parameter supplies the error number.

Active: Becomes TRUE, if compensation is active. If the working direction does not match to the current direction, it will be false.

Hint

For a sample refer 'Example Pitch Compensation [▶ 70]'

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1314	PC (i386)	TcNc.Lib

4 Functions

4.1 Analyse signals from NC

4.1.1 Status signals of a PTP-axis

(Signals from the cyclic axis interface of the NC and MC_xxx function blocks)

Requirements

NcToPlc.nStateDword (DWORD), variable of NcToPlc cyclic axis interface [▶ 54] structure, consists of various axis status signal flags. The flags discussed herewith are:

AxisHasJob [▶ 39]

AxisIsMoving [▶ 39] (AxisIsMovingForward OR AxisIsMovingBackwards)

AxisIsNotMoving [▶ 36]

AxisHasBeenStopped [▶ 38]

AxisIsAtTargetPosition [▶ 37]

AxisInPositionWindow [▶ 37]

NOTE:

Only the flags **AxisInPositionWindow** and **AxisIsAtTargetPosition** depend on the actual values (actual position, actual velocity). On the other hand, the rest of the flags discussed here depend on the setpoint values (set position, set velocity, set acceleration).

AxisHasJob [▶ 39] gives the correct and accurate indication of the axis command status. A transition of AxisHasJob from FALSE to TRUE indicates that the axis has got a new job and hence new parameters (like target position, velocity, acceleration or jerk). Completion of a command of an axis is indicated by a FALSE status of AxisHasJob flag.

AxisIsMoving [▶ 39] goes to TRUE when the motion command reaches the axis and the setpoint generator is active. Normally if the flag AxisIsMoving is TRUE, then the setpoint velocity has a value other than zero. An exception for this case is velocity override 0%, which implies that the axis has job and is logically moving but with a velocity zero (in this case AxisHasJob and AxisIsMoving are both TRUE but setpoint velocity is zero). AxisHasJob additionally includes the communication time for the request and response of the command.

AxisIsNotMoving [▶ 36] has an exact opposite status to that of AxisIsMoving. TRUE status of AxisIsNotMoving indicates that the axis is logically (not necessary physical standstill) standstill.

AxisHasBeenStopped [▶ 38] indicates the execution of a stop command for the corresponding axis. AxisHasBeenStopped goes TRUE as soon as the stop command is issued and executed and remains TRUE throughout the stopping phase (throughout the time axis stop command is active) and is set to FALSE only when a new command is executed.

AxisIsAtTargetPosition [▶ 37] depends on two parameters specified in the Global parameters of the corresponding axis in the TwinCAT System Manager; *Target Position Window* and *Target Position Monitoring Time*. AxisIsAtTargetPosition is TRUE when actual position is within the *Target Position Window* for a minimum of the time mentioned in *Target Position Monitoring Time* (without oscillating outside the *Target Position Window*).**NOTE:**

This flag to use should be enabled in TwinCAT System Manager, in the Global parameters of the corresponding axis under the name *Target Position Monitoring*.

AxisInPositionWindow [▶ 37] depends on the *Position Range Window* parameter in the Global parameters of the corresponding axis in the TwinCAT System Manager. The flag, AxisInPositionWindow is set TRUE when the actual position is within the value specified in *Position Range Window* and goes FALSE when the actual position value is outside this window.

NOTE:

This flag in order to use should be enabled in TwinCAT System Manager, in the Global parameters of the corresponding axis under the name *Position Range Monitoring*.

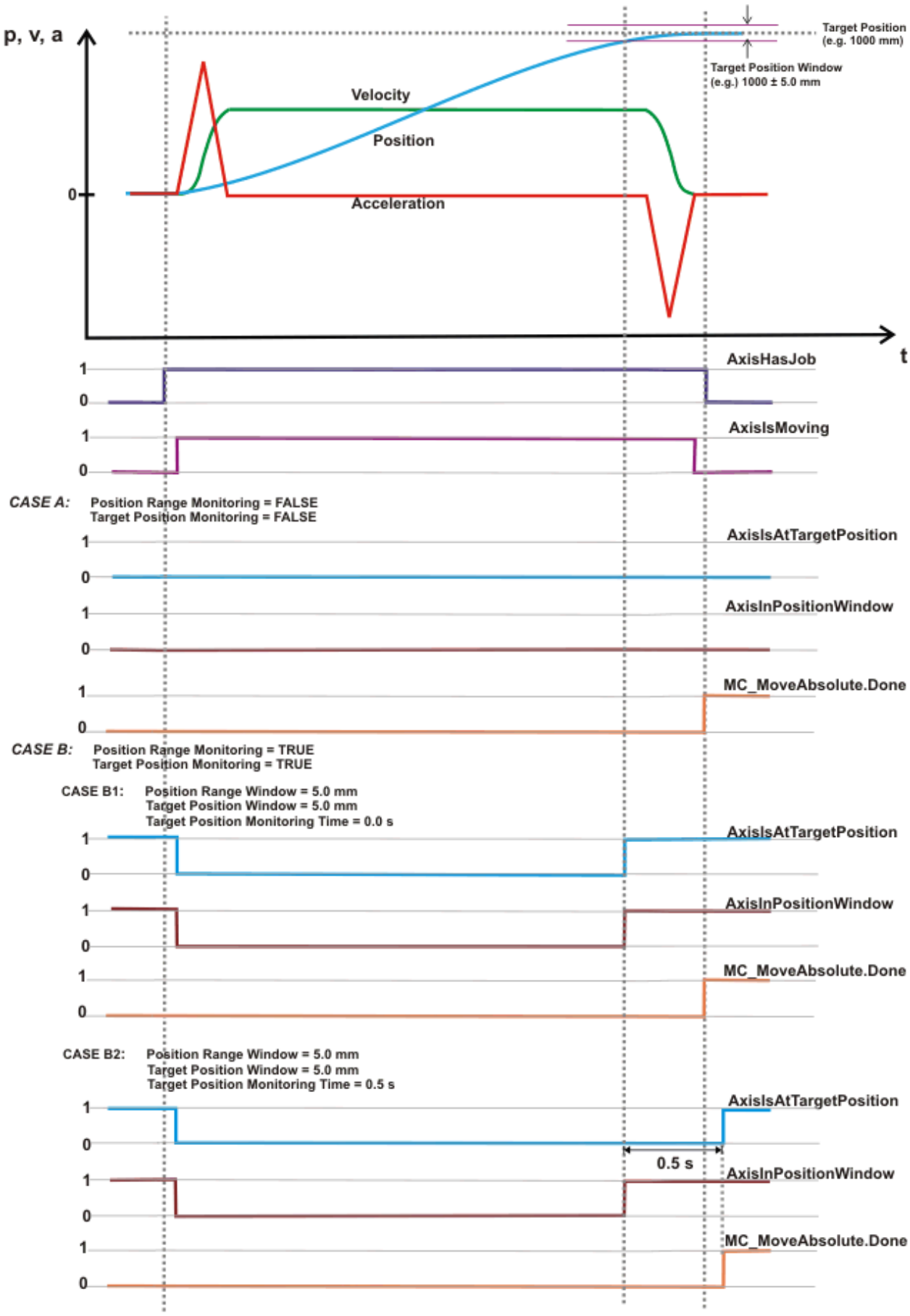
After positioning, all “MC_Move…” blocks check whether positioning was completed successfully. In the simplest case, the “AxisHasJob” flag of the NC axis is checked, which initially signifies that positioning was logically completed. Depending on the parameterization of the NC axis, further checks (quality criteria) are used:

- “Target position monitoring” (Important)
  If target position monitoring is active, the system waits for feedback from the NC. After positioning, the axis must be within the specified target position window for at least the specified time. If necessary, the position controller ensures that the axis is moved to the target position. If the position controller is switched off (Kv = 0) or weak, the target may not be reached. Floating position control may lead to the axis oscillating around the window but not remaining inside the window.
- “Position range monitoring”
  If position range monitoring is active, the system waits for feedback from the NC. After positioning, the axis must be within the specified positioning range window. If necessary, the position controller ensures that the axis is moved to the target position. If the position controller is switched off (Kv = 0) or weak, the target may not be reached.

If the axis is logically at the target position (logical standstill) but the parameterized position window has not been reached, monitoring of the above-mentioned NC feedback is aborted with error 19207 (0x4B07) after a constant timeout of 6 seconds.

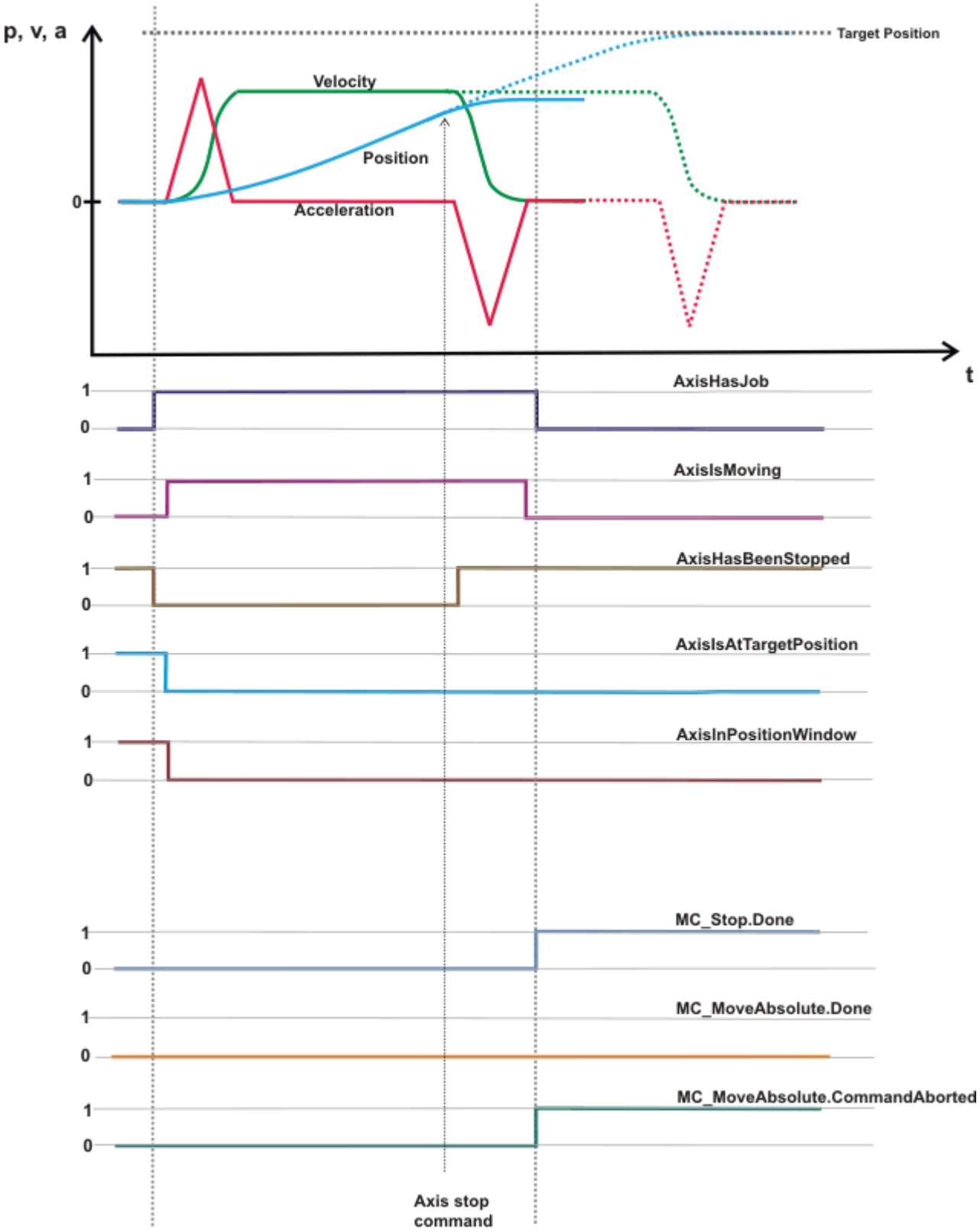
1.1. Positioning using MC_MoveAbsolute, MC_MoveRelative, …

A normal PTP sequence has been programmed to show the variations of signals corresponding to three different cases as shown below. the three different cases are made depending upon the Position Range Monitoring and Target Position Monitoring options.


p, v, a
Target Position
(e.g. 1000 mm)
Target Position Window
(e.g.) 1000 ± 5.0 mm
Velocity
Position
Acceleration
0
t
AxisHasJob
AxisIsMoving
CASE A:
Position Range Monitoring = FALSE
Target Position Monitoring = FALSE
AxisIsAtTargetPosition
AxisInPositionWindow
MC_MoveAbsolute.Done
CASE B:
Position Range Monitoring = TRUE
Target Position Monitoring = TRUE
CASE B1:
Position Range Window = 5.0 mm
Target Position Window = 5.0 mm
Target Position Monitoring Time = 0.0 s
AxisIsAtTargetPosition
AxisInPositionWindow
MC_MoveAbsolute.Done
CASE B2:
Position Range Window = 5.0 mm
Target Position Window = 5.0 mm
Target Position Monitoring Time = 0.5 s
AxisIsAtTargetPosition
0.5 s
AxisInPositionWindow
MC_MoveAbsolute.Done

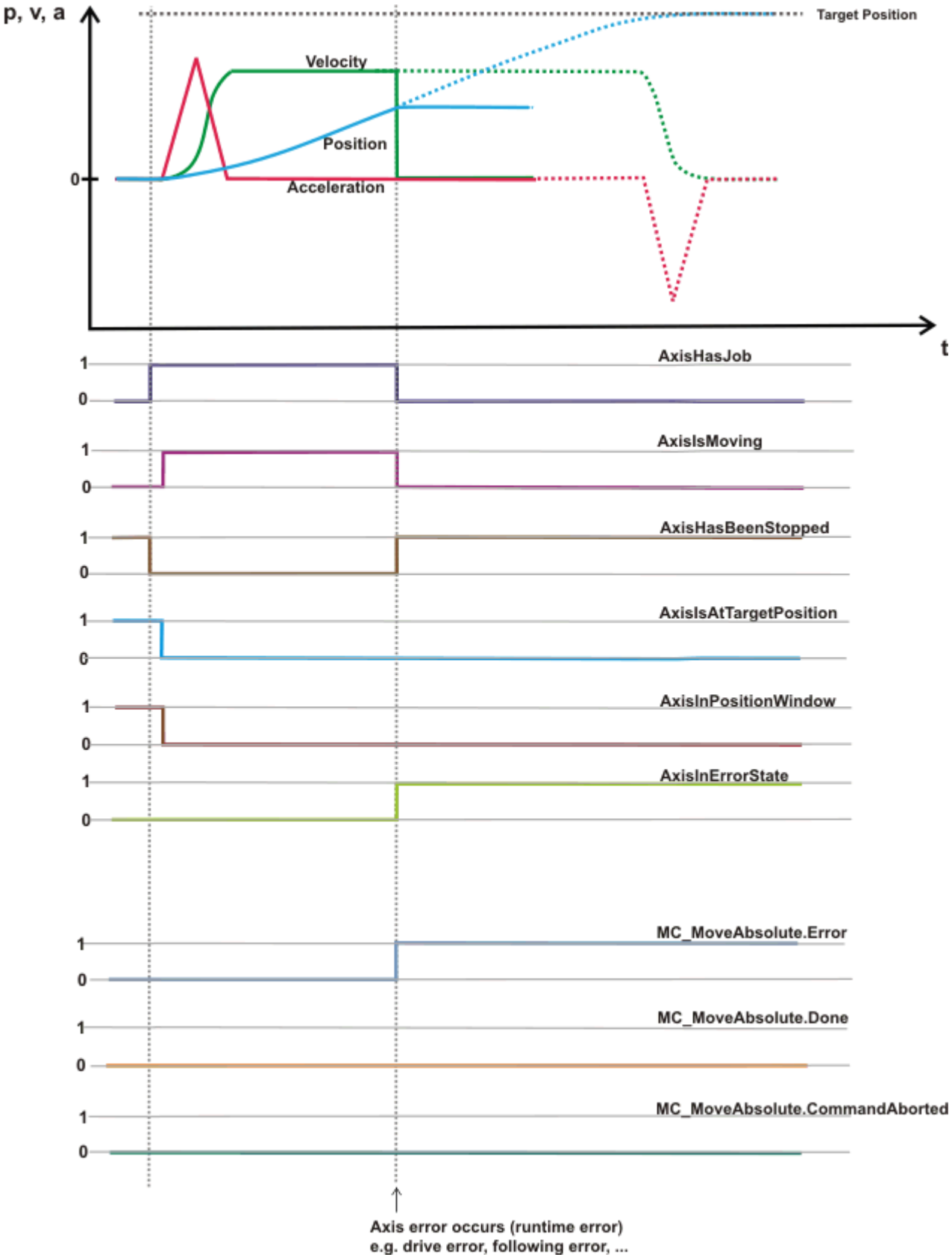
1.2. Stopping (aborting) of an activated positioning using MC_Stop, ...

If an axis is stopped during the axis positioning, the axis comes to rest with velocity decreasing at the deceleration rate.

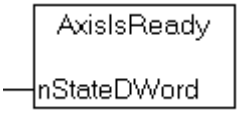


1.3. Occurrence of an NC or drive error (runtime error) during positioning

Occurence of an NC or drive error during positioning leads to abrupt fall of velocity to zero and hence the position remains constant, and no further operation is possible unless the axis is reset.



4.1.2 AxisIsReady



AxisIsReady returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsReady: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

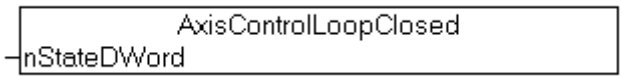
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Ready : BOOL;
END_VAR
```

```
Ready :=  AxisIsReady( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.3 AxisControlLoopClosed



AxisControlLoopClosed returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisControlLoopClosed: BOOL

```
VAR_INPUT
    nStateDWord     : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Beispiel

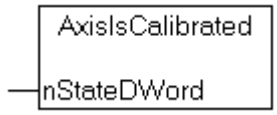
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    CamDataQueued : BOOL;
END_VAR
```

```
ControlLoopClosed := AxisControlLoopClosed ( NcToPlc1.nStateDWord );
```

Requirements

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9 from Build 1012	PC (i386)	TcNC.Lib

4.1.4 AxisIsCalibrated



AxisIsCalibrated returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsCalibrated: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

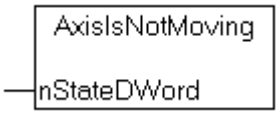
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Calibrated : BOOL;
END_VAR
```

```
Calibrated := AxisIsCalibrated( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.5 AxisIsNotMoving



AxisIsNotMoving returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsNotMoving: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

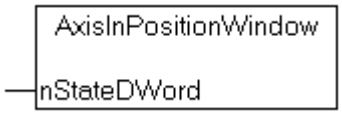
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Standstill : BOOL;
END_VAR
```

```
Standstill :=  AxisIsNotMoving( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.6 AxisInPositionWindow



AxisInPositionWindow returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisInPositionWindow: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

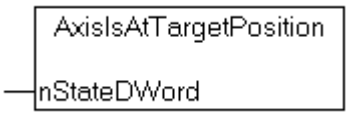
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    NearTarget : BOOL;
END_VAR
```

```
NearTarget := AxisInPositionWindow( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.7 AxisIsAtTargetPosition



AxisIsAtTargetPosition returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsAtTargetPosition: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
```

```
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    AtTarget : BOOL;
END_VAR
```

```
AtTarget := AxisIsAtTargetPosition( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.8 AxisInProtectedMode

AxisInProtectedMode returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisInProtectedMode: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

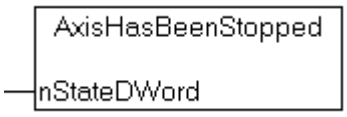
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Protected : BOOL;
END_VAR
```

```
Protected := AxisInProtectedMode( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.9 AxisHasBeenStopped



AxisHasBeenStopped returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisHasBeenStopped: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Stopped : BOOL;
END_VAR
```

```
Stopped := AxisHasBeenStopped( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.10 AxisHasJob

AxisHasJob returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisHasJob: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

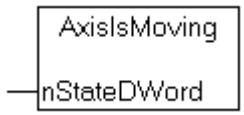
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    HasJob : BOOL;
END_VAR
```

```
HasJob := AxisHasJob( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.11 AxisIsMoving



AxisIsMoving returns TRUE if one of the corresponding signals, AxisIsMovingForward or AxisIsMovingBackwards, from the cyclic axis interface [▶ 54] from the NC to the PLC is set.

FUNCTION AxisIsMoving: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

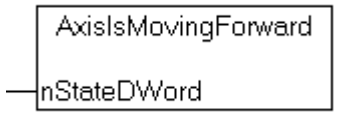
Example

```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Moving : BOOL;
END_VAR
Moving := AxisIsMoving( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.12 AxisIsMovingForward



AxisIsMovingForward returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsMovingForward: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

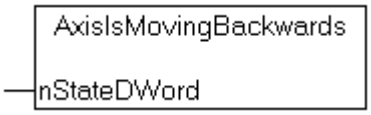
Example

```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Forward : BOOL;
END_VAR
Forward := AxisIsMovingForward( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.13 AxisIsMovingBackwards



AxisIsMovingBackwards returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsMovingBackwards: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

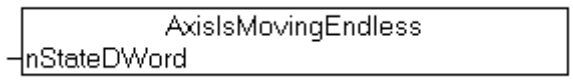
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Backwards : BOOL;
END_VAR
```

```
Backwards := AxisIsMovingBackwards( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.14 AxisIsMovingEndless



AxisIsMovingEndless returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsMovingEndless: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Beispiel

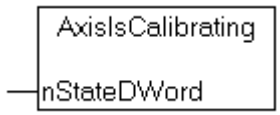
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Moving : BOOL;
END_VAR
```

```
MovingEndless := AxisIsMovingEndless( NcToPlc1.nStateDWord );
```

Requirements

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9 from Build 1012	PC (i386)	TcNC.Lib

4.1.15 AxisIsCalibrating



AxisIsCalibrating returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsCalibrating: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

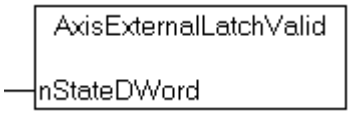
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Calibrating : BOOL;
END_VAR
```

```
Calibrating := AxisIsCalibrating( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.16 AxisExternalLatchValid



AxisExternalLatchValid returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisExternalLatchValid : BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
```

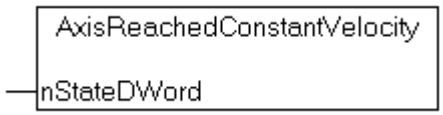
```
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    LatchValid : BOOL;
END_VAR
```

```
LatchValid := AxisExternalLatchValid( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.17 AxisReachedConstantVelocity



AxisReachedConstantVelocity returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisReachedConstantVelocity: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

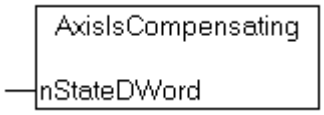
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    ConstVelocity : BOOL;
END_VAR
```

```
ConstVelocity := AxisReachedConstantVelocity( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.18 AxisIsCompensating



AxisIsCompensating returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisIsCompensating : BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    IsCompensating : BOOL;
END_VAR
```

```
IsCompensating := AxisIsCompensating( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.19 AxisHasExtSetPointGen

AxisHasExtSetPointGen returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisHasExtSetPointGen: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Beispiel

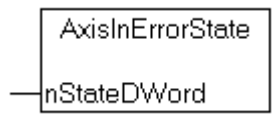
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    HasExtSetPointGen : BOOL;
END_VAR
```

```
HasExtSetPointGen:= AxisHasExtSetPointGen( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.20 AxisInErrorState



AxisInErrorState returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisInErrorState: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

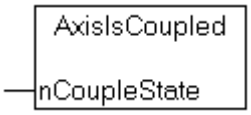
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Error : BOOL;
END_VAR
```

```
Error := AxisInErrorState( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.21 AxisIsCoupled



AxisIsCoupled evaluates the coupling state of the cyclic axis interface [▶ 54] between the NC and the PLC. **AxisIsCoupled** returns TRUE if nCoupleState > 1. This means that the axis is coupled to a master axis as a slave axis.

FUNCTION AxisIsCoupled: BOOL

```
VAR_INPUT
    nCoupleState : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

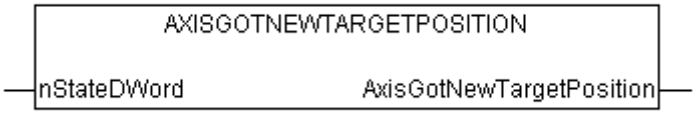
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Coupled : BOOL;
END_VAR
```

```
Coupled := AxisIsCoupled( NcToPlc1.nCoupleState );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.22 AxisGotNewTargetPosition



This function recognises, if the axis got a new position during the regular moving (e.g. with the function block FB_AxisNewTargPosAndVelo). Internally the status word of the cyclic axis interface [▶ 54] from the NC to the PLC is analysed.

FUNCTION AxisGotNewTargetPosition : BOOL

```
VAR_INPUT
    nStateDWord  DWORD;
END_VAR
```

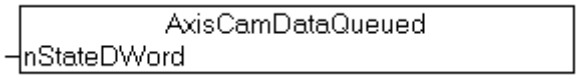
nStateDWord: Status word from the cyclic axis interface [▶ 54] from the NC to the PLC.

Return parameters	Description
TRUE	A new target position was pretended.
FALSE	No new target position was pretended at the axis reset and axis restart.

Requirements

Development Environment	Target System	PLC Libraries to include
TwinCAT v2.8 (Build > 707)	PC (i386)	TcNc.Lib

4.1.23 AxisCamDataQueued



AxisCamDataQueued returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisCamDataQueued: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

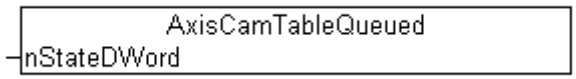
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    CamDataQueued : BOOL;
END_VAR
```

```
CamDataQueued := AxisCamDataQueued( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.9 from build 1025	PC (i386)	TcNC.Lib

4.1.24 AxisCamTableQueued



AxisCamTableQueued returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisCamTableQueued : BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

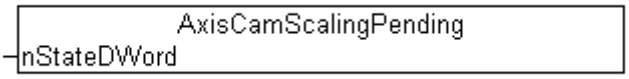
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    CamTableQueued : BOOL;
END_VAR
```

```
AxisCamTableQueued := AxisCamTableQueued ( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.9 from build 1032	PC (i386)	TcNC.Lib

4.1.25 AxisCamScalingPending



AxisCamScalingPending returns the corresponding signal from the cyclic axis interface [▶ 54] from the NC to the PLC

FUNCTION AxisCamScalingPending: BOOL

```
VAR_INPUT
    nStateDWord : DWORD;
END_VAR
```

nStateDWord : Status word from the cyclic axis interface [▶ 54] from the NC to the PLC

Example

```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    CamScalingPending : BOOL;
END_VAR
```

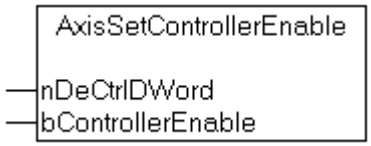
```
CamScalingPending := AxisCamScalingPending ( NcToPlc1.nStateDWord );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT V2.9 from build 1025	PC (i386)	TcNC.Lib

4.2 Set signals to NC

4.2.1 AxisSetControllerEnable



AxisSetControllerEnable sets the controller enable signal in *nDeCtrlDWord* in accordance with the *bControllerEnable* input, and returns *nDeCtrlDWord* as the result of the function.

FUNCTION AxisSetControllerEnable : DWORD

```
VAR_INPUT
    nDeCtrlDWord      : DWORD;
    bControlerEnable  : BOOL;
END_VAR
```

nDeCtrlDWord : Control-Word from the cyclic axis interface [▶ 63] from the PLC to the NC

bControllerEnable : Controller enable requiring to be set [FALSE,TRUE]

Function result: nDeCtrlDWord

Example

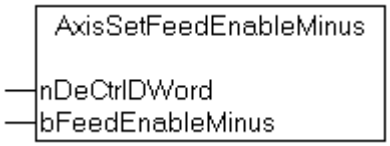
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

```
(* set controller enable signal *)
PlcToNc1.nDeCtrlDWord := AxisSetControllerEnable( PlcToNc1.nDeCtrlDWord, TRUE );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.2 AxisSetFeedEnableMinus



AxisSetFeedEnableMinus sets the feed enable signal for the reverse feed direction in *nDeCtrlDWord* in accordance with the *bFeedEnableMinus* input, and returns *nDeCtrlDWord* as the result of the function.

FUNCTION AxisSetFeedEnableMinus: DWORD

```
VAR_INPUT
    nDeCtrlDWord     : DWORD;
    bFeedEnableMinus : BOOL;
END_VAR
```

nDeCtrlDWord : Control-Word from the cyclic axis interface [▶ 63] from the PLC to the NC

bFeedEnableMinus : Feed enable requiring to be set [FALSE,TRUE]

Function result: nDeCtrlDWord

Example

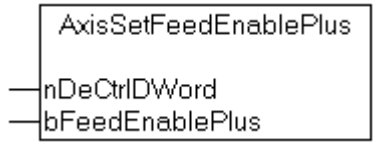
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

```
PlcToNc1.nDeCtrlDWord := AxisSetFeedEnableMinus(PlcToNc1.nDeCtrlDWord, TRUE );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.3 AxisSetFeedEnablePlus



AxisSetFeedEnablePlus sets the feed enable signal for the forward feed direction in *nDeCtrlDWord* in accordance with the *bFeedEnablePlus* input, and returns *nDeCtrlDWord* as the result of the function.

FUNCTION AxisSetFeedEnablePlus: DWORD

```
VAR_INPUT
    nDeCtrlDWord    : DWORD;
    bFeedEnablePlus : BOOL;
END_VAR
```

nDeCtrlDWord : Control-Word from the cyclic axis interface [▶ 63] from the PLC to the NC

bFeedEnablePlus : Feed enable requiring to be set [FALSE,TRUE]

Function result: nDeCtrlDWord

Example

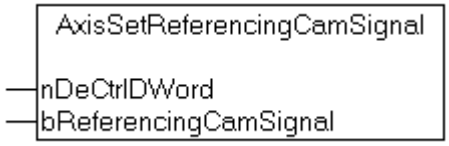
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

```
PlcToNc1.nDeCtrlDWord := AxisSetFeedEnablePlus(PlcToNc1.nDeCtrlDWord, TRUE );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.4 AxisSetReferencingCamSignal



AxisSetReferencingCamSignal sets the reference cam signal in *nDeCtrlDWord* according to the *bReferencingCamSignal* input, and returns *nDeCtrlDWord* as the result of the function.

FUNCTION AxisSetReferencingCamSignal: DWORD

```
VAR_INPUT
    nDeCtrlDWord          : DWORD;
    bReferencingCamSignal : BOOL;
END_VAR
```

nDeCtrlDWord : Control-Word from the cyclic axis interface [▶ 63] from the PLC to the NC

bReferencingCamSignal : Reference cam signal requiring to be set [FALSE,TRUE]

Function result: nDeCtrlDWord

Example

```
PROGRAM MAIN
VAR
    PlcToNc1            AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1            AT %IB1000 : NCTOPLC_AXLESTRUCT;
    ReferencingCamInput AT %IX0.0  : BOOL;
END_VAR
```

```
PlcToNc1.nDeCtrlDWord := AxisSetReferencingCamSignal(PlcToNc1.nDeCtrlDWord, ReferencingCamInput );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.5 AxisSetAcceptBlockedDriveSignal

AxisSetAcceptBlockedDriveSignal sets a signal in *nDeCtrlDWord* that allows the NC controller to move an axis off from a hardware limit switch although the drive does not signal its ready state. This function is hardware dependent.

FUNCTION AxisSetAcceptBlockedDriveSignal: DWORD

```
VAR_INPUT
    nDeCtrlDWord : DWORD;
    bEnable      : BOOL;
END_VAR
```

nDeCtrlDWord : Control-Word from the cyclic axis interface [▶ 63] from the PLC to the NC

bEnable : Signal requiring to be set [FALSE,TRUE]

Function result: nDeCtrlDWord

Example

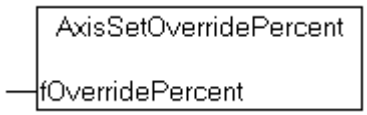
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

```
PlcToNc1.nDeCtrlDWord := AxisSetAcceptBlockedDriveSignal(PlcToNc1.nDeCtrlDWord, TRUE );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.10 from build 1313	PC (i386)	TcNC.Lib

4.2.6 AxisSetOverridePercent



AxisSetOverridePercent returns the corresponding NC-conform integer value for a percentage override value in *fOverridePercent*.

FUNCTION AxisSetOverridePercent: DWORD

```
VAR_INPUT
    fOverridePercent : LREAL;
END_VAR
```

fOverridePercent : Speed override as a percentage

Function result: NC-conform override value

Example

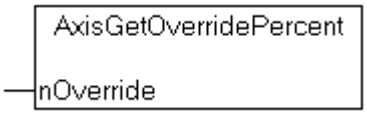
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

```
PlcToNc1.nOverride := AxisSetOverridePercent(100);
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.7 AxisGetOverridePercent



AxisGetOverridePercent determines the axis override from the cyclic axis interface [▶ 63] from the PLC to the NC, and returns it as a percentage value in the function result.

FUNCTION AxisGetOverridePercent : LREAL

```
VAR_INPUT
    nOverride : DWORD;
END_VAR
```

nOverride : nOverride from the cyclic axis interface [▶ 63] from the PLC to the NC

Function result: Axis override as a percentage

Example

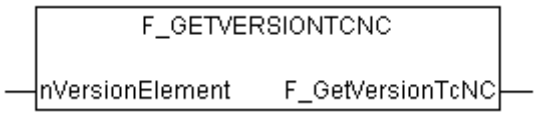
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    fOverride : LREAL;
END_VAR
```

```
fOverride := AxisGetOverridePercent( PlcToNc1.nOverride );
```

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.3 F_GetVersionTcNC



The function returns library version info.

FUNCTION F_GetVersionTcNC : UINT

```
VAR_INPUT
    nVersionElement : INT;
END_VAR
```

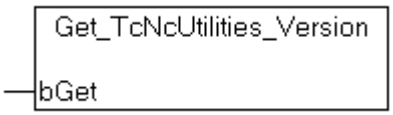
nVersionElement : Version parameter:

- 1 : major number;
- 2 : minor number;
- 3 : revision number;

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.4 Get_TcNcUtilities_Version



Get_TcNcUtilities_Version determines the version number of the PLC library TcNcUtilities.lib. The function returns the version number in a string.

FUNCTION Get_TcNcUtilities_Version: STRING(20)

```
VAR_INPUT
    bGet : BOOL;
END_VAR
```

bGet : Signal for execution of the command.

Requirements

Development environment	Target system type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib: Function included only for TwinCAT 2.7 compatibility!. Please use **F_GetVersionTcNC**() function in future PLC projects.

5 Data types

5.1 Cyclical NC/PLC interface

5.1.1 NCTOPLC_AXLESTRUCT2

```
TYPE NCTOPLC_AXLESTRUCT
STRUCT
    nStateDWord         : DWORD; (* Status double word *)
    nErrorCode          : DWORD; (* Axis error code *)
    nAxisState          : DWORD; (* Axis moving status *)
    nAxisModeCon        : DWORD; (* Axis mode confirmation (feedback from NC) *)
    nCalibrationState   : DWORD; (* State of axis calibration (homing) *)
    nCoupleState        : DWORD; (* Axis coupling state *)
    nSvbEntries         : DWORD; (* SVB entries/orders (SVB = Set preparation task) *)
    nSafEntries         : DWORD; (* SAF entries/orders (SAF = Set execution task *)
    nAxisId             : DWORD; (* Axis ID *)
    nOpModeDWord        : DWORD; (* Current operation mode *)
    nReserved2_HIDDEN   : DWORD; (* reserved *)
    fPosIst             : LREAL; (* Actual position (absolut value from NC) *)
    fModuloPosIst       : LREAL; (* Actual position as modulo value (e.g. in degrees) *)
    nModuloTurns        : DINT;  (* Actual modulo turns *)
    fVeloIst            : LREAL; (* Actual velocity (optional) *)
    fPosDiff            : LREAL; (* Position difference (lag distance) *)
    fPosSoll            : LREAL; (* Setpoint position *)
    fVeloSoll           : LREAL; (* Setpoint velocity *)
    fAccSoll            : LREAL; (* Setpoint acceleration, OLD: "fReserve1_HIDDEN" *)
    fReserve2_HIDDEN    : LREAL; (* reserved *)
    fReserve3_HIDDEN    : LREAL; (* reserved *)
    fReserve4_HIDDEN    : LREAL; (* reserved *)
END_STRUCT
END_TYPE
```

```
TYPE NCTOPLC_AXLESTRUCT2
STRUCT
    nStateDWord             : DWORD; (* Status double word *)
    nErrorCode              : DWORD; (* Axis error code *)
    nAxisState              : DWORD; (* Axis moving status *)
    nAxisModeCon            : DWORD; (* Axis mode confirmation (feedback from NC) *)
    nCalibrationState       : DWORD; (* State of axis calibration (homing) *)
    nCoupleState            : DWORD; (* Axis coupling state *)
    nSvbEntries             : DWORD; (* SVB entries/orders (SVB = Set preparation task) *)
    nSafEntries             : DWORD; (* SAF entries/orders (SAF = Set execution task *)
    nAxisId                 : DWORD; (* Axis ID *)
    nOpModeDWord            : DWORD; (* Current operation mode *)
    nActiveControlLoopIndex : WORD;  (* Active control loop index (equivalent to old variable
"nCtrlLoopIndex") *)
    nControlLoopIndex       : WORD;  (* Axis control loop index (0, 1, 2, … when multiple control
loops are used) *)
    fPosIst                 : LREAL; (* Actual position (absolut value from NC) *)
    fModuloPosIst           : LREAL; (* Actual position as modulo value (e.g. in degrees) *)
    nModuloTurns            : DINT;  (* Actual modulo turns *)
    fVeloIst                : LREAL; (* Actual velocity (optional) *)
    fPosDiff                : LREAL; (* Position difference (lag distance) *)
    fPosSoll                : LREAL; (* Setpoint position *)
    fVeloSoll               : LREAL; (* Setpoint velocity *)
    fAccSoll                : LREAL; (* Setpoint acceleration, OLD: "fReserve1_HIDDEN" *)
    fPosTarget              : LREAL; (* Estimated target position *)
    fModuloPosSoll          : LREAL; (* Setpoint modulo position (e.g. in degrees) *)
    nModuloTurnsSoll        : DINT;  (* Setpoint modulo turns *)
    nCmdNo                  : WORD;  (* Continuous actual command number *)
    nCmdState               : WORD; (* Command state *)
END_STRUCT
END_TYPE
```

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
1	UINT32	0-3	-	-	nStateDWord	StateDWord	Status double word See detailed description as well [▶ 69]
			0	0/1	Operational	Operational	Axis is ready for operation
			1	0/1	Homed	Homed	Axis has been referenced/ homed ("Axis calibrated")
			2	0/1	NotMoving	NotMoving	Axis is logically stationary ("Axis not moving")
			3	0/1	InPositionArea	InPositionArea	Axis is in position window (physical feedback)
			4	0/1	InTargetPosition	InTargetPosition	Axis is at target position (PEH) (physical feedback)
			5	0/1	Protected	Protected	Axis is in a protected operating mode (e.g. as a slave axis)
			6	0/1	ErrorPropagationDelayed	ErrorPropagationDelayed	Axis signals an error pre warning (from TC 2.11)
			7	0/1	HasBeenStopped	HasBeenStopped	Axis has been stopped or is presently executing a stop
			8	0/1	HasJob	HasJob	Axis has instructions, is carrying instructions out
			9	0/1	PositiveDirection	PositiveDirection	Axis moving to logically larger values

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
			10	0/1	NegativeDirection	NegativeDirection	Axis moving to logically smaller values
			11	0/1	HomingBusy	HomingBusy	Axis referenced ("Axis being calibrated")
			12	0/1	ConstantVelocity	ConstantVelocity	Axis has reached its constant velocity or rotary speed
			13	0/1	Compensating	Compensating	Section compensation passive[0]/ active[1] (s. "MC_MoveSuperImposed")
			14	0/1	ExtSetPointGenEnabled	ExtSetPointGenEnabled	External setpoint generator enabled
			15	0/1			Operating mode not yet executed (Busy). Not implemented yet!
			16	0/1	ExternalLatchValid	ExternalLatchValid	External latch value or sensing switch has become valid
			17	0/1	NewTargetPos	NewTargetPos	Axis has a new target position or a new velocity
			18	0/1			Axis is not at target position or cannot reach the target position (e.g. stop).Not implemented yet!
			19	0/1	ContinuousMotion	ContinuousMotion	Axis has target position (±) endless
			20	0/1	ControlLoopClosed	ControlLoopClosed	Axis is ready for operation and axis

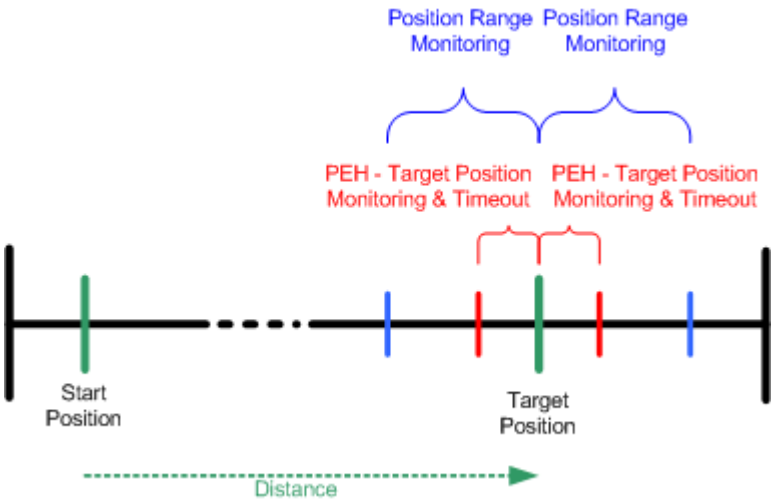
No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
							control loop is closed (e.g. position control)
			21	0/1	CamTableQueued	CamTableQueued	CAM table is queued for "Online Change" and waiting for activation
			22	0/1	CamDataQueued	CamDataQueued	CAM data (only MF) are queued for "Online Change" and waiting for activation
			23	0/1	CamScalingPending	CamScalingPending	CAM scaling are queued for "Online Change" and waiting for activation
			24	0/1	CmdBuffered	CmdBuffered	Following command is queued in then command buffer (s. Buffer Mode) (from TwinCAT V2.10 Build 1311)
			25	0/1	PTPmode	PTPmode	Axis in PTP mode (no slave, no NCI axis, no FIFO axis) (from TC 2.10 Build 1326)
			26	0/1	SoftLimitMinExceeded	SoftLimitMinExceeded	Position software limit switch minimum is exceeded (from TC 2.10 Build 1327)
			27	0/1	SoftLimitMaxExceeded	SoftLimitMaxExceeded	Position software limit switch maximum is exceeded

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
							(from TC 2.10 Build 1327)
			28	0/1	DriveDeviceError	DriveDeviceError	Hardware drive device error (no warning), interpretation only possible when drive is data exchanging, e.g. EtherCAT "OP"-state (from TC 2.10 Build 1326)
			29	0/1	MotionCommandsLocked	MotionCommandsLocked	Axis is locked for motion commands (TcMc2)
			30	0/1	IoDataInvalid	IoDataInvalid	IO data invalid (e.g. 'WcState' or 'CdlState')
			31	0/1	Error	Error	Axis is in a fault state
2	UINT32	4-7	-	≥0	nErrorCode	ErrorCode	Axis error code
3	UINT32	8-11	-	ENUM	nAxisState	AxisState	Present state of the axis movement
4	UINT32	12-15	-	ENUM	nAxisModeCon	nAxisModeConfirmation	Axis operating mode (feedback from the NC)
5	UINT32	16-19	-	ENUM	nCalibrationState	HomingState	Axis referencing status ("Calibration status")
6	UINT32	20-23	-	ENUM	nCoupleState	CoupleState	Axis coupling status
7	UINT32	24-27	-	≥0	nSvbEntries	SvbEntries	SVB entries/ tasks
8	UINT32	28-31	-	≥0	nSafEntries	SafEntries	SAF entries/ tasks (NC interpreter, FIFO group)

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
9	UINT32	32-35	-	>0	nAxisId	AxisId	Axis ID
10	DWORD	36-39	-	0/1	nOpModeDWord, bOpMode...	OpModeDWord, OpMode...	Axis run modus Double Word
			0	0/1	...PosAreaMonitoring	...PosAreaMonitoring	Position range monitoring
			1	0/1	...TargetPosMonitoring	...TargetPosMonitoring	Target position window monitoring
			2	0/1	...Loop	...Loop	Loop movement
			3	0/1	...MotionMonitoring	...MotionMonitoring	Physical motion monitoring
			4	0/1	...PEHTimeMonitoring	...PEHTimeMonitoring	PEH time monitoring
			5	0/1	...BacklashComp	...BacklashComp	Backlash compensation
			6	0/1	...DelayedErrorReaction	...DelayedErrorReaction	Delayed nc error reaction
			7	0/1	...Modulo	...Modulo	Modulo axis (modulo values displayed)
			8-15	0/1			RESERVED
			16	0/1	...PosLagMonitoring	...PosLagMonitoring	Following Error monitoring position
			17	0/1	...VeloLagMonitoring	...VeloLagMonitoring	Following Error monitoring velocity
			18	0/1	...SoftLimitMinMonitoring	...SoftLimitMinMonitoring	End location monitoring min.
			19	0/1	...SoftLimitMaxMonitoring	...SoftLimitMaxMonitoring	End location monitoring max.
			20	0/1	...PosCorrection	...PosCorrection	Position compensation (position correction)
			21	0/1	...AllowSlaveCommands	...AllowSlaveCommands	Allow motion commands to slave
			22	0/1			RESERVED

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
			23	0/1	ApplicationRequest	ApplicationRequest	Application request bit for the PLC (PLC code), e.g. for an "application homing request" from TwinCAT V2.11 Build 1546
			24-31	0/1			RESERVED
11	UINT16	40-41	-	≥0	nActiveControlLoopIndex	ActiveControlLoopIndex	Active axis control loop index (equivalent to old variable "nCtrlLoopIndex")) from TwinCAT V2.10 Build 1311
12	UINT16	42-43	-	≥0	nControlLoopIndex	ControlLoopIndex	Axis control index (0, 1, 2, … when multiple control loops are used) from TwinCAT V2.10 Build 1311
13	REAL64	44-51	-	±∞	fPosIst	ActPos	Actual position (calculated absolute value)
14	REAL64	52-59	-	>∞	fModuloPosIst	ModuloActPos	Modulo actual position (calculate value in, for example, degrees)
15	INT32	60-63	-	±∞	nModuloTurns	ModuloActTurns	Modulo actual rotations
16	REAL64	64-71	-	±∞	fVeloIst	ActVelo	Actual velocity (optional)
17	REAL64	72-79	-	±∞	fPosDiff	PosDiff	Following error (position)

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
18	REAL64	80-87	-	±∞	fPosSoll	SetPos	Set position (calculated absolute value)
19	REAL64	88-95	-	±∞	fVeloSoll	SetVelo	Set velocity
20	REAL64	96-103	-	±∞	fAccSoll	SetAcc	Set acceleration
21	REAL64	104-111	-	±∞	fPosTarget	TargetPos	Estimated target position from TwinCAT V2.10 Build 1311
22	REAL64	112-119	-	>∞	fModuloPosSoll	ModuloSetPos	Modulo setpoint position (calculate value in, for example, degrees) from TwinCAT V2.10 Build 1311
23	INT32	120-123	-	±∞	nModuloTurnsSoll	ModuloSetTurns	Modulo setpoint rotations from TwinCAT V2.10 Build 1311
24	UINT16	124-125	-	≥0	nCmdNo	CmdNo	Continuous command number from the current command (s. Buffer Mode) from TwinCAT V2.10 Build 1311
25	UINT16	126-127	-	≥0	nCmdState	CmdState	Command state (s. Buffer Mode) from TwinCAT V2.10 Build 1311



Description of the contents of the individual fields:

Define	Axis referencing status (*nCalibrationState* resp. *HomingState*)
0	Referencing process completed (READY)
1	Continuous start in the direction of the referencing cam (Note: If the referencing cam is active from the beginning, the referencing status will immediately start with 3)
2	Wait for a rising edge from the referencing cam and initiate the axis stop
3	Wait until the axis settled, check if the referencing cam is still active and then start from the referencing cam into the direction of the synchronising pulse
4	Wait for the falling edge of the referencing cam
5	Activate the latch and stop the axis when the latch has become valid
6	When the axis is stationary, set the calculated actual position (actual position = reference position + break distance)

See also notes in MC_Home

Define	Axis coupling status (*nCoupleState* resp. *CoupleState*)
0	Single axis that is neither a master nor a slave (SINGLE)
1	Master axis with any number of slaves (MASTER)
2	Slave axis that is the master of another slave (MASTERSLAVE)
3	Just a slave axis (SLAVE)

Define	Master: Present state of the axis movement / moving phase of the continuous master axis (servo) (*nAxisState* resp. *AxisState*)
0	Set value generator not active (INACTIVE)
1	Set value generator active (RUNNING)
2	Velocity override is zero (OVERRIDE_ZERO)
3	Constant velocity (PHASE_VELOCONST)
4	Acceleration phase (PHASE_ACCPOS)
5	Deceleration phase (PHASE_ACCNEG)

Define	Master: Present state of the axis movement / moving phase of the discrete master axis (high/low speed) (*nAxisState* resp. *AxisState*)
0	Set value generator not active

Define	Master: Present state of the axis movement / moving phase of the discrete master axis (high/low speed) (*nAxisState* resp. *AxisState*)
1	Moving phase (rapid or creep movement)
2	Switchover delay from rapid to creep motion
3	Creep motion (within the creep region)
4	Braking time (starting from the braking distance in front of the target)

Define	Slave: Present state of the axis movement / moving phase of the continuous slave axis (servo) (*nAxisState* resp. *AxisState*) Remark: at the time only for salves of type *synchronised generator*!.
0	Slave generator not active (INACTIVE)
11	Slave is located in a movement prephase (PREPHASE)
12	Slave is synchronising (SYNCHRONIZING)
13	Slave is synchronised and moves synchronously (SYNCHRON)

Requirements

Development environement	Target System Type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

5.1.2 PLCTONC_AXLESTRUCT

```
TYPE PLCTONC_AXLESTRUCT
STRUCT
    nDeCtrlDWord      : DWORD; (* control double word *)
    nOverride         : DWORD; (* velocity override *)
    nAxisModeReq      : DWORD; (* axis operating mode (PLC request) *)
    nAxisModeDWord    : DWORD; (* *)
    fAxisModeLReal    : LREAL; (* *)
    fActPosCorrection : LREAL; (* correction value for current position *)
    fExtSetPos        : LREAL; (* external position setpoint *)
    fExtSetVelo       : LREAL; (* external velocity setpoint *)
    fExtSetAcc        : LREAL; (* external acceleration setpoint *)
    nExtSetDirection  : DINT;  (* external direction setpoint *)
    nReserved1        : DWORD; (* reserved *)
    fExtCtrlOutput    : LREAL; (* external controller output *)
    nReserved_HIDDEN  : ARRAY [72..127] OF BYTE;
END_STRUCT
END_TYPE
```

For each NC axis a data block of 128 bytes is available for data transport from the NC to the PLC, and another data block, also 128 bytes, is available for data transport from the PLC to the NC. The PLC programmer must create a variable for each direction and each axis, and must fix them in the I/O area with the AT instruction to the input and output area. The assignment of NC variables to PLC variables is carried out by the TwinCAT System Manager.

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
1	UINT32	0-3	-	0/1	nDeCtrlDWord	ControlDWord	Control double word:
			0	0/1	Enable	Enable	Enable controller
			1	0/1	FeedEnablePlus	FeedEnablePlus	Feed enable plus
			2	0/1	FeedEnableMinus	FeedEnableMinus	Feed enable minus

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
			3	0/1	-	-	RESERVED
			4	0/1	-	-	RESERVED
			5	0/1	HomingSensor	HomingSensor	Referencing cam signal or homing sensor
			6	0/1	-	-	RESERVED
			7	0/1	-	-	RESERVED
			8	0/1	AcceptBlockedDrive	AcceptBlockedDrive	Accept drive setpoint blocking (e.g. hardware position limits) (from TwinCAT V2.10 Build 1311)
			9	0/1	BlockedDriveDetected	BlockedDriveDetected	Axis is blocked (e.g. mechanical limit). Not implemented yet!
			10-29	0/1	-	-	RESERVED
			30	0/1	PlcDebugFlag	PlcDebugFlag	Debug function PLC. Only for internal use!
			31	0/1	NcDebugFlag	NcDebugFlag	Debug function NC. Only for internal use!
2	UINT32	4-7	-	0...1000000	nOverride	Override	Velocity override (0% to 100%)
3	UINT32	8-11	-		nAxisModeReq	AxisModeRequest	Axis operating mode. Only provided for internal use!
4	UINT32	12-15	-		nAxisModeDWord	AxisModeDWord	Only provided for internal use!
5	REAL64	16-23	-		fAxisModeLReal	AxisModeLReal	Only provided for internal use!
6	REAL64	24-31	-		fActPosCorrection	PositionCorrection	Actual position correction value

No	Data type	Byte	Bit	Def. range	Variable name	Variable name (since 2.11 resp. TcMc2)	Description
7	REAL64	32-39	-		fExtSetPos	ExtSetPos	External position setpoint
8	REAL64	40-47	-		fExtSetVelo	ExtSetVelo	External velocity setpoint
9	REAL64	48-55	-		fExtSetAcc	ExtSetAcc	External acceleration setpoint
10	INT32	56-59	-		nExtSetDirection	ExtSetDirection	External direction setpoint [-1, 0, 1]
11	UINT32	60-63	-		nReserved1	-	RESERVED
12	REAL64	64-71	-		fExtCtrlOutput	ExtControllerOutput	External controller output. Not implemented yet!
13	REAL64	72-79	-	±∞	-	GearRatio1	Gear ration (couple factor) 1
14	REAL64	80-87	-	±∞	-	GearRatio2	Gear ration (couple factor) 2
15	REAL64	88-95	-	±∞	-	GearRatio3	Gear ration (couple factor) 3
16	REAL64	96-103	-	±∞	-	GearRatio4	Gear ration (couple factor) 4
17	-	104-127	-	-	nReserved		RESERVED

Requirements

Development enviroment	Target System Type	PLC libraries to include
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

5.2 E_CmdTypeNewTargPosAndVelo

```
TYPE E_CmdTypeNewTargPosAndVelo :(
    CHANGE_POS                      := 1,
    CHANGE_VELO ,
    CHANGE_POSANDVELO ,
    CHANGE_POS_AT_SWITCHPOS         := 9,
    CHANGE_VELO_AT_SWITCHPOS ,
    CHANGE_POSANDVELO_AT_SWITCHPOS ,
    REACH_VELO_AT_POS               := 14
);
END_TYPE
```

This data type is required in conjunction with the MC_NewPosAndVelo function, which can be used for changing the target position and the velocity of an axis in motion. This type defines the mode for this function:

CHANGE_POS: The target position is changed instantaneously, i.e. the axis immediately aims for the new target.

CHANGE_VELO: The velocity of the axis is changed instantaneously.

CHANGE_POSANDVELO: Both parameters, i.e. target position and velocity, are changed instantaneously.

CHANGE_POS_AT_SWITCHPOS: The target position is changed when a switching position is reached.

CHANGE_VELO_AT_SWITCHPOS: The velocity is changed when a switching position is reached.

CHANGE_POSANDVELO_AT_SWITCHPOS: Both parameters, i.e. target position and velocity, are changed when a switching position is reached.

REACH_VELO_AT_POS : The velocity is changed such that the new velocity is reached at the switching position. This mode is only available in conjunction with the optimised set value generator for an axis (see global axis parameters from TwinCAT 2.10 Build 1052).

Requirements

Development Environment	Target System	PLC LIbraries to include
TwinCAT v2.8	PC (i386)	TcNc.Lib

5.3 E_TargPosType

```
TYPE E_TargPosType :
(
    POS_ABSOLUTE  := 1,   (*Absolute position*)
    POS_RELATIVE,         (*Relative position*)
    POS_MODULO    := 5    (*Modulo position*)
);
END_TYPE
```

Requirements

Development Environment	Target System	PLC Libraries to include
TwinCAT v2.8	PC (i386)	TcNc.Lib

5.4 E_StartPosType

```
TYPE E_StartPosType :
(
    START_ABSOLUTE       := 1,  (*Start to absolute position*)
    START_RELATIVE ,            (*Start to relative position*)
    START_ENDLESS_PLUS ,        (*Start to endless positive position*)
    START_ENDLESS_MINUS ,       (*Start to endless negative position*)
    START_MODULO                (*Start to modulo position *)
);
END_TYPE
```

Requirements

Development Environment	Target System	PLC Libraries to include
TwinCAT v2.8	PC (i386)	TcNc.Lib

5.5 E_PositionCorrectionMode

[This is preliminary documentation and subject to change.]

```
TYPE E_PositionCorrectionMode:
(
   POSITIONCORRECTIONMODE_UNLIMITED, (* no limitation - pass correction immediately *)
   POSITIONCORRECTIONMODE_FAST,      (* limitatation to maximum position change per cycle *)
   POSITIONCORRECTIONMODE_FULLLENGTH (* limitation uses full length to adapt to correction in small
steps *);
);
END_TYPE
```

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1330	PC (i386)	TcNc.Lib (version >= 1.0.42)

5.6 ST_CompensationDesc

[This is preliminary documentation and subject to change.]

ST_CompensationDesc

```
TYPE ST_CompensationDesc:
STRUCT
   fPosMin        : LREAL; (*compensation starts with this position*)
   fPosMax        : LREAL; (*compensation ends with this position*)
   nTableElements : UDINT; (* number of entries in table *)
   eDirection     : E_WorkingDirection:=WorkingDirectionBoth; (*compensation is just working in the
selected working direction*)
   bModulo        : BOOL := FALSE;
   eTableType     : E_CompensationTableType:=TableType1DEquidistant;
END_STRUCT
END_TYPE
```

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1330	PC (i386)	TcNc.Lib (version >= 1.0.42)

5.7 E_CompensationTableType

[This is preliminary documentation and subject to change.]

E_CompensationTableType

```
TYPE E_CompensationTableType:
(
   TableTypeNone := 0,
   TableType1DEquidistant := 1
);
END_TYPE
```

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1330	PC (i386)	TcNc.Lib (version >= 1.0.42)

5.8 E_WorkingDirection

[This is preliminary documentation and subject to change.]

E_WorkingDirection

```
TYPE E_WorkingDirection:
(
   WorkingDirectionNone := 0,
   WorkingDirectionBoth := 1,
   WorkingDirectionPlus := 2,
   WorkingDirectionMinus := 3
);
END_TYPE
```

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1330	PC (i386)	TcNc.Lib (version >= 1.0.42)

5.9 ST_CompensationElement

[This is preliminary documentation and subject to change.]

ST_CompensationElement

```
TYPE ST_CompensationElement:
STRUCT
   fPos: LREAL; (* uncorrected absolute position *)
   fCompensation: LREAL; (* correction value *)
END_STRUCT
END_TYPE
```

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1330	PC (i386)	TcNc.Lib (version >= 1.0.42)

6 Appendix

6.1 Discrete high/low speed axis (two speed)

Starting velocity value range V	Interpretation of the starting velocity with 100% override (required running velocity / running stage)
V > 50	rapid traverse
0 < V≤ 50	creep distance
V ≤0	ERROR

Value range override X	Interpretation of the override value (100% º 1.000.000)
X > 50% (500000)	rapid traverse
0% < X ≤ 50% (500000)	creep distance
X = 0%	stationary (tolerance window: <100 ≡ <0.01%)

An override change (also override = 0) only becomes effective within the main travel phase
If the override is set to 0 within one of the braking phases, the initiated braking phase is terminated unaffected.

6.2 Drive interface for high/low speed axes NC->IO (12 bytes)

No	Data type	Byte	Bit	Def. Range	Variable Name	Description
1	UINT32	0-3	-	-	nOutData1	Drive output data 1 (NC->IO)
2	UINT32	4-7	-	-	nOutData2	Drive output data 2 (NC->IO)
3	UINT8	8	-	-	nControlByte	Control byte
			0	0/1	bMinusHigh	Direction: negative Velocity: fast
			1	0/1	bMinusLow	Direction: negative Velocity: slowly
			2	0/1	bPlusLow	Direction: positive Velocity: slowly
			3	0/1	bPlusHigh	Direction: positive Velocity: fast
			4	0/1	-	RESERVED
			5	0/1	-	RESERVED
			6	0/1	bBreakInv	Inverse braking bit (0 ≡ ACTIVE, 1 ≡ PASSIVE)
			7	0/1	bBreak	Braking bit (0 ≡ PASSIVE, 1 ≡ ACTIVE)
4	UINT8	9	-	-	nExtControlByte	Extended control byte
			0	0/1	bDirectionMinus	Direction: negative
			1	0/1	bDirectionPlus	Direction: positive
			2	0/1	bVeloLow	Velocity: slowly
			3	0/1	bVeloHigh	Velocity: fast
			4	0/1	-	RESERVED
			5	0/1	-	RESERVED

No	Data type	Byte	Bit	Def. Range	Variable Name	Description
			6	0/1	bBreakInv	Inverse braking bit (0 ≡ ACTIVE, 1 ≡ PASSIVE)
			7	0/1	bBreak	Braking bit (0 ≡ PASSIVE, 1 ≡ ACTIVE)
5	UINT16	10-11	-	-	nReserved	Reserved bytes

An axis start will only be initiated if the distance from the target point is in fact larger than the parameterised braking distance.

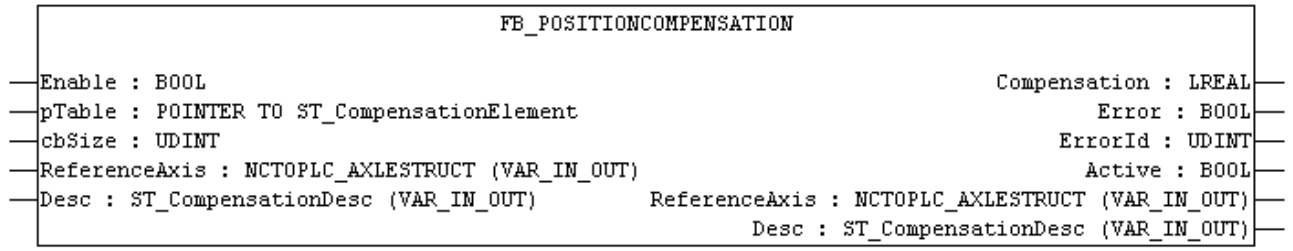
6.3 "Low Cost" stepper motor axis with digital control (stepper)

Drive interface for "Low Cost" stepper motor axes NC->IO (12 bytes)

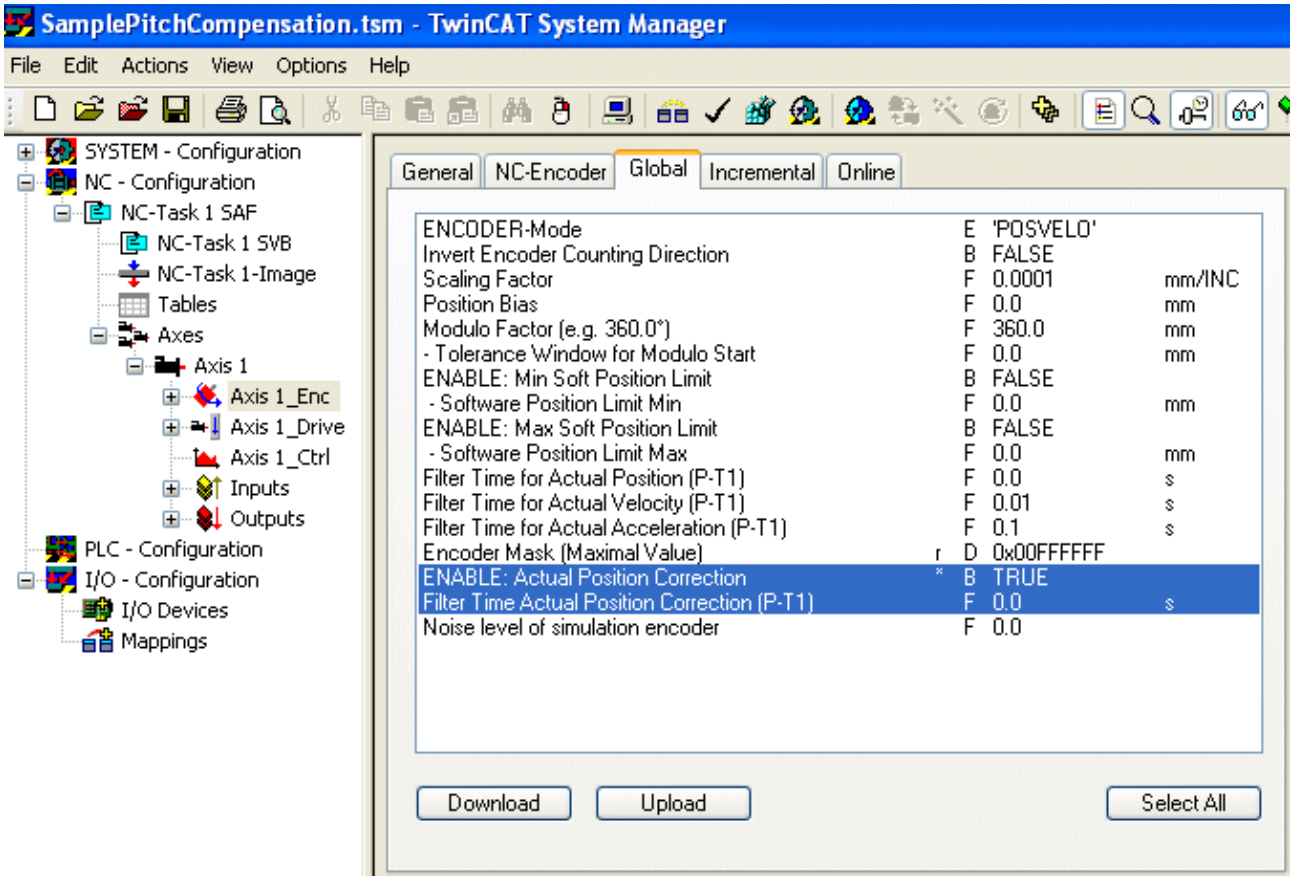
No	Data type	Byte	Bit	Def. Range	Variable Name	Description
1	INT32	0-3	-	-	nOutData1	Drive output data 1 (NC->IO)
2	INT32	4-7	-	-	nOutData2	Drive output data 2 (NC->IO)
3	UINT8	8	-	-	nControlByte	Control byte
3.0	...	8	0	0/1	bPhaseA	Phase A
3.1	...	8	1	0/1	bPhaseAInv	Phase A inverse
3.2	...	8	2	0/1	bPhaseB	Phase B
3.3	...	8	3	0/1	bPhaseBInv	Phase B inverse
3.4	...	8	4	0/1	-	RESERVED
3.5	...	8	5	0/1	-	RESERVED
3.6	...	8	6	0/1	bBreakInv	Inverse braking bit (0 ≡ ACTIVE, 1 ≡ PASSIVE)
3.7	...	8	7		bBreak	Braking bit (1 ≡ ACTIVE, 0 ≡ PASSIVE)
4	UINT8	9	-	-	nExtControlByte	Extended control byte
4.0	...	9	0	0/1	bFrequency	Frequency (square wave signal)
4.1	...	9	1	0/1	bDirectionPlus	Direction: positive
4.2	...	9	2	0/1	-	RESERVED
4.3	...	9	3	0/1	-	RESERVED
4.4	...	9	4	0/1	-	RESERVED
4.5	...	9	5	0/1	-	RESERVED
4.6	...	9	6	0/1	-	RESERVED
4.7	...	9	7	0/1	-	RESERVED
5	UINT16	10-11	-	-	nReserved	Reserved bytes

6.4 Example Pitch Compensation

[This is preliminary documentation and subject to change.]



This sample should illustrate how to use FB_PositionCompensation [▶ 28] and FB_WritePositionCorrection [▶ 26] for a pitch compensation of a spindle. Depending on the required accuracy it is recommended to run the PLC task with the same cycle time as the NC-SAF task. It is also necessary to enable the actual position correction of the axis (System Manager).




```
VAR
   fbPitchCompensation      :FB_PositionCompensation;
   fbWritePosCorrection     :FB_WritePositionCorrection;
   bAxisIsHomed             : BOOL;
   bEnablePitchCompensation : BOOL := FALSE;
   fCompensationValue       : LREAL;
   bError                   : BOOL;
   nErrorId                 : UDINT;
   bActive                  : BOOL;
   bLimiting                : BOOL;
END_VAR

VAR CONSTANT
   stXDescPitch: ST_CompensationDesc :=
(fPosMin:=0.0, fPosMax:=100.0, nTableElements:=11);
   stXPitchTable: ARRAY[0..10] OF ST_CompensationElement
:=(fPos:=0.0, fCompensation:=0.0),
  (fPos:=10.0, fCompensation:=0.1),
```

```
  (fPos:=20.0, fCompensation:=0.2),
  (fPos:=30.0, fCompensation:=0.3),
  (fPos:=40.0, fCompensation:=0.4),
  (fPos:=50.0, fCompensation:=0.5),
  (fPos:=60.0, fCompensation:=0.6),
  (fPos:=70.0, fCompensation:=0.7),
  (fPos:=80.0, fCompensation:=0.8),
  (fPos:=90.0, fCompensation:=0.9),
  (fPos:=100.0, fCompensation:=1.0);
END_VAR
```

The compensation table and description is defined as constant in this case. fPos holds the uncorrected absolute position and for each position there is a compensation value fCompensation.

In almost all applications the compensation is just useful if the axis is referenced (homed). So FB_PositionCompensation should only be enabled if this is given.

```
IF bAxisIsHomed THEN
   (* generally the compensation is just allowed if the axis is referenced *)
   (* so we check here, if the axis is homed & enable the pitch compensation if so *)
   bEnablePitchCompensation := TRUE;
ELSE
   bEnablePitchCompensation := FALSE;
END_IF
fbPitchCompensation(
   Enable:= bEnablePitchCompensation ,
   pTable:= ADR(stXPitchTable),
   cbSize:= SIZEOF(stXPitchTable),
   ReferenceAxis:= in_stXNcToPlc,
   Desc:= stXDescPitch,
   Compensation=>fCompensationValue,
   Error=>bError,
   ErrorId=>nErrorId,
   Active=>bActive);

fbWritePosCorrection(
   Enable:=TRUE,
   PositionCorrectionValue:=fCompensationValue,
   CorrectionMode:= POSITIONCORRECTIONMODE_FAST,
   Acceleration:= 500,
   CorrectionLength:= ,
   AxisRefIn:= in_stXNcToPlc,
   AxisRefOut:= out_stXPlcToNc,
   Busy=> bLimiting,
   Error=> ,
   ErrorID=> ,
   Limiting=> );
```

Requirements

Development environment	Target system type	PLC libraries to be linked
from TwinCAT v2.10 Build 1314	PC (i386)	TcNc.Lib

Trademark statements

More Information:
www.beckhoff.com/tx1200

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com