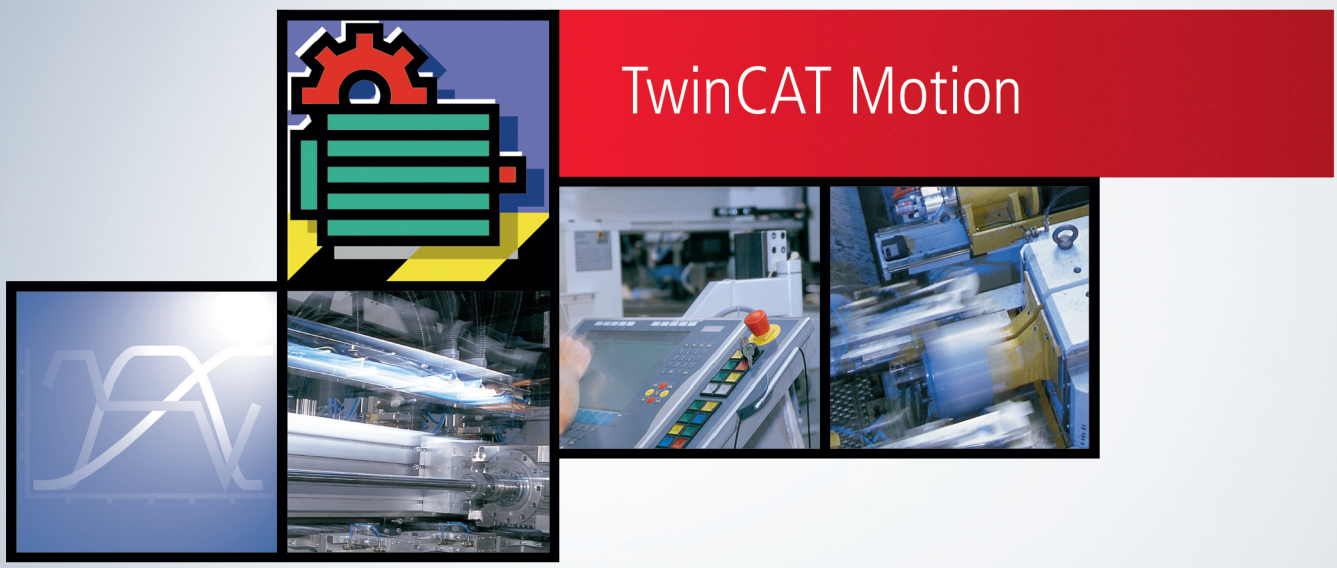
BECKHOFF New Automation Technology

Handbuch | DE

PLC-Bibliothek: TcNC

TwinCAT 2 | TX1200, PlcNc, TcNcUtilities



06.10.2021 | Version: 1.0

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT®

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Erklärung der Symbole

In der vorliegenden Dokumentation werden die folgenden Symbole mit einem nebenstehenden Sicherheitshinweis oder Hinweistext verwendet. Die Sicherheitshinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt oder Geräten

Wenn der Hinweis neben diesem Symbol nicht beachtet wird, können Umwelt oder Geräte geschädigt werden.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

2 Übersicht

Die Kommunikation zwischen SPS und NC erfolgt auf zwei Arten:

- Zyklisches Interface von der SPS zur NC [▶ 64] (Freigaben, Override usw.) und von der NC zur SPS [▶ 55] (Istwerte, Stati usw.): Austausch der, in jedem SPS-Zyklus benötigten Informationen, über das zyklische Prozessabbild.
- Funktionsbausteine: Es werden Funktionsbausteine (im Folgenden kurz NC-FB genannt) im Sinne der IEC1131 bereitgestellt, in denen jeweils thematisch verwandte Funktionalitäten zusammengefasst sind. Die NC-FBs sind als Firmware-Bausteine implementiert, d.h. sie sind Teil der Steuerungssoftware und in ihrem Verhalten fest definiert. Die Bausteine verfügen über Ein- und Ausgänge, deren Datentypen ausschließlich elementare (also keine abgeleiteten) IEC1131-Datentypen sind.

Die Verwendung der NC-FBs erfolgt durch das Konzept der Instanziierung:

Der SPS-Programmierer erzeugt bei Bedarf eine Variable (Instanz) von dem gewünschten Baustein und kann diese Instanz dann parametriert aufrufen.

Vorgesehen ist eine Kommunikation zwischen der SPS- und NC-Achsen, sowie zwischen SPS und Steuerungskanälen. In beiden Fällen ist ein direkter Austausch auf der Ebene von Prozessabbildern für Daten realisiert, die ständig verfügbar sein müssen. Funktionsaufrufe werden durch entsprechende Bausteine ausgelöst, die einen Datenaustausch über ADS-Dienste verwirklichen.

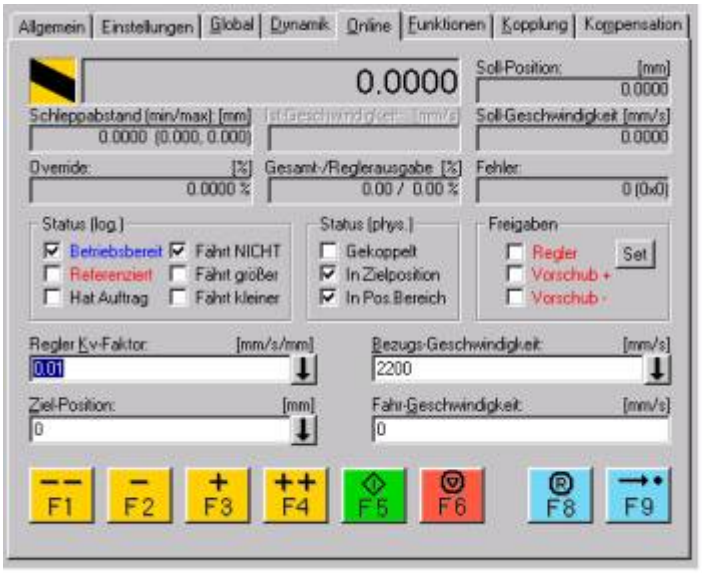
Auf die genannten Bausteine kann mit allen verfügbaren SPS-Sprachen zugegriffen werden.

Die NC-Architektur der Anlage wird im System Manager abgebildet: Es werden für die NC-Kanäle und -Achsen eingetragen bzw. optional noch spezielle Gruppen (Interpolation). Diese Elemente müssen anschließend parametriert werden. Somit ist es möglich, durch Anfügen im Baum eine neue NC-Achse oder einen neuen NC-Kanal einer bestehenden Konfiguration hinzuzufügen (freie Skalierbarkeit der NC). Es ist möglich, alle vorhandenen Achsen einzeln zu betreiben oder sie zu zweit oder zu dritt in Gruppen zu platzieren, um interpolierendes Verfahren zu ermöglichen. Auch ist es beispielsweise möglich, einzelne Achsen als Slave zu einer beliebigen anderen Achse (Master) zu koppeln.

Die durch einen Rechner maximal unterstützte Anzahl von NC-Elementen (Achsen, Gruppen, Kanäle) ergibt sich aus der zur Verfügung stehenden Rechenleistung. Aufgrund der Softwarestruktur stehen in der maximalen Ausbaustufe bis zu 255 Achsen zur Verfügung.

Die Inbetriebnahme von Achsen geschieht mit Hilfe der jeweiligen Dialoge im Bereich der NC-Konfiguration des System Managers.

Beispiel



Weiterführende Informationen entnehmen Sie bitte der Dokumentation oder Online-Hilfe des System Managers.

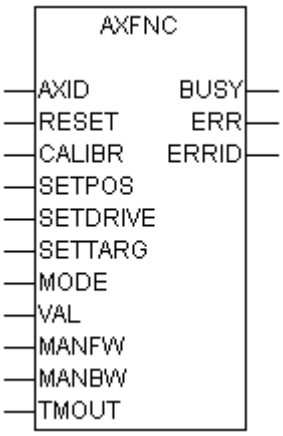
Funktionen der SPS-Bibliothek

Die SPS-Bibliothek fasst nützliche Funktionen und Funktionsbausteine zur Programmierung von Achssteuerungen zusammen. Die Bibliothek enthält Bausteine, die universell einsetzbar sind. Bausteine für Sonderfunktionen der NC finden sich in weiteren spezialisierten Bibliotheken.

3 Funktionsbausteine

3.1 Achs-Interface

3.1.1 AXFNC (Achsfunktionen)



In diesem Baustein sind Funktionen zusammengefasst, die benötigt werden, um eine Achse beim Anlagenstart nach Fehlern für den regulären Betrieb vorzubereiten. Von den Funktionen dieses Bausteins darf immer nur eine aktiv sein !

Bei mehreren gesetzten bzw. zurückgesetzten Eingängen gibt es folgende Prioritäten und es wird die Funktion mit der höchsten Priorität ausgeführt:

- Fallende Flanke an SETDRIVE (höchste Priorität);
- Fallende Flanke an MANFW;
- Fallende Flanke an MANBW;
- Steigende Flanke an SETDRIVE;
- Steigende Flanke an MANFW;
- Steigende Flanke an MANBW;
- Steigende Flanke an RESET;
- Steigende Flanke an CALIBR;
- Steigende Flanke an SETPOS;
- Steigende Flanke an SETTARG (niedrigste Priorität);

Eingänge

Der Baustein besitzt folgende Eingänge:

Eingang	Datentyp	Beschreibung
AXID	INT	ID der Achse (s. System Manager)
RESET	BOOL	Versetzt die Achse, soweit logisch und physikalisch möglich, in einen fehlerfreien Grundzustand. Darf nicht während einer Fahrt angestoßen werden, da die Achse sonst schlagartig gestoppt wird.
CALIBR	BOOL	Stößt bei einer positiven Flanke auf diesem Eingang das Referenzieren der angewählten Achse an.
SETPOS	BOOL	Setzt den Istwert einer Achse auf den in 'Val' angegebenen Wert, unter Beachtung der in ‚Mode' angegebenen Setzart.

Eingang	Datentyp	Beschreibung
SETDRIVE	BOOL	Bei einer positiven Flanke wird der in 'Val' angegebene Ausgabewert (üblicherweise ein Spannungswert) nach Maßgabe des Mode-Parameters auf dem der Achse zugeordneten Regler ausgegeben. Die Achse wird somit nicht mehr geregelt gefahren. Die Reglerfreigabe muss aber während dieser Funktion durchgehend gesetzt bleiben. Die Ausgabe erfolgt, bis eine negative Flanke auf ‚SetDrive' auftritt.
SETTARG	BOOL	Ändert die Zielposition einer Achse in laufender Fahrt. Die neue Zielposition muss in ‚Val' angegeben werden. Die Zielposition wird nach Angabe des ‚Mode'-Parameters interpretiert.
MODE	DWORD	Modus s.u.
VAL	LREAL	enthält je nach ausgeführter Funktion den dazu benötigten Wert: **SetPos:** Neuer Istwert (Mode beachten !) **SetDrive:** Wert für direkte Drive-Ausgabe (Mode beachten !) **SetTargP:** Wert für neue Zielposition (Mode beachten !) **ManFw:** Wert für die Handfahrgeschwindigkeit vorwärts **ManBw**: Wert für die Handfahrgeschwindigkeit rückwärts
MANFW	BOOL	Startet die Achse bei einer positiven Flanke als Endlosfahrt in positiver Geberzählrichtung, bei negativer Flanke wird gestoppt. Die Software-Endschalter sind, sofern nicht über die Achskonfiguration abgewählt, wirksam. An ‚Val' muss die gewünschte Fahrgeschwindigkeit angelegt werden. Die Achse kann alternativ auch über eine Instanz des Bausteins AXACT oder AXACTEX gestoppt werden.
MANBW	BOOL	Startet die Achse bei einer positiven Flanke als Endlosfahrt in negativer Geberzählrichtung, bei negativer Flanke wird gestoppt. Die Software-Endschalter sind, sofern nicht über die Achskonfiguration abgewählt, wirksam. An ‚Val' muss die gewünschte Fahrgeschwindigkeit angelegt werden. Die Achse kann alternativ auch über eine Instanz des Bausteins AXACT oder AXACTEX gestoppt werden.
TMOUT	TIME	ADS Timeout-Delay

Ausgang	Datentyp	Beschreibung
BUSY	BOOL	Dieser Ausgang bleibt solange auf TRUE, bis der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der, an dem 'Timeout'-Eingang angelegten, Zeit. Während Busy = TRUE wird an den Eingängen kein neuer Befehl angenommen. Bitte beachten Sie, dass nicht die Ausführung des Dienstes, sondern nur dessen Annahme zeitlich überwacht wird.
ERR	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in 'ErrorId' enthalten. Wenn der Baustein ein Timeout-Fehler hat, so ist 'Error' = TRUE und 'ErrorId' = 1861 (Hexadezimal 0x745). Wird durch das Ausführen eines Befehls an den Eingängen auf FALSE zurückgesetzt..
ERRID	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das Ausführen eines Befehls an den Eingängen auf 0 zurückgesetzt. Die Fehlernummern in ErrId können in der ADS Fehlerdokumentation

MODE: enthält je nach ausgeführter Funktion einen Modus, der die Wirkung dieser Funktion genauer spezifiziert:

Define	Istwert Setz-Typen
1	Absolut
2	Relativ (±Verfahrweg)
3	Reserviert

Define	Istwert Setz-Typen
4	Reserviert
5	Modulo (auch größer als der Modulofaktor möglich)

Define	Drive-Ausgabe
1	Drive Output in % im Bereich [-100%, 100%] des Maximalstellbereichs
2	Drive Output als absolute Ausgabegeschwindigkeit (z.B. mm/s)

Define	Neue Zielposition
1	Absolut
2	Relativ (±Verfahrweg)
3	Reserviert
4	Reserviert
5	Modulo (auch größer als der Modulofaktor möglich)

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.2 AXACT (Achsaktionen)

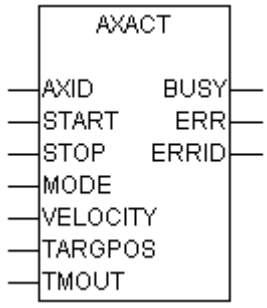


Abb. 1: axact

Der STOP-Eingang hat eine höhere Priorität als START. Wenn gleichzeitig eine positive Flanke an START und STOP gelegt wurde, wird ein Stoppbefehl an die Achse gesendet.

Der Baustein besitzt folgende Eingänge:

Eingang	Datentyp	Beschreibung
AXID	INT	Achsen-Id
START	BOOL	Mit einer positiven Flanke an diesem booleschen Eingang wird ein Startbefehl an die Achse gesendet. Sobald der BUSY-Ausgang des Bausteins auf FALSE geht, ist das Bit ‚Achse hat Fahrauftrag' in dem zyklischen Prozessabbild gesetzt und die NC hat den Start der Achse angenommen; damit ist jedoch nicht gemeint, dass die Achse sich schon physikalisch in Bewegung gesetzt hat oder am Ziel angekommen ist. Erst

Eingang	Datentyp	Beschreibung
		wenn das Bit ‚Achse hat Fahrauftrag' zu Null wird und in der Zwischenzeit kein Achsfehler aufgetreten ist, wird die logische Positionierung der Achse abgeschlossen. Ist ein Zielpositionsfenster gesetzt, so kann das Erreichen dieses Zielpositionsfenster als Erreichen der physikalischen Position gewertet werden.
STOP	BOOL	Mit einer positiven Flanke an diesem booleschen Eingang wird ein Stoppbefehl an die Achse gesendet. Erst wenn das Bit ‚Achse hat Fahrauftrag' im zyklischen Prozeßabbild zu Null wird, ist die logische Positionierung der Achse abgeschlossen.
MODE	DWORD	Startmodus: Absolut, Endlos, Modulo s.u. **Achtung**: Es werden nicht alle Starttypen von allen Achstypen unterstützt !
VELOCITY	LREAL	Der Parameter enthält die geforderte Fahrgeschwindigkeit für einen nachfolgenden Fahrauftrag z.B. mm/s..
TARGPOS	LREAL	Zielposition in physikalischen Größen z.B. mm, Grad
TMOUT	TIME	ADS Timeout-Delay

Ausgang	Datentyp	Beschreibung
BUSY	BOOL	Dieser Ausgang bleibt so lange auf TRUE, bis der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der, an dem 'Timeout'-Eingang angelegten, Zeit. Während Busy = TRUE wird an den Eingängen kein neuer Befehl angenommen. Bitte beachten Sie, dass nicht die Ausführung des Dienstes, sondern nur dessen Annahme zeitlich überwacht wird.
ERR	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in 'ErrorId' enthalten. Wenn der Baustein ein Timeout-Fehler hat, so ist 'Error' = TRUE und 'ErrorId' = 1861 (Hexadezimal 0x745). Wird durch das Ausführen eines Befehls an den Eingängen auf FALSE zurückgesetzt..
ERRID	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls.Wird durch das Ausführen eines Befehls an den Eingängen auf 0 zurückgesetzt. Die Fehlernummern in ErrId können in der ADS Fehlerdokumentation

Mode: Starttypen (1D)

unterstützter Achstyp	Define	Istwert Setz-Typen
alle	1	Absolut (±Verfahrweg)
alle	2	Relativ (±Verfahrweg)
alle	3	Endlos positiv
alle	4	Endlos negativ
alle	5	Modulo (±Verfahrweg) (auch größer als der Modulofaktor möglich)
nur Servoachsen	272	Endlos positiv mit Handgeschwindigkeit langsam
nur Servoachsen	528	Endlos positiv mit Handgeschwindigkeit schnell
nur Servoachsen	784	Endlos positiv mit Eilgangsgeschwindigkeit
nur Servoachsen	288	Relativ positiv um Pulsweite
nur Servoachsen	544	Relativ positiv um 1/1000
nur Servoachsen	800	Relativ positiv um 1/100
nur Servoachsen	1056	Relativ positiv um 1/10
nur Servoachsen	1312	Relativ positiv um 1/1
nur Servoachsen	273	Endlos negativ mit Handgeschwindigkeit langsam

unterstützter Achstyp	Define	Istwert Setz-Typen
nur Servoachsen	529	Endlos negativ mit Handgeschwindigkeit schnell
nur Servoachsen	785	Endlos negativ mit Eilgangsgeschwindigkeit
nur Servoachsen	289	Relativ negativ um Pulsweite
nur Servoachsen	545	Relativ negativ um 1/1000
nur Servoachsen	801	Relativ negativ um 1/100
nur Servoachsen	1057	Relativ negativ um 1/10
nur Servoachsen	1313	Relativ negativ um 1/1

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.3 AXACTEX (Achsaktionen erweitert)

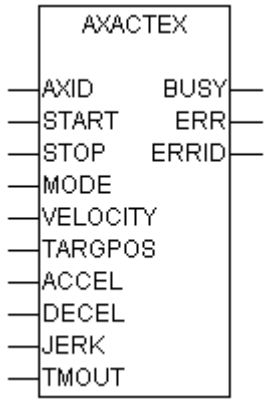


Abb. 2: axactex

Der STOP-Eingang hat eine höhere Priorität als START. Wenn gleichzeitig eine positive Flanke an START und STOP gelegt wurde, wird ein Stoppbefehl an die Achse gesendet.

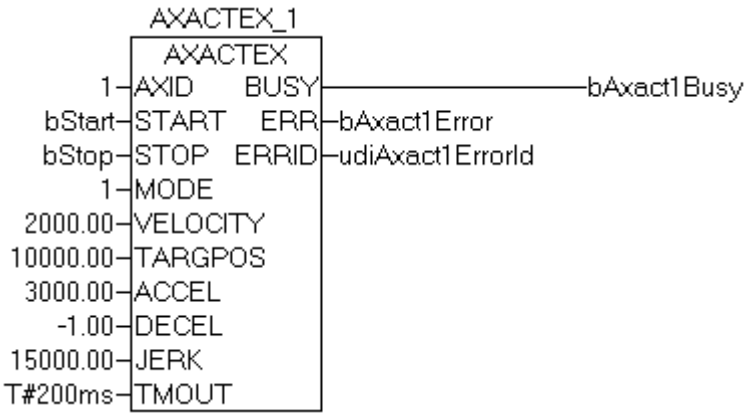
Der Baustein besitzt folgende Eingänge:

Eingang	Datentyp	Beschreibung
AXID	INT	Achsen-Id
START	BOOL	Mit einer positiven Flanke an diesem boolschen Eingang wird ein Startbefehl an die Achse gesendet. Sobald der BUSY-Ausgang des Bausteins auf FALSE geht, ist das Bit ‚Achse hat Fahrauftrag' in dem zyklischen Prozessabbild gesetzt und die NC hat den Start der Achse angenommen; damit ist jedoch nicht gemeint, dass die Achse sich schon physikalisch in Bewegung gesetzt hat oder am Ziel angekommen ist. Erst wenn das Bit ‚Achse hat Fahrauftrag' zu Null wird und in der Zwischenzeit kein Achsfehler aufgetreten ist, wird die logische Positionierung der Achse abgeschlossen. Ist ein Zielpositionsfenster gesetzt, so kann das Erreichen dieses Zielpositionsfenster als Erreichen der physikalischen Position gewertet werden.
STOP	BOOL	Mit einer positiven Flanke an diesem boolschen Eingang wird ein Stoppbefehl an die Achse gesendet. Erst wenn das Bit ‚Achse hat Fahrauftrag' im zyklischen Prozessabbild zu Null wird, ist die logische Positionierung der Achse abgeschlossen.

Eingang	Datentyp	Beschreibung
MODE	DWORD	Startmodus: Absolut, Endlos, Modulo s.u.
VELOCITY	LREAL	Der Parameter enthält die geforderte Fahrgeschwindigkeit für einen nachfolgenden Fahrauftrag z.B. mm/s..
ACCEL	LREAL	Beschleunigung z.B. mm/s²
DECEL	LREAL	Verzögerung z.B. mm/s²
JERK	LREAL	Ruck z.B. mm/s³
TARGPOS	LREAL	Zielposition in physikalischen Größen z.B. mm, Grad
TMOUT	TIME	ADS Timeout-Delay

Ausgang	Datentyp	Beschreibung
BUSY	BOOL	Dieser Ausgang bleibt solange auf TRUE, bis der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der, an dem 'Timeout'-Eingang angelegten, Zeit. Während Busy = TRUE wird an den Eingängen kein neuer Befehl angenommen. Bitte beachten Sie, dass nicht die Ausführung des Dienstes, sondern nur dessen Annahme zeitlich überwacht wird.
ERR	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in 'ErrorId' enthalten. Wenn der Baustein ein Timeout-Fehler hat, so ist 'Error' = TRUE und 'ErrorId' = 1861 (Hexadezimal 0x745). Wird durch das Ausführen eines Befehls an den Eingängen auf FALSE zurückgesetzt..
ERRID	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das Ausführen eines Befehls an den Eingängen auf 0 zurückgesetzt. Die Fehlernummern in ErrId können in der ADS Fehlerdokumentation

Beispiel

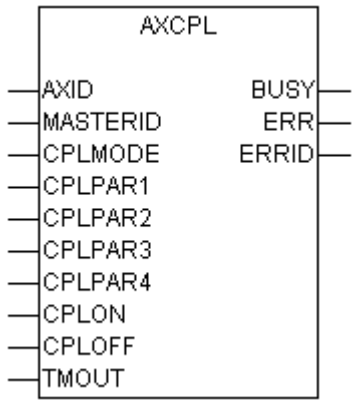


Das Bild zeigt eine Instanz "AXACTEX_1" des Bausteins AXACTEX, mit dem Achse Nr.1 gestartet und gestoppt werden kann. Die Zielposition beträgt 10000.00 mm, die Geschwindigkeit 2000 mm/s. Zusätzlich wird durch diesen Baustein die Beschleunigung (3000 mm/s2) und der Ruck (15000 m/s3) vorgegeben, wobei als Verzögerung der Standardwert aus der Achskonfiguration genommen wird (spezielle Kennung "-1").

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.4 AXCPL (Achskopplung)



Der Baustein wird benötigt, um eine beliebige Servo-Achse als Slave an eine beliebige andere Servo-Achse zu koppeln. Je nach Kopplungsart kann die Kopplung im Stillstand oder bei fahrender Master-Achse erfolgen. Eine Slave-Achse darf nach erfolgter Kopplung nicht mehr mit Fahrbefehlen angesprochen werden, da sie ihre Selbständigkeit verloren hat und erst durch ein Entkoppeln wiedergewinnt. Die einfachste Kopplungsart ist die Linearkopplung mit einem festen Übersetzungsverhältnis (elektronisches Getriebe), diese Kopplung ist auch mit negativen Getriebefaktor möglich.

Der CPLOFF-Eingang hat eine höhere Priorität als CPLON. Wenn gleichzeitig eine positive Flanke an CPLOFF und CPLON angelegt wurde, wird die Kopplung deaktiviert.

HINWEIS

Wenn der Sollwertgeneratortype der Achse auf "7 Phasen (optimiert)" eingestellt ist, wird die Slaveachse nach dem Abkoppeln beschleunigungsfrei gefahren und mit der sich einstellenden konstanten Geschwindigkeit weitergefahren.
Es erfolgt keine Positionierung um den mit dem Koppelfaktor umgerechneten Masterverfahrweg, sondern es stellt sich ein Verhalten wie nach einem MC_MoveVelocity ein.
In TwinCAT 2.10 ist der Sollwertgeneratortyp wählbar.
Ab TwinCAT 2.11 ist der Sollwertgeneratortype fest auf "7 Phasen (optimiert)" eingestellt.
Bei der Umstellung eines Projektes von TwinCAT 2.10 auf TwinCAT 2.11 ergibt sich damit das hier beschriebene Verhalten.
Ein Update bestehender Applikationen auf Version 2.11 kann daher, je nach Anwendung, eine Anpassung des SPS-Programms erforderlich machen.

Der Baustein besitzt folgende Eingänge:

Eingang	Datentyp	Beschreibung
AXID	INT	Id der Slaveachse
MASTERID	INT	Id der Masterachse
CPLMODE	INT	Kopplungsart
CPLPAR1	LREAL	Kopplungsparameter 1
CPLPAR2	LREAL	Kopplungsparameter 2
CPLPAR3	LREAL	Kopplungsparameter 3
CPLPAR4	LREAL	Kopplungsparameter 4
CPLON	BOOL	Kopplung aktivieren
CPLOFF	BOOL	Kopplung deaktivieren
TMOUT	TIME	ADS Timeout-Delay

Ausgang	Datentyp	Beschreibung
BUSY	BOOL	Dieser Ausgang bleibt solange auf TRUE, wie der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der, an dem 'Timeout'-Eingang angelegten, Zeit. Während Busy = TRUE wird an den

Ausgang	Datentyp	Beschreibung
		Eingängen kein neuer Befehl angenommen. Bitte beachten Sie, dass nicht die Ausführung des Dienstes, sondern nur dessen Annahme zeitlich überwacht wird.
ERR	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in 'ErrorId' enthalten. Wenn der Baustein ein Timeout-Fehler hat, so ist 'Error' = TRUE und 'ErrorId' = 1861 (Hexadezimal 0x745). Wird durch das Ausführen eines Befehls an den Eingängen auf FALSE zurückgesetzt..
ERRID	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das Ausführen eines Befehls an den Eingängen auf 0 zurückgesetzt. Die Fehlernummern in ErrId können in der ADS Fehlerdokumentation

CPLMODE

Define	Kopplungsart	
1	**"Lineare Kopplung (Geradengleichung)"** Lineare Kopplung mit konstantem Getriebeverhältnis, das sowohl positive als auch negative Werte annehmen darf, allerdings nur Werte ungleich Null. Anmerkung: Ebenfalls kann das Getriebeverhältnis bei einer bestehenden Kopplung online per ADS geändert werden. Da diese Änderung allerdings abrupt wirkt, darf der Getriebefaktor nur um sehr kleine Beträge innerhalb seiner Nachkommastellen verändert werden (Verstimmung).	
	CplPara1':	Getriebefaktor
	CplPara2':	reserviert
	CplPara3':	reserviert
	CplPara4':	reserviert
2	**"Fliegende Säge Kopplung (Diagonal-Kopplung) auf Geschwindigkeit"** Diagonal synchronisiertes Aufkoppeln ("Fliegende Säge") mit ruckbegrenztem Geschwindigkeitsprofil. Masterverfahrweg während der Slavebeschleunigungsphase ist gleich dem doppelten Slaveverfahrweg während der Slavebeschleunigungsphase.	
	CplPara1':	reserviert
	CplPara2':	reserviert
	CplPara3':	Neigungswinkel der Diagonalsäge in Bezug auf die Orthogonale zur Masterstrecke [Grad]. Der Winkel muss sich im Wertebereich von echt größer 0 Grad und kleiner gleich 90 Grad (parallel) befinden.
	CplPara4':	Getriebefaktor (falls 0, dann wird diese Variable mit 1.0 vorbelegt)
3	**"Fliegende Säge Kopplung (Diagonal-Kopplung) auf Position und Geschwindigkeit"** Diagonal synchronisiertes Aufkoppeln ("Fliegende Säge") mit ruckbegrenztem Geschwindigkeitsprofil. Der Masterverfahrweg während der Slavebeschleunigungsphase ist gleich dem doppelten Slaveverfahrweg während der Slavebeschleunigungsphase.	
	CplPara1':	Absolute Master-Synchronposition [mm]
	CplPara2':	Absolute Slave-Synchronposition [mm]
	CplPara3':	Neigungswinkel der Diagonalsäge in Bezug auf die Orthogonale zur Masterstrecke [Grad]. Der Winkel muss sich im Wertebereich von echt größer 0 Grad und kleiner gleich 90 Grad (parallel) befinden.
	CplPara4':	Getriebefaktor (falls 0.0, dann wird diese Variable mit 1.0 vorbelegt)
12	**"Fliegende Säge Modulo Kopplung (Beschleunigungsbegrenzt)"** Diese Kopplungsart ist speziell für Moduloachsen (eine sich periodisch fortsetzende Bewegung, wobei die Periode nicht zwingend 360.0 Grad entsprechen muss) entwickelt	

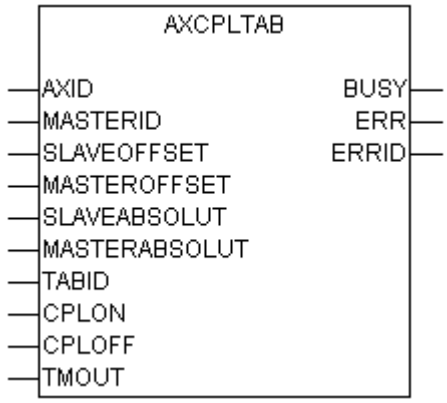
Define	Kopplungsart	
	worden. Das besondere an diesem Slavetyp ist, dass die Slaveachse erstmalig verschiedene Unterbetriebsarten besitzt, wobei ein extern angeforderter Wechsel von der einen in die andere Betriebsart jederzeit möglich ist. Die Slaveachse beantwortet dies mit einer Zwischenphase, die ein Aufsynchronisieren auf die angeforderte Betriebsart bedeutet. Dieses Aufsynchronisieren ist als Streckensteuerung implementiert, d. h. in weiten Grenzen unbeeinflusst von den Veränderungen der Masterdynamik. Das Aufsynchronisieren bedeutet bei dieser Modulo-Kopplung, dass bezogen auf die Moduloperiode auf kürzestem Wege (± ½ Periode) eine feste Orientierung hergestellt wird.	
	CplPara1':	Getriebefaktor (momentan zwingend mit 1.0 vorausgesetzt !)
	CplPara2':	Maximale Slavebeschleunigung in Prozent bezogen auf das Minimum von Beschleunigung und Verzögerung. Wertebereich: >0.0 bis ... 1.0
	CplPara3':	Masterachs-ID 2 für Tabellenkopplung
	CplPara4':	Optionale Tabellen-ID (Tabellentypen: äquidistant zyklisch und nichtäquidistant zyklisch) Funktion noch nicht freigegeben !
15	**"Lineare Kopplung mit zyklisch variablem Getriebefaktor"** Bei einer zyklischen Slave-Achse wird der Getriebefaktor über das Achsinterface von der SPS vorgegeben (s. Variable "fAxisModeLReal" in der Struktur PLCTONC_AXLESTRUCT) und kann in jedem SPS-Zyklus geändert werden. Diese Änderung wird dann automatisch durch den zyklischen Datenaustausch des Achsinterfaces von der SPS zur NC übertragen. Um extreme Sprünge des Getriebefaktors und damit sprungartig hohe Beschleunigungen zu vermeiden, lässt sich die bei einer Getriebefaktoränderung resultierende Beschleunigung des Slaves über einen Begrenzungsparameter p_a limitieren. Der Begrenzungsparameter wird bei der Achskopplung durch den ersten Koppelparameter festgelegt.	
	CplPara1':	entspricht indirekt, nämlich bezogen auf eine maximale Mastergeschwindigkeit, einer maximal erlaubten Beschleunigung (p_a = $a_{SlaveMax}$ / $v_{MasterMax}$). Der Begrenzungsparameter p_a entspricht dabei dem Kehrwert der Hochlaufzeit t_H = 1 / p_a .
	CplPara2':	reserviert
	CplPara3':	reserviert
	CplPara4':	reserviert
16	**"Bi-Lineare Kopplung"** Die Bi-Lineare Kopplung ist vergleichbar mit der Linearen Kopplung (Koppelmode: 1), allerdings sind hier zeitgleich zwei Linearkopplungen über den Getriebefaktor 1 und Getriebefaktor 2 (zur zusätzlichen Masterachs-ID 2) wirksam. Anwendungsfall könnte das Gewindebohren/Gewindeschneiden sein, wo ein linearer Vorschub zusammen mit einer rotatorischen Bewegung koordiniert werden müssen. *Anmerkung:* Wenn die Hauptmasterachse 1 nicht logisch in Bewegung ist, dann werden unabhängig von der Bewegungsphase der Masterachse 2 keine Slavesollwerte berechnet. Ebenfalls sei in diesem Zusammenhang angemerkt,, dass eine Achse, die mit dem Geschwindigkeitsoverride 0% verfährt, ebenfalls als logisch in Bewegung gilt!	
	CplPara1':	Getriebefaktor 1 (Hauptgetriebefaktor)
	CplPara2':	Getriebefaktor 2 (Nebengetriebefaktor)
	CplPara3':	Masterachs-ID 2 für den Getriebefaktor 2
	CplPara4':	reserviert
18	**"Kopplung mit konstante Oberflächengeschwindigkeit und zyklisch variablem Getriebefaktor"** Dieser Typ beinhaltet die mathematische Rechnung für eine Kopplung zwischen einer rotatorischen Slaveachse und einer translatorischen Masterachse. Zweck ist, für die rotatorisch eingemessene und geregelte Slaveachse in Abhängigkeit ihres Trommeldurchmessers eine konstante Oberflächengeschwindigkeit (Umfangsgeschwindigkeit) bezogen auf die Masterachse herzustellen und nachzuregeln. Der Trommelradius dieser Slaveachse wird mittels eines zweiten Encoders (Nebenencoder), der für die Slaveachse im Systemmanager konfiguriert sein muss,	

Define	Kopplungsart	
	automatisch in jedem NC-SAF-Zyklus ausgewertet und für die Berechnung verwendet. Ebenfalls kann und darf dieser Radius niemals den Wert 0.0 mm besitzen, da sonst eine Berechnung nicht mehr möglich ist. Zusätzlich wird über das zyklische Achsinterface (s. "Lineare Kopplung mit zyklisch variablem Getriebefaktor") ein Getriebefaktor g(t) vorgegeben (s. Variable "fAxisModeLReal" in der Struktur PLCTONC_AXLESTRUCT), der im trivialsten Fall konstant den Wert 1.0 haben kann (keine weitere Beeinflussung). Wenn also die Slaveachse mit ihrem Hauptencoder auf Grad und ihrem Nebenencoder (Radiuserfassung r(t)) auf mm eingemessen ist und der Master als translatorische Achse in mm, dann ergibt sich die Berechnung der Slavegeschwindigkeit nach folgender Formel: $v_{Slave} = 360° / (2PI * r(t)) * g(t) * v_{Master}$.	
	'CplPara1':	entspricht indirekt, nämlich bezogen auf eine maximale Mastergeschwindigkeit, einer maximal erlaubten Beschleunigung ($p_a = a_{SlaveMax} / v_{MasterMax}$). Der Begrenzungsparameter p_a entspricht dabei dem Kehrwert der Hochlaufzeit $t_H = 1 / p_a$.
	'CplPara2':	reserviert
	'CplPara3':	reserviert
	'CplPara4':	reserviert

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.5 AXCPLTAB (Tabellen-Achskopplung)



Der Baustein wird benötigt, um eine beliebige Servo-Achse als Slave an eine beliebige andere Servo-Achse zu koppeln. Die Kopplung geschieht in diesem Fall in Form einer Tabelle, die wiederum aus einer vorgebbaren Anzahl von Stützpunkten besteht (Ortskurve).

Bei den Tabellen unterscheidet man zwischen vier verschiedenen Typen:

- Äquidistante Lineare Tabellen,
- Äquidistante Zyklische Tabellen,
- Monotone Lineare Tabellen und
- Monotone Zyklische Tabellen.

Diese Art der Kopplung kann nur im Stillstand erfolgen. Eine Slave-Achse darf nach erfolgter Kopplung nicht mehr mit Fahrbefehlen angesprochen werden, da sie ihre Selbständigkeit verloren hat und erst durch ein Entkoppeln wiedergewinnt. Der CPLOFF-Eingang hat eine höhere Priorität als CPLON. Wenn gleichzeitig eine positive Flanke an CPLOFF und CPLON angelegt wurde, wird die Kopplung deaktiviert.

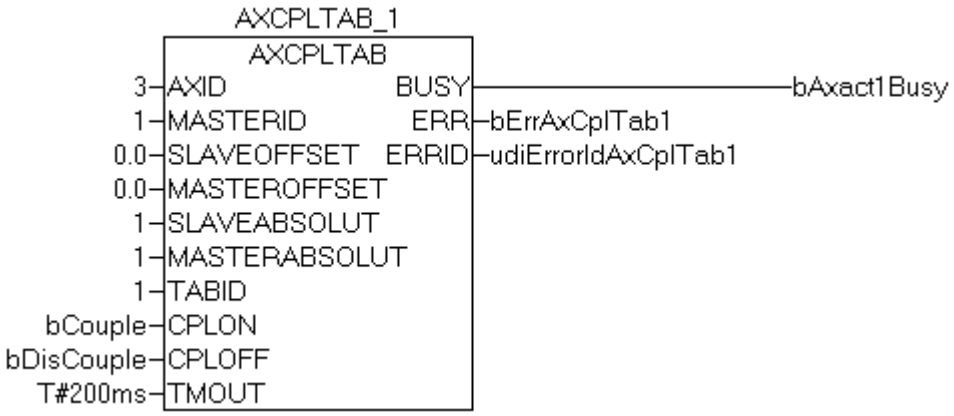
Der Baustein besitzt folgende Eingänge:

Eingang	Datentyp	Beschreibung
AXID	UINT	Achsen-Id
MASTERID	UINT	Id der Master-Achse
SLAVEOFFSET	LREAL	Für die Interpretation der Slaveposition in der Koppel-Tabelle kann Positionsoffset angegeben werden, der im Standardfall 0.0 ist. Die physikalische Einheit entspricht der Basis-Längeneinheit der Achse (z. B. mm).
MASTEROFFSET	LREAL	Für die Interpretation der Masterposition in der Koppel-Tabelle kann Positionsoffset angegeben werden. der im Standardfall 0.0 ist. Die physikalische Einheit entspricht der Basis-Längeneinheit der Achse (z. B. mm).
SLAVEABSOLUT	UDINT	Interpretation der Slave-Tabellenpositionen als absolute oder relative Positionen 1 : Absolut 0 : Relativ
MASTERABSOLUT	UDINT	Interpretation der Master-Tabellenpositionen als absolute oder relative Positionen 1 : Absolut 0 : Relativ
TABID	UDINT	Hier wird die systemweite gültige Tabellen-Id angegeben. Eine Tabellenkopplung setzt voraus, dass diese Tabelle mit der entsprechenden Dimension bereits existiert und mit gültigen Werten gefüllt worden ist. Hierbei wird zwischen vier verschiedenen Tabellentypen unterschieden: • Äquidistante Lineare Tabellen (Typ: 1) • Äquidistante Zyklische Tabellen (Typ: 2) • Monotone Lineare Tabellen (Typ: 3) • Monotone Zyklische Tabellen (Typ: 4)
CPLON	BOOL	Kopplung ein
CPLOFF	BOOL	Kopplung aus
TMOUT	TIME	ADS Timeout-Delay

Ausgang	Datentyp	Beschreibung
BUSY	BOOL	Dieser Ausgang bleibt solange auf TRUE, bis der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der, an dem 'Timeout'-Eingang angelegten, Zeit. Während Busy = TRUE wird an den Eingängen kein neuer Befehl angenommen. Bitte beachten Sie, dass nicht die Ausführung des Dienstes, sondern nur dessen Annahme zeitlich überwacht wird.
ERR	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in 'ErrorId' enthalten. Wenn der Baustein ein Timeout-Fehler hat, so ist 'Error' = TRUE und 'ErrorId' = 1861 (Hexadezimal 0x745). Wird durch das Ausführen eines Befehls an den Eingängen auf FALSE zurückgesetzt..

Ausgang	Datentyp	Beschreibung
ERRID	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das Ausführen eines Befehls an den Eingängen auf 0 zurückgesetzt. Die Fehlernummern in ErrId können in der ADS Fehlerdokumentation

Beispiel



Das Bild zeigt eine Instanz "AXCPLTAB_1" des Bausteins AXCPLTAB, mit dem die Achse mit der ID 3 an die Achse mit der ID 1 (Masterachse) gekoppelt bzw. entkoppelt werden kann (steigende Flanke des Zustands der Variable "bCouple" bzw. "bDiscouple"). Die Positionssollwerte des Masters und des Slaves werden aus einer vorliegenden, binären Tabelle entnommen. Im Beispiel wird eine äquidistante lineare Tabelle verwendet (TabId=1). Positionsoffsets sind im Beispiel beim Master und beim Slave mit 0.0 vordefiniert. Es folgt ein Beispiel einer äquidistanten, linearen Tabelle im ASCII-Format mit 81 Zeilen (Masterposition von 0.0 bis 80.0 mm bzw. Grad; Slaveposition von 0.0 bis 279.72 mm bzw. Grad).

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

Tabellentyp:

Äquidistante lineare Tabelle mit 81 Wertepaaren (Stützstellen)

Erste Zeile:

Anzahl Zeilen (n=81) und Anzahl Spalten (m=2)

Folgende Zeilen:

n-Zeilen mit Masterposition (absolut oder relativ) und zugehöriger Slaveposition (absolut oder relativ)

Anmerkung:

Zwischen den Stützstellen findet in jedem SAF-Zyklus der Achse eine lineare Interpolation statt. Die Tabellengröße und die Diskretisierung in der Position ist frei wählbar.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.6 AXSCOM (Achsenstreckenkompensation)

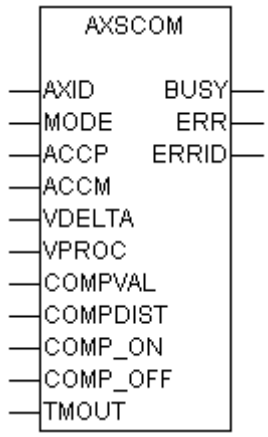


Abb. 3: AXSCOM

Der Baustein wird benötigt, um bei einer fahrenden Achse (Master oder Slave) auf einer vorgegebenen Strecke eine angebbare Wegstrecke aufzuholen oder zu verzögern. Dieses geschieht über eine, von der Achspositioniersoftware ausgerechneten, zeitbegrenzten Geschwindigkeitserhöhung oder Geschwindigkeitsverminderung. Der COMP_OFF-Eingang hat eine höhere Priorität als COMP_ON. Wenn gleichzeitig eine positive Flanke an COMP_OFF und COMP_ON angelegt wurde, wird die Kompensation deaktiviert.

Der Baustein besitzt folgende Eingänge:

Eingang	Datentyp	Beschreibung
AXID	INT	Achsen-Id
MODE	UDINT	Kompensationsmodus s.u.
ACCP	LREAL	Max. Beschleunigungswert
ACCM	LREAL	Max. Verzögerungswert
VDELTA	LREAL	Max. erlaubte Geschwindigkeitsänderung
VPROC	LREAL	Grundgeschwindigkeit des Prozesses
COMPVAL	LREAL	Kompensationswert, um den aufgeholt bzw. verzögert werden soll
COMPDIST	LREAL	Kompensationsstrecke, die zur Kompensation zur Verfügung steht
COMP_ON	BOOL	Kompensation ein
COMP_OFF	BOOL	Kompensation aus
TMOUT	TIME	ADS Timeout-Delay

Ausgang	Datentyp	Beschreibung
BUSY	BOOL	Dieser Ausgang bleibt solange auf TRUE, bis der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der, an dem 'Timeout'-Eingang angelegten, Zeit. Während Busy = TRUE wird an

Ausgang	Datentyp	Beschreibung
		den Eingängen kein neuer Befehl angenommen. Bitte beachten Sie, dass nicht die Ausführung des Dienstes, sondern nur dessen Annahme zeitlich überwacht wird.
ERR	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in 'ErrorId' enthalten. Wenn der Baustein ein Timeout-Fehler hat, so ist 'Error' = TRUE und 'ErrorId' = 1861 (Hexadezimal 0x745). Wird durch das Ausführen eines Befehls an den Eingängen auf FALSE zurückgesetzt.
ERRID	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das Ausführen eines Befehls an den Eingängen auf 0 zurückgesetzt. Die Fehlernummern in ErrId können in der ADS Fehlerdokumentation

Mode

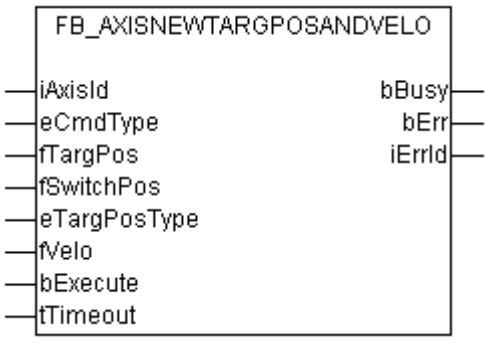
Define	Kompensations-Profil
1	Trapezförmiges Geschwindigkeitsprofil

Achtung: Wenn die Kompensation mit den geforderten Parametern nicht komplett durchgeführt werden kann, wird der Start mit dem NC-Fehlercode "0x4243" beantwortet. Die Rückmeldung ist nur als Warnung anzusehen, da die Kompensation, soweit wie unter den Randbedingungen möglich, trotzdem ausgeführt wird.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PlcNc.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

3.1.7 FB_AxisNewTargPosAndVelo



Mit dem Funktionsbaustein "FB_AixsNewTartAndVelo" kann die Zielposition und die Geschwindigkeit einer Achse während der Fahrt verändert werden.

Mit der Funktion AxisIsMoving [▶ 40] kann festgestellt werden, ob sich die Achse bewegt. Sollte sie sich nicht bewegen, so führt dieses Kommando zu einem Fehler. Weiterhin sollte das Kommando nicht sehr kurz vor dem Ende der Positionierung abgesetzt werden, weil sonst die Achse beim Eintreffen des Kommandos bereits abgeschlossen sein könnte.

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: Achs-ID der Achse

eCmdType: Der Typ (E_CmdTypeNewTargPosAndVelo [▶ 67]) des Kommandos legt fest, ob Zielposition, Geschwindigkeit oder beide Werte der aktuellen Fahrt geändert werden. Weiterhin legt dieser Parameter fest, ob die Änderung instantan oder erst an der Umschaltschwelle fSwitchPos wirkt.

fTargPos: Zielposition der aktuellen Fahrt der Achse.

fSwitchPos: Optionale Umschaltposition bei deren Erreichen das Kommando wirkt.

eTargPosType: Typ der Zielposition (E_TargPosType [▶ 68]). Die relative Positionierung sollte in diesem Zusammenhang nicht verwendet werden. Die Zielposition wäre sonst abhängig von der Position an der das Kommando aktiviert wird und somit nicht exakt.

fVelo: Neue Geschwindigkeit, mit der zur Zielposition gefahren werden soll.

bExecute: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

tTimeout: Timeout bis zur Quittierung des Kommandos durch die NC.

VAR_OUTPUT

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

bBusy: Bei der Aktivierung des Funktionsbausteins wird dieser Ausgang gesetzt und bleibt gesetzt bis eine Rückmeldung erfolgt.

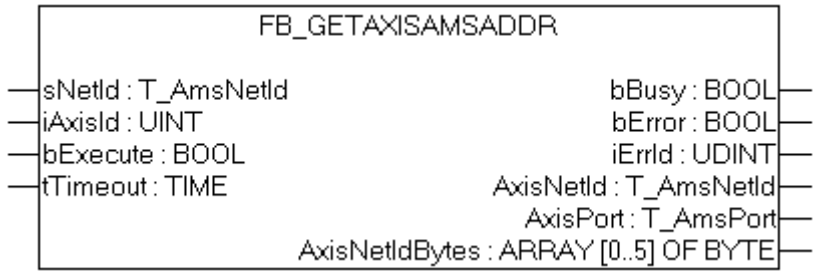
bErr: Ist ein Fehler bei der Ausführung des Kommandos aufgetreten, dann wird dieser Ausgang gesetzt.

iErrId: Liefert bei einem gesetzten bErr-Ausgang die Fehlernummer.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.8	PC (i386)	TcNc.Lib

3.2 FB_GetAxisAmsAddr



Mit dem Funktionsbaustein kann die ADS-Adresse (NetId und Port), der mit der Achse verknüpften IO-Hardware gelesen werden.

VAR_INPUT

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

sNetid : Hier kann die Netzwerkadresse des TwinCAT-Rechners angegeben werden, auf dem die Adressinformation der Achse gelesen werden soll. Für die Achsen auf dem lokalen Rechner kann auch ein Leerstring angegeben werden.

iAxisId: Achs-ID der Achse.

bExecute: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

tTimeout: Timeout bis zur Quittierung des Kommandos durch die NC.

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: Bei der Aktivierung des Funktionsbausteins wird dieser Ausgang gesetzt und bleibt gesetzt bis eine Rückmeldung erfolgt.

bError: Ist ein Fehler bei der Ausführung des Kommandos aufgetreten, dann wird dieser Ausgang gesetzt.

iErrId: Liefert bei einem gesetzten bError-Ausgang die Fehlernummer.

AxisNetId: Die ADS-NetId der NC Achs-Hardware.

AxisPort: Die ADS-Portnummer der NC-Achs-Hardware.

AxisNetIdBytes: Die ADS-NetId als Bytearray.

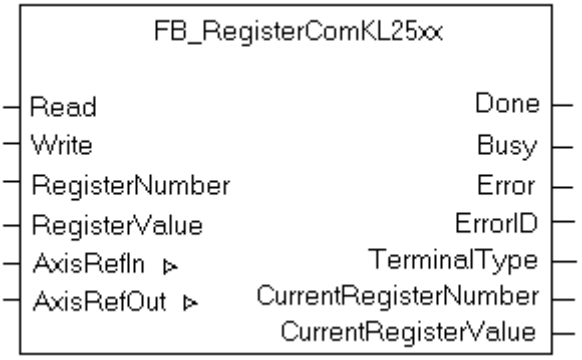
AxisChannel: Die Kanalnummer der NC-Achs-Hardware bei mehrkanaligen Geräten.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.8 Build > 746	PC (i386)	TcNc.Lib

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9 Build > 947		

3.3 FB_RegisterComKL25xx



Der Funktionsblock FB_RegisterComKL25xx wird zur Registerkommunikation zwischen SPS und den Busklemmen KL2502, KL2521, KL2531, KL2541 und KL5001 verwendet..

Bevor der Funktionsblock vom oder in das Register schreibt bzw. liest, wird die AxisID, EncoderID und DriveID gelesen, mit der die Klemmenvariablen gemappt sind. Darüber hinaus unterbricht der Funktionsblock die Verbindung zwischen Antrieb o/p und Klemme, so dass der NC task Zugriff verhindert wird. Von jetzt an besteht keine Kommunikation mehr zwischen dem Ausgang des Antriebs und der Klemme, solange dieser Funktionsblock ausgeführt wird. Dem schließt sich das Register Lesen oder Schreiben an. Der NC Task Zugriff ist wiederhergestellt, sobald die Registerkommunikation stattfindet (Antriebsausgang ist aktiviert).

Voraussetzungen:

Es ist notwendig, die Prozessdaten der Klemmenvariablen mit dem Encoder und den Antriebsvariablen der entsprechenden Achse zu mappen, indem der dazugehörige Encoder und Antriebstyp ausgewählt wird. Basierend auf das Mapping zwischen IO Klemmen und NC Variablen können die Klemmen in zwei Gruppen aufgeteilt werden: Einmal in die Gruppe mit den Klemmen KL2531 und KL2541 und zum anderen mit KL2502 und KL2521.

Das erforderliche Mapping zwischen Busklemme und NC Variablen:

```
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]
```

Hinweis :

Wurde durch eine positive Flanke am Eingang *Read* (Lesen) und am Eingang *Write* (Schreiben) ein Registerkommunikationszyklus ausgelöst, schreibt der Funktionsbaustein zunächst den vorgegebenen *RegisterValue* (Registerwert) in die spezifizierte *RegisterNumber* (Registernummer) und liest den Registerwert aus dieser *RegisterNumber* wieder aus. Dies wird nicht als Fehler angesehen.

Auslesen und Schreiben des Registers erfolgt vom bzw. in dem EEPROM. Beim Schreiben in ein Register gibt es eine Ausnahme: Es ist mit diesem Funktionsbaustein nicht möglich ins Herstellerregister zu schreiben. Dies würde zu einer Fehlernummer 0x4B41 führen.

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 : Der Befehl Lesen wird bei einer positiven Flanke an diesem Eingang ausgeführt.

Write : Der Befehl Schreiben wird bei einer positiven Flanke an diesem Eingang ausgeführt.

RegisterNumber : In dieser Variablen wird die zu schreibende/lesende Registernummer spezifiziert.

RegisterValue : Wenn Eingang *Write* =TRUE, spezifiziert diese Variable den zu schreibenden Registerwert. Wenn Eingang *Read* = TRUE, wird der Wert in dieser Variablen nicht berücksichtigt.

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 : Wird TRUE, wenn eine Registerkommunikation erfolgreich ausgeführt worden ist. Die geschriebene/ausgelesene Registernummer wird in *CurrentRegisterNumber* (aktuelle Registernummer) spezifiziert, der entsprechende Wert in *CurrentRegisterValue* (aktueller Registerwert).

Busy: Wird TRUE, sobald der Funktionsbaustein aktiv ist und FALSE, wenn er in den ursprünglichen Zustand zurückgekehrt.

Error : Wird TRUE, sobald ein Fehler auftritt.

ErrorID : Wenn der Fehlerausgang gesetzt ist, liefert dieser Parameter die Fehlernummer.

TerminalType : Wenn Done=TRUE enthält dieser Ausgang den entsprechenden Klemmentyp/die Klemmennummer, die in die Registerkommunikation involviert ist.

CurrentRegisterNumber : Wenn Done =TRUE gibt dieser Ausgang die Registernummer an, die bearbeitet wurde.

CurrentRegisterValue : Wenn Done =TRUE gibt dieser Ausgang den Registerwert des Registers an, das gerade bearbeitet wird.

VAR_IN_OUT

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

AxisRefIn : Achsstruktur aus der NC.

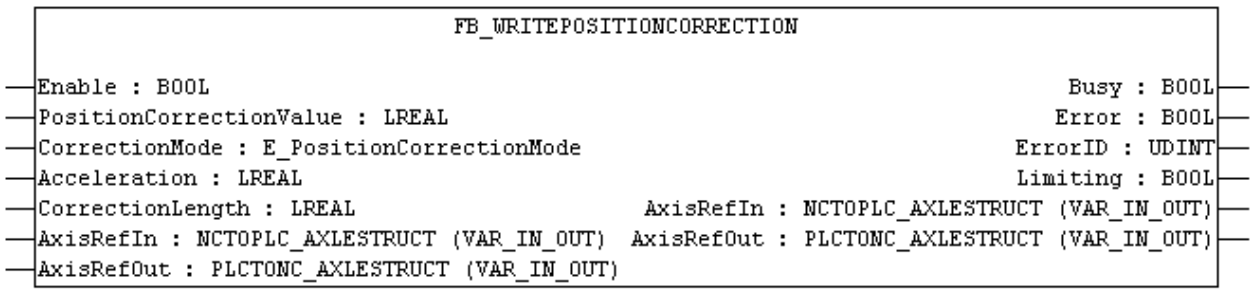
AxisRefOut : Achsstruktur aus der SPS.

Voraussetzungen

Entwicklungsumgebung	Zielsystem	Einzubindende SPS-Bibliotheken
ab TwinCAT v2.10 Build 1314	PC (i386)	TcNc.Lib

3.4 FB_WritePositionCorrection

[Dies ist die vorläufige Dokumentation, Änderungen sind vorbehalten.]



Der Funktionsbaustein FB_WritePositionCorrection schreibt ein Offset (PositionCorrectionValue) auf die Nennposition einer Achse. Abhängig vom Korrekturmodus werden die Daten unmittelbar oder „gefiltert“ auf das zyklische Achsinterface geschrieben.

VAR_INPUT

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

Enable: Das kontinuierliche Schreiben von PositionCorrectionValue wird durch eine steigende Flanke an diesem Eingang aktiviert. Er muss TRUE sein, solange neue Korrekturwerte akzeptiert werden sollen.

PositionCorrectionValue: Der Korrekturwert, der auf das zyklische Achsinterface geschrieben werden soll

CorrectionMode: Abhängig von diesem Modus wird der PositionCorrectionValue unmittelbar oder „gefiltert“ geschrieben. Für eine ausführliche Beschreibung siehe E_PosiitonCorrectionMode [▶ 68]

Acceleration: Abhängig vom CorrectionMode wird hier die maximale Beschleunigung zur Erreichung des neuen Korrekturwerts vorgegeben. Bei PositionCorrectionMode_Fast [▶ 27] hat dieser Wert einen unmittelbaren Einfluss auf das Positionsdelta per PLC-Tick.
Max. zulässiger Korrekturwert Positionsdelta = Beschleunigung * (SPS-Zykluszeit)^2

CorrectionLength: Wenn der CorrectionMode mit dem PositionCorrectionMode_FullLength übereinstimmt, wird dieser Parameter aktiv. Eine Änderung beim PositonCorrectionValue wird auf dieser Korrekturlänge aufgeteilt

VAR_IN_OUT

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

AxisRefIn : Achsstruktur der NC.

AxisRefOut : Achsstruktur der SPS.

VAR_OUTPUT

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

Busy: Wird TRUE, sobald der Funktionsbaustein aktiv ist, und FALSE wenn er in den ursprünglichen Zustand zurückgekehrt.

Error : Wird TRUE, sobald ein Fehler auftritt.

ErrorId : Wenn der Fehlerausgang gesetzt ist, liefert dieser Parameter die Fehlernummer.

Limiting: Wird TRUE, wenn der geforderte PositionCorrectionValue noch nicht vollständig akzeptiert ist.

Hinweis:

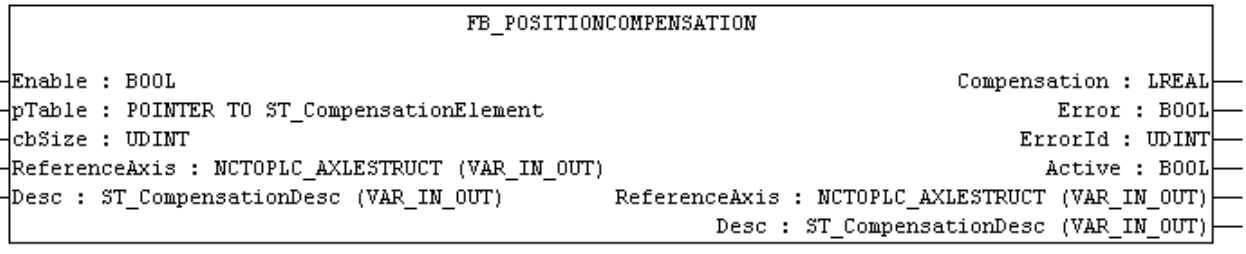
Um diesen Funktionsbaustein erfolgreich zu nutzen, muss Istpositionskorrektur im System Manager aktiviert werden.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
ab TwinCAT v2.10 Build 1330	PC (i386)	TcNc.Lib (Version >= 1.0.42)

3.5 FB_PositionCompensation

[Dies ist die vorläufige Dokumentation, Änderungen sind vorbehalten.]



Der Funktionsbaustein FB_PositionCompensation liefert den Ausgleichswert beispielsweise für den Steigungs- und Durchhangskompensation. Wenn der FB aktiviert ist, wird der Ausgleichswert durch die Position der Referenzachse und anhand der Korrekturtabelle (pTable) berechnet. Bei einer Steigungskompensation stimmen Referenzachse und kompensierte Achse überein. Wenn der FB für die Durchhangskompensation verwendet wird, sind sie unterschiedlich.

Der von diesem FB gelieferte Kompensationswert wird mittels FB_WritePositionCompensation auf die Achse geschrieben.

VAR_INPUT

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

Enable: Die kontinuierliche Berechnung des Kompensationswerts wird durch eine steigende Flanke an diesem Eingang aktiviert. Er muss TRUE sein, solange die Kompensationsdaten berechnet werden sollen.

pTable: Pointer auf Kompensationstabelle. Diese Tabelle ist ein Array vom Typ ST_CompensationElement.

cbSize: Größe der Kompensationstabelle in Bytes.

VAR_IN_OUT

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

ReferenceAxis: Achsstruktur von NC. Diese Achse kann sich von der zu kompensierenden Achse unterscheiden. Bei einer Durchhangskompensation beispielsweise stimmt sie nicht mit der kompensierten Achse überein.

Desc: Beschreibung Struktur [▶ 69] für die Kompensation.

VAR_OUTPUT

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

Compensation: Berechneter Kompensationswert. Kompensation beträgt 0.0 wenn Enable oder Active FALSE sind.

Error : Wird TRUE, sobald ein Fehler auftritt.

ErrorId : Wenn der Fehlerausgang gesetzt ist, liefert dieser Parameter die Fehlernummer.

Active: Wird TRUE, wenn die Kompensation aktiv ist. Wenn die Arbeitsrichtung nicht mit der aktuellen Richtung übereinstimmt, ist es FALSE.

Hinweis

Für ein Beispiel siehe „Beispiel Steigungskompensation [▶ 72]“

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
ab TwinCAT v2.10 Build 1314	PC (i386)	TcNc.Lib

4 Funktionen

4.1 Signale der NC auswerten

4.1.1 Statusmeldungen einer PTP-Achse

(Meldungen des zyklischen Achsinterface der Funktionsbausteine NC und MC_xxx)

Voraussetzungen

NcToPlc.nStateDword (DWORD), Variable der cyclic axis interface [▶ 55] (zyklisches Achsinterface) Struktur zwischen NC und SPS (NcToPlc), besteht aus verschiedenen Achsstatus-Flags. Das sind unter anderem:

AxisHasJob [▶ 39]

AxisIsMoving [▶ 40] (Achse bewegt sich vorwärts ODER Achse bewegt sich rückwärts)

AxisIsNotMoving [▶ 36]

AxisHasBeenStopped [▶ 39]

AxisIsAtTargetPosition [▶ 37]

AxisInPositionWindow [▶ 37]

HINWEIS:

Nur die Flags **AxisInPositionWindow** und **AxisIsAtTargetPosition** sind abhängig von den Istwerten (Istposition, Istgeschwindigkeit). Die anderen obengenannten Flags sind abhängig von den Sollwerten (Sollposition, Sollgeschwindigkeit, Sollbeschleunigung).

AxisHasJob [▶ 39] zeigt den korrekten und genauen Befehlsstatus der Achse an. Der Übergang von AxisHasJob von FALSE zu TRUE zeigt an, dass die Achse einen neuen Befehl und somit auch neue Parameter empfangen hat (wie Zielposition, Geschwindigkeit, Beschleunigung oder Ruck). Die Ausführung eines Achsbefehls wird durch den Status FALSE des Flags AxisHasJob angezeigt.

AxisIsMoving [▶ 40] wird TRUE, wenn der Fahrbefehl die Achse erreicht und der Sollwertgenerator aktiv ist. Wenn AxisIsMoving = TRUE ist, liegt die Sollwertgeschwindigkeit normalerweise nicht bei Null. Eine Ausnahme ist ein Geschwindigkeits-Override von 0%, der impliziert, dass die Achse einen Befehl erhalten hat und sich logisch mit der Geschwindigkeit Null bewegt (in diesem Fall sind AxisHasJob und AxisIsMoving TRUE, die Sollwertgeschwindigkeit ist Null). AxisHasJob schließt darüber hinaus die Kommunikationszeit für Anforderung und Reaktion auf den Befehl ein.

AxisIsNotMoving [▶ 36] hat immer den gegenteiligen Status von AxisIsMoving. Wenn AxisIsNotMoving = TRUE ist, steht die Achse logisch (aber nicht unbedingt physisch) still.

AxisHasBeenStopped [▶ 39] zeigt an, dass ein Stoppbefehl für die entsprechende Achse ausgeführt wurde. AxisHasBeenStopped wechselt auf TRUE, sobald der Stoppbefehl erteilt und ausgeführt worden ist und bleibt während der gesamten Stoppphase (solange der Stoppbefehl der Achse aktiv ist) TRUE. Er wird nur dann auf FALSE gesetzt, wenn ein neuer Befehl ausgeführt wird.

AxisIsAtTargetPosition [▶ 37] ist von zwei in den globalen Parametern der entsprechenden Achse im TwinCAT System Manager spezifizierten Parametern abhängig, *Target Position Window* und *Target Position Monitoring Time*. AxisIsAtTargetPosition ist TRUE, wenn die Istposition wenigstens für die Zeitspanne der *Target Position Monitoring Time* innerhalb des *Target Position Window* liegt (und dabei nicht aus dem *Target Position Window* ausbricht).

HINWEIS:

Dieses Flag muss im TwinCAT System Manager in den globalen Parametern der entsprechenden Achse unter dem Namen *Target Position Monitoring* aktiviert werden.

AxisInPositionWindow [▶ 37] ist abhängig vom *Position Range Window* -Parameter in den globalen Parametern der entsprechenden Achse im TwinCAT System Manager. Das Flag AxisInPositionWindow wird auf TRUE gesetzt, wenn sich die Istposition innerhalb der im *Position Range Window* spezifizierten Werte befindet; es wechselt auf FALSE, wenn sich der Wert der Istposition außerhalb dieses Fensters befindet.**HINWEIS:**

Dieses Flag muss im TwinCAT System Manager in den globalen Parametern der entsprechenden Achse unter dem Namen *Position Range Monitoring* aktiviert werden.

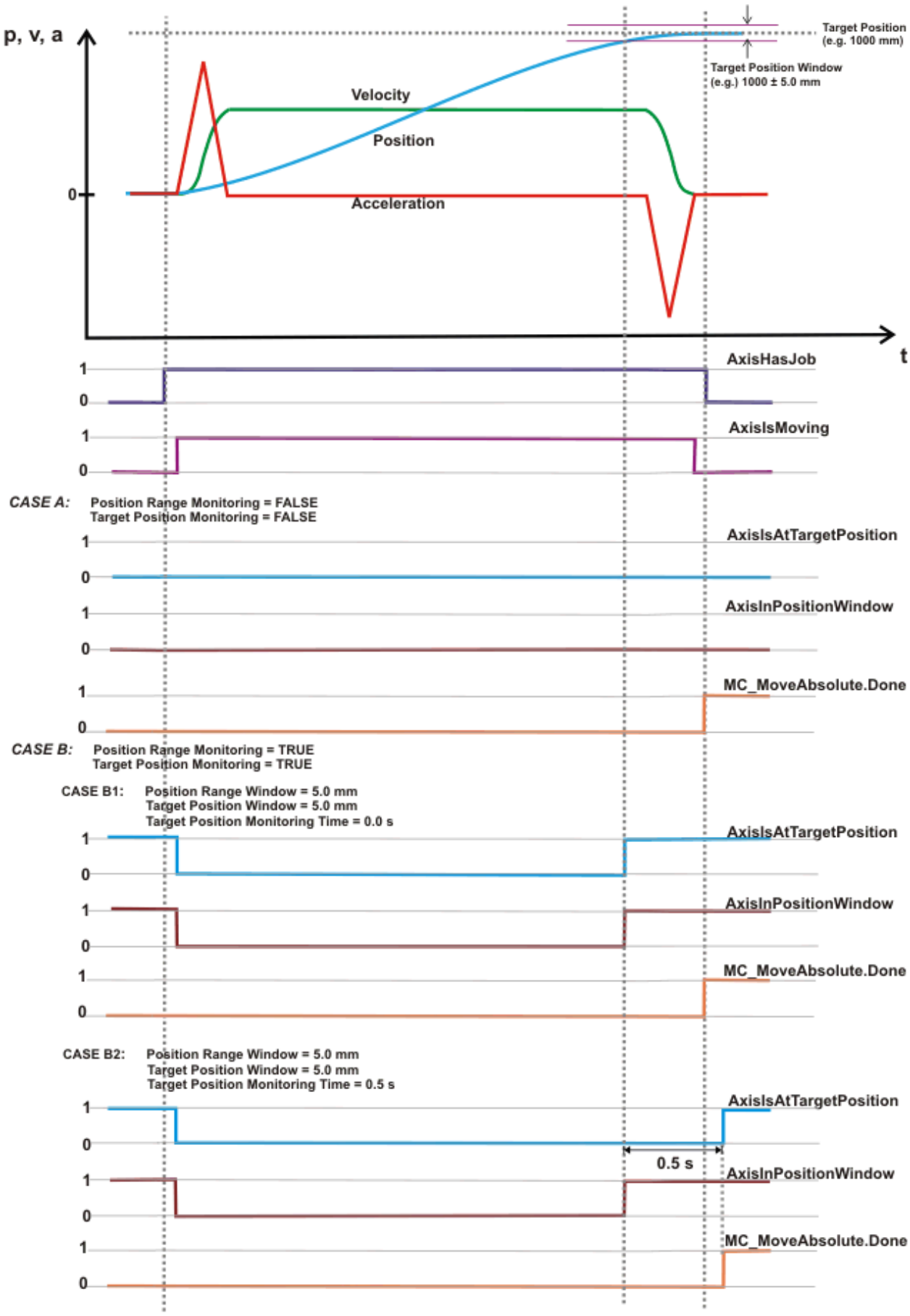
Nach der Positionierung prüfen alle ”MC_Move…”-Bausteine, ob die Positionierung erfolgreich ausgeführt wurde. Im einfachsten Fall wird das Flag „AxisHasJob“ der NC-Achse geprüft, was zunächst bedeutet, dass die Positionierung logisch ausgeführt wurde. Je nach Parametrierung der NC-Achse werden weitere Überprüfungen (Qualitätskriterien) durchgeführt:

- ”Target position monitoring”(Wichtig)
  Wenn die Zielpositionsüberwachung aktiv ist, wartet das System auf eine Rückmeldung der NC. Nach der Positionierung muss sich die Achse mindestens für die definierte Zeitspanne im spezifizierten Zielpositionsfenster befinden. Gegebenenfalls bewegt der Positionsregler die Achse in die Zielposition. Wird der Lageregler ausgeschaltet (Kv = 0) oder gibt nur ein schwaches Signal wird das Ziel möglicherweise nicht erreicht. Eine triftende Lageregelung kann dazu führen, dass die Achse um das Fenster herum oszilliert, jedoch nicht im Fenster bleibt.
- ”Position range monitoring”
  Ist die Positionsbereichsüberwachung aktiv, wartet das System auf eine NC-Rückmeldung. Nach der Positionierung muss sich die Achse im spezifizierten Positionsbereichsfenster befinden. Gegebenenfalls bewegt der Positionsregler die Achse in die Zielposition. Wird der Lageregler ausgeschaltet (Kv = 0) oder gibt nur ein schwaches Signal wird das Ziel möglicherweise nicht erreicht.

Befindet sich die Achse logisch auf Zielposition (logischer Stillstand), aber das parametrierte Positionsfenster wurde nicht erreicht, wird die Überwachung der obengenannten NC-Rückmeldung nach einem konstanten Timeout von 6 Sekunden mit Fehler 19207 (0x4B07) abgebrochen.

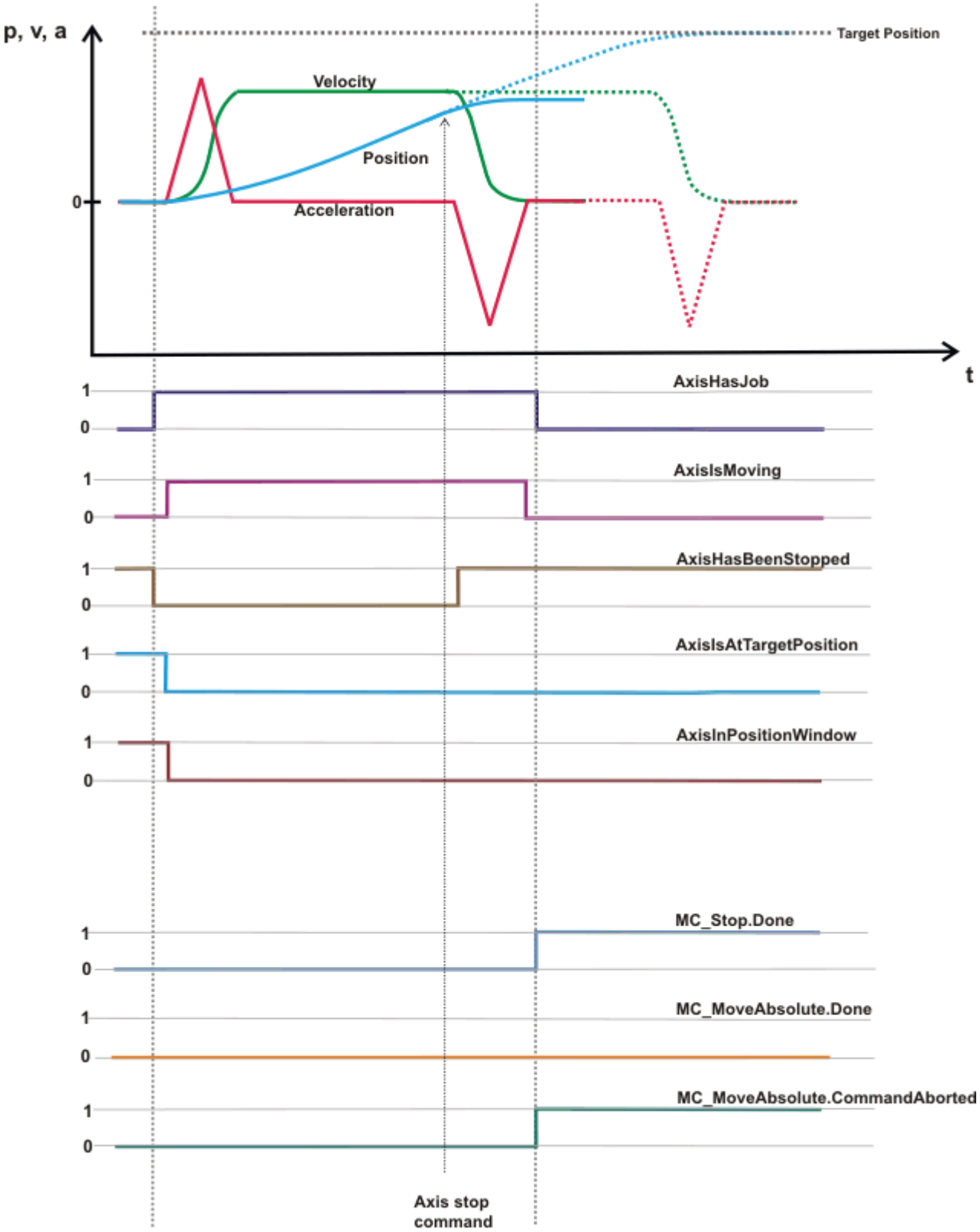
1.1. Positionierung mit MC_MoveAbsolute, MC_MoveRelative, ...

Es wurde eine normale PTP-Sequenz programmiert, um die Varianten der Meldungen in den drei verschiedenen Fällen zu zeigen (siehe unten). Diese sind abhängig von den Optionen Positionsbereichüberwachung und Zielpositionsüberwachung.


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

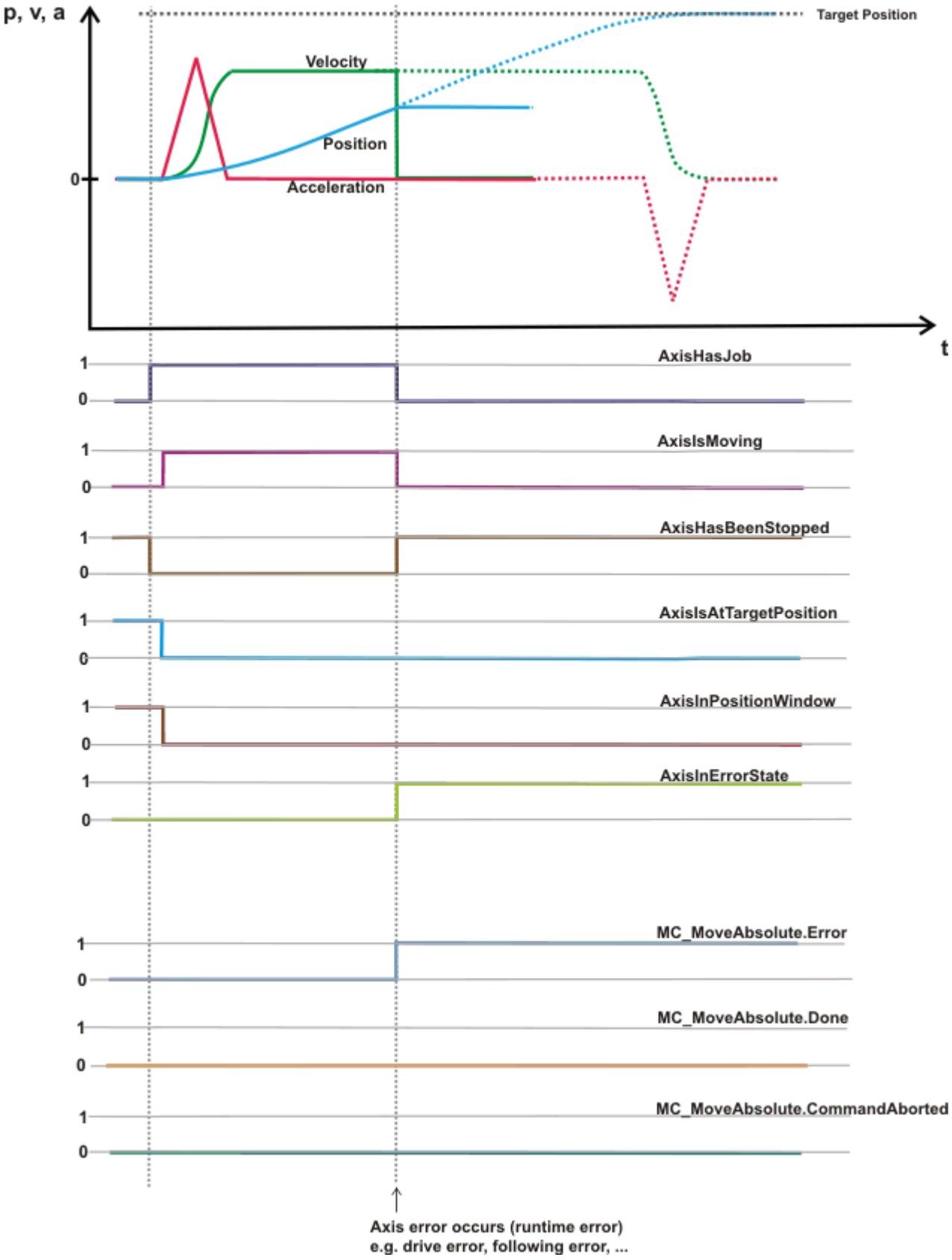
1.2. Anhalten (Abbrechen) einer aktivierten Positionierung mit MC_Stop.

Wird eine Achse während der Achspositionierung angehalten, kommt die Achse abhängig von Geschwindigkeit und Bremsverzögerung zum Stillstand.

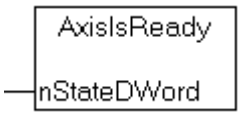


1.3. NC- oder Antriebfehler (Laufzeitfehler) während der Positionierung.

Tritt während der Positionierung ein NC- oder Antriebsfehler auf, springt die Geschwindigkeit auf Null. Somit bleibt die Position konstant und es ist bis zum Reset der Achse keine weitere Operation möglich.


p, v, a
Target Position
Velocity
Position
Acceleration
0
t
AxisHasJob
AxisIsMoving
AxisHasBeenStopped
AxisIsAtTargetPosition
AxisInPositionWindow
AxisInErrorState
MC_MoveAbsolute.Error
MC_MoveAbsolute.Done
MC_MoveAbsolute.CommandAborted
1
0
Axis error occurs (runtime error)
e.g. drive error, following error, ...

4.1.2 AxisIsReady



AxisIsReady liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsReady: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

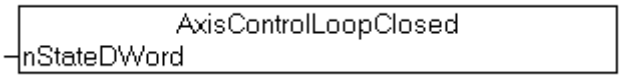
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Ready : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.3 AxisControlLoopClosed



AxisControlLoopClosed liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisControlLoopClosed: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

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

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

Voraussetzungen

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

4.1.4 AxisIsCalibrated

AxisIsCalibrated liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsCalibrated: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

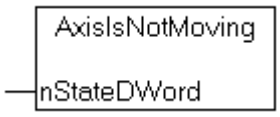
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Calibrated : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.5 AxisIsNotMoving



AxisIsNotMoving liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsNotMoving: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

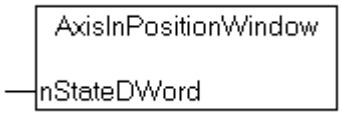
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Standstill : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.6 AxisInPositionWindow



AxisInPositionWindow liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisInPositionWindow: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

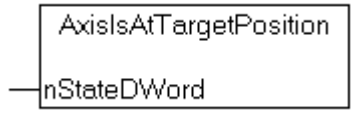
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    NearTarget : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.7 AxisIsAtTargetPosition



AxisIsAtTargetPosition liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsAtTargetPosition: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

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

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.8 AxisInProtectedMode

AxisInProtectedMode liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisInProtectedMode: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

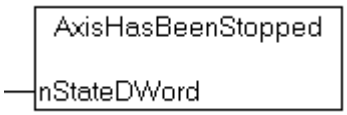
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Protected : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.9 AxisHasBeenStopped



AxisHasBeenStopped liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisHasBeenStopped: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

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

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.10 AxisHasJob

AxisHasJob liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisHasJob: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

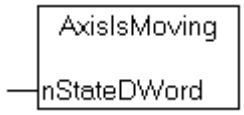
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    HasJob : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.11 AxisIsMoving



AxisIsMoving liefert TRUE, wenn eines der korrespondierenden Signale AxisIsMovingForward oder AxisIsMovingBackwards aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS gesetzt ist.

FUNCTION AxisIsMoving: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

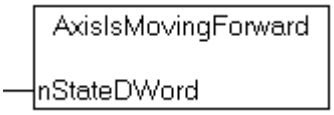
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Moving : BOOL;
END_VAR
```

```
Moving := AxisIsMoving( NcToPlc1.nStateDWord );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.12 AxisIsMovingForward



AxisIsMovingForward liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsMovingForward: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

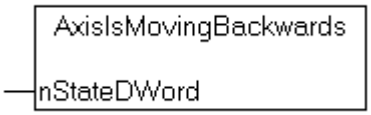
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Forward : BOOL;
END_VAR
```

```
Forward := AxisIsMovingForward( NcToPlc1.nStateDWord );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.13 AxisIsMovingBackwards



AxisIsMovingBackwards liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsMovingBackwards: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

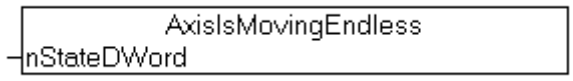
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Backwards : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.14 AxisIsMovingEndless



AxisIsMovingEndless liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsMovingEndless: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

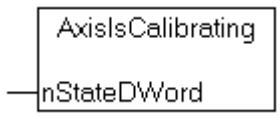
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Moving : BOOL;
END_VAR
```

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

Voraussetzungen

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

4.1.15 AxisIsCalibrating



AxisIsCalibrating liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsCalibrating: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

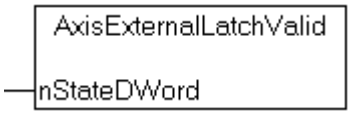
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Calibrating : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.16 AxisExternalLatchValid



AxisExternalLatchValid liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisExternalLatchValid : BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

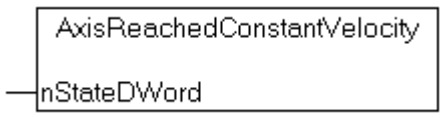
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    LatchValid : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.17 AxisReachedConstantVelocity



AxisReachedConstantVelocity liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisReachedConstantVelocity: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

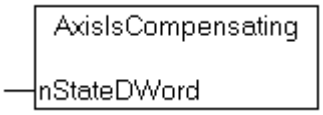
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    ConstVelocity : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.18 AxisIsCompensating



AxisIsCompensating liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisIsCompensating : BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

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

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.19 AxisHasExtSetPointGen

AxisHasExtSetPointGen liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisHasExtSetPointGen: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

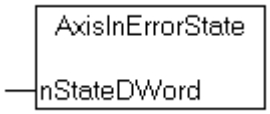
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    HasExtSetPointGen : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.20 AxisInErrorState



AxisInErrorState liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisInErrorState: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

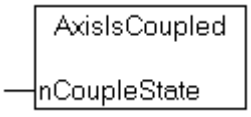
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000  : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000  : NCTOPLC_AXLESTRUCT;
    Error : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.21 AxisIsCoupled



AxisIsCoupled wertet den Koppelzustand aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS aus. **AxisIsCoupled** liefert TRUE, wenn nCoupleState>1 ist. Das bedeutet die Achse ist als Slave-Achse an eine Master-Achse gekoppelt.

FUNCTION AxisIsCoupled: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

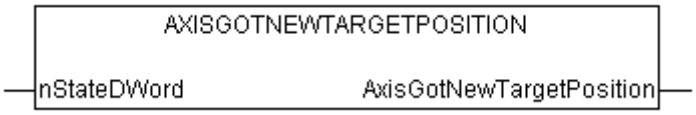
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    Coupled : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.1.22 AxisGotNewTargetPosition



Mit der Funktion kann erkannt werden ob während der regulären Achsfahrt eine neue Zielposition vorgegeben wurde (z.B. mit dem Funktionsbaustein FB_AxisNewTargPosAndVelo). Intern wird das Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS ausgewertet.

FUNCTION AxisGotNewTargetPosition: BOOL

```
VAR_INPUT

    nStateDWord     :
DWORD;

END_VAR
```

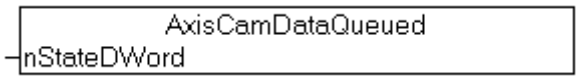
nStateDWord: Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS.

Rückgabeparameter	Beschreibung
TRUE	Eine neue Zielposition wurde vorgegeben.
FALSE	Es wurde keine neue Zielposition vorgegeben und beim Achs-Reset und -Neustart.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.8 (Build > 707)	PC (i386)	TcNc.Lib

4.1.23 AxisCamDataQueued



AxisCamDataQueued liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisCamDataQueued: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

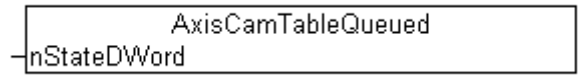
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    CamDataQueued : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9 ab Build 1025	PC (i386)	TcNC.Lib

4.1.24 AxisCamTableQueued



AxisCamTableQueued liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisCamTableQueued: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

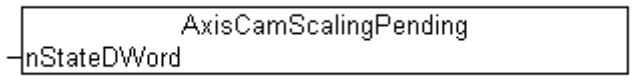
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    CamTableQueued : BOOL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9 ab Build 1032	PC (i386)	TcNC.Lib

4.1.25 AxisCamScalingPending



AxisCamScalingPending liefert das korrespondierende Signal aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

FUNCTION AxisCamScalingPending: BOOL

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

nStateDWord : Statuswort aus dem zyklischen Achsinterface [▶ 55] der NC zur SPS

Beispiel

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

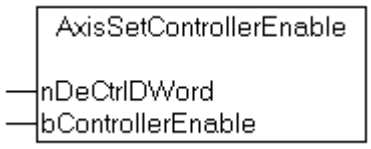
```
CamScalingPending := AxisCamScalingPending( NcToPlc1.nStateDWord );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9 ab Build 1025	PC (i386)	TcNC.Lib

4.2 Signale zur NC setzen

4.2.1 AxisSetControllerEnable



AxisSetControllerEnable setzt in *nDeCtrlDWord* das Reglerfreigabe-Signal gemäß dem Eingang *bControllerEnable* und liefert als Funktionsergebnis *nDeCtrlDWord* zurück.

FUNCTION AxisSetControllerEnable : DWORD

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

nDeCtrlDWord : Control-Word aus dem zyklischen Achsinterface [▶ 64] der SPS zur NC

bControllerEnable : Einzustellende Reglerfreigabe [FALSE,TRUE]

Funktionsergebnis : nDeCtrlDWord

Beispiel

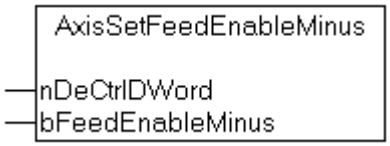
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.2 AxisSetFeedEnableMinus



AxisSetFeedEnableMinus setzt in *nDeCtrlDWord* das Vorschubfreigabe-Signal für die negative Vorschubrichtung gemäß dem Eingang *bFeedEnableMinus* und liefert als Funktionsergebnis *nDeCtrlDWord* zurück.

FUNCTION AxisSetFeedEnableMinus: DWORD

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

nDeCtrlDWord : Control-Word aus dem zyklischen Achsinterface [▶ 64] der SPS zur NC

bFeedEnableMinus : Einzustellende Vorschubfreigabe [FALSE,TRUE]

Funktionsergebnis : nDeCtrlDWord

Beispiel

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

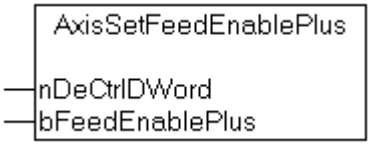
```
PlcToNc1.nDeCtrlDWord := AxisSetFeedEnableMinus(PlcToNc1.nDeCtrlDWord, TRUE );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.3 AxisSetFeedEnablePlus



AxisSetFeedEnablePlus setzt in *nDeCtrlDWord* das Vorschubfreigabe-Signal für die positive Vorschubrichtung gemäß dem Eingang *bFeedEnablePlus* und liefert als Funktionsergebnis *nDeCtrlDWord* zurück.

FUNCTION AxisSetFeedEnablePlus: DWORD

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

nDeCtrlDWord : Control-Word aus dem zyklischen Achsinterface [▶ 64] der SPS zur NC

bFeedEnablePlus : Einzustellende Vorschubfreigabe [FALSE,TRUE]

Funktionsergebnis : nDeCtrlDWord

Beispiel

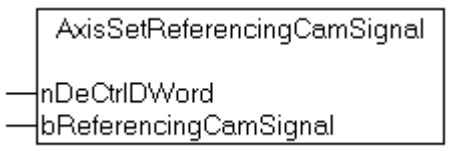
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.4 AxisSetReferencingCamSignal



AxisSetReferencingCamSignal setzt in *nDeCtrlDWord* das Referenznockensignal gemäß dem Eingang *bReferencingCamSignal* und liefert als Funktionsergebnis *nDeCtrlDWord* zurück.

FUNCTION AxisSetReferencingCamSignal: DWORD

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

nDeCtrlDWord : Control-Word aus dem zyklischen Achsinterface [▶ 64] der SPS zur NC

bReferencingCamSignal : Einzustellendes Referenznockensignal [FALSE,TRUE]

Funktionsergebnis : nDeCtrlDWord

Beispiel

```
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 );
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.5 AxisSetAcceptBlockedDriveSignal

AxisSetReferencingCamSignal setzt in *nDeCtrlDWord* ein Signal, das es der Steuerung erlaubt von einem Hardware-Endschalter herunterzufahren obwohl der Antrieb keine Betriebsbereitschaft meldet. Diese Funktion ist Hardware-abhängig.

FUNCTION AxisSetAcceptBlockedDriveSignal : DWORD

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

nDeCtrlDWord : Control-Word aus dem zyklischen Achsinterface [▶ 64] der SPS zur NC

bEnable : Einzustellendes Signal [FALSE,TRUE]

Funktionsergebnis : nDeCtrlDWord

Beispiel

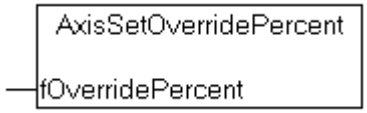
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
ND_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10 ab Build 1313	PC (i386)	TcNc.Lib

4.2.6 AxisSetOverridePercent



AxisSetOverridePercent liefert zu einem prozentualen Override-Wert in *fOverridePercent* den korrespondierenden NC-konformen Integerwert.

FUNCTION AxisSetOverridePercent: DWORD

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

fOverridePercent : Geschwindigkeits-Override in Prozent

Funktionsergebnis : NC-konformer Override-Wert

Beispiel

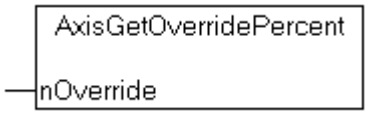
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.2.7 AxisGetOverridePercent



AxisGetOverridePercent ermittelt den Achs-Override aus dem zyklischen Achsinterface [▶ 64] der SPS zur NC und liefert ihn als Prozentwert im Funktionsergebnis zurück.

FUNCTION AxisGetOverridePercent : LREAL

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

nOverride : nOverride aus dem zyklischen Achsinterface [▶ 64] der SPS zur NC

Funktionsergebnis : Achs-Override in Prozent

Beispiel

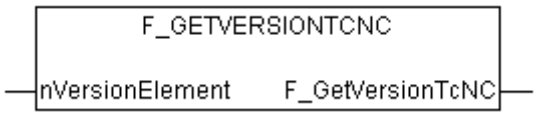
```
PROGRAM MAIN
VAR
    PlcToNc1 AT %QB1000 : PLCTONC_AXLESTRUCT;
    NcToPlc1 AT %IB1000 : NCTOPLC_AXLESTRUCT;
    fOverride : LREAL;
END_VAR
```

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.3 F_GetVersionTcNC



Mit dieser Funktion können Versionsinformationen der SPS-Bibliothek ausgelesen werden.

FUNCTION F_GetVersionTcNC : UINT

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

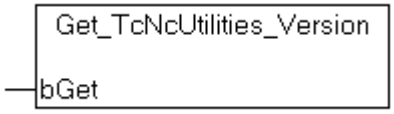
nVersionElement : Versionselement, das gelesen werden soll. Mögliche Parameter:

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

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.8.0	PC (i386)	TcNC.Lib

4.4 Get_TcNcUtilities_Version



Die Funktion ermittelt die Versionsnummer der SPS-Bibliothek. Der Funktionsaufruf liefert die Versionsnummer in einem String zurück.

FUNCTION Get_TcNcUtilities_Version: STRING(20)

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

bGet : Signal zum Ausführen des Kommandos

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	TcNcUtilities.Lib
TwinCAT v2.8.0	PC (i386)	TcNC.Lib: Diese Funktion befindet sich nur aus Kompatibilitätsgründen in der Bibliothek!. Bitte benutzen Sie die Funktion **F_GetVersionTcNC**() in neueren PLC-Projekten.

5 Datentypen

5.1 Zyklisches NC/SPS 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 "nCtrlL
oopIndex") *)
    nControlLoopIndex       : WORD; (* Axis control loop index (0, 1, 2, … when multiple control loo
ps 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
```

Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
1	UINT32	0-3	-	-	nStateDWord	StateDWord	Status-Doppelwort. Siehe auch detaillierte Beschreibung des StateDWord [▶ 71]
			0	0/1	Operational	Operational	Achse ist betriebsbereit
			1	0/1	Homed	Homed	Achse ist referenziert ("Achse geeicht")
			2	0/1	NotMoving	NotMoving	Achse ist im logischen Stillstand ("Achse fährt nicht")
			3	0/1	InPositionArea	InPositionArea	Achse ist im Positionsbereichsfenster (physikalische Rückmeldung)
			4	0/1	InTargetPosition	InTargetPosition	Achse ist in Zielposition (PEH) (physikalische Rückmeldung)
			5	0/1	Protected	Protected	Achse ist in geschützter Betriebsart (z.B. Slave-Achse)
			6	0/1	ErrorPropagationDelayed	ErrorPropagation Delayed	Achse signalisiert eine Fehlervorwarnung (ab TC 2.11)
			7	0/1	HasBeenStopped	HasBeenStopped	Achse ist gestoppt worden bzw. führt einen Stopp aus
			8	0/1	HasJob	HasJob	Achse hat Auftrag, führt Auftrag aus

Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
			9	0/1	PositiveDirection	PositiveDirection	Achse fährt logisch größer
			10	0/1	NegativeDirection	NegativeDirection	Achse fährt logisch kleiner
			11	0/1	HomingBusy	HomingBusy	Achse referenziert ("Achse wird geeicht")
			12	0/1	ConstantVelocity	ConstantVelocity	Achse hat V-Konst bzw. Drehzahl erreicht
			13	0/1	Compensating	Compensating	Streckenko mpensation passiv[0]/ aktiv[1] (s. "MC_MoveS uperImpose d")
			14	0/1	ExtSetPointGenEnab led	ExtSetPointGenE nabled	Externe Sollwertgen erierung freigegeben
			15	0/1			Betriebsart noch nicht ausgeführt (Busy). Noch nicht freigegeben!
			16	0/1	ExternalLatchValid	ExternalLatchVali d	Externer Latchwert bzw. Messtaster gültig geworden
			17	0/1	NewTargetPos	NewTargetPos	Achse hat neue Endposition bzw. neue Geschwindig keit erhalten
			18	0/1			Achse nicht in Zielposition bzw. kann/ wird diese nicht erreichen (z.B. Achs-Stopp). Noch nicht freigegeben!

Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
			19	0/1	ContinuousMotion	ContinuousMotion	Achse führt Endlos-Positioniera uftrag aus
			20	0/1	ControlLoopClosed	ControlLoopClose d	Achse betriebsbere it und Achsregelkr eis geschlossen (z.B. Lageregelun g)
			21	0/1	CamTableQueued	CamTableQueue d	Neue Tabelle steht für "Online Change" bereit und wartet auf Aktivierung
			22	0/1	CamDataQueued	CamDataQueued	Tabellendat en (MF) stehen für "Online Change" bereit und warten auf Aktivierung
			23	0/1	CamScalingPending	CamScalingPendi ng	Tabellenskal ierungen stehen für "Online Change" bereit und warten auf Aktivierung
			24	0/1	CmdBuffered	CmdBuffered	Nachfolgeko mmando liegt im Auftragspuff er bereit (s. Buffer Mode) (ab TwinCAT V2.10 Build 1311)
			25	0/1	PTPmode	PTPmode	Achse in PTP Betriebsart (kein Slave, keine NCI-Achse, keine FIFO-Achse) (ab TC 2.10 Build 1326)

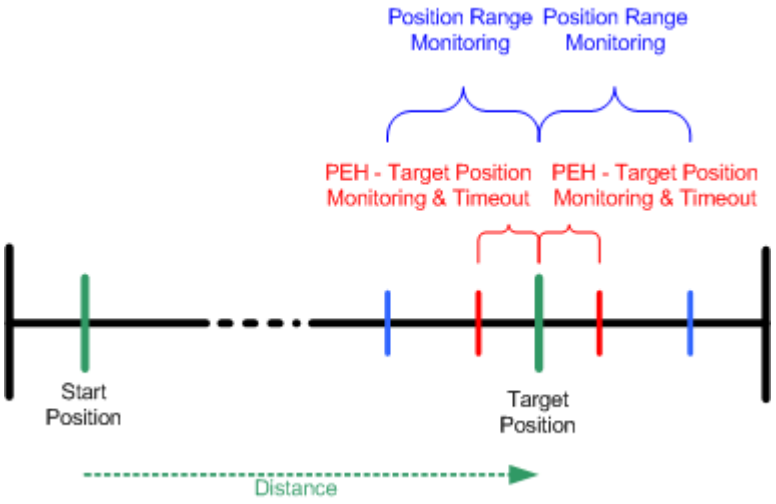
Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
			26	0/1	SoftLimitMinExceeded	SoftLimitMinExceeded	Software Endlage Minimum ist aktiv/belegt (ab TC 2.10 Build 1327)
			27	0/1	SoftLimitMaxExceeded	SoftLimitMaxExceeded	Software Endlage Maximum ist aktiv/belegt (ab TC 2.10 Build 1327)
			28	0/1	DriveDeviceError	DriveDeviceError	Antriebshardware hat einen Fehler (keine Warnung), Interpretation nur möglich wenn Antrieb im IO-Datenaustausch, z.B. EtherCAT "OP"-State (ab TC 2.10 Build 1326)
			29	0/1	MotionCommandsLocked	MotionCommandsLocked	Achse ist gesperrt für Bewegungskommandos (TcMc2)
			30	0/1	IoDataInvalid	IoDataInvalid	IO Daten ungültig (z.B. 'WcState' oder 'CdlState' des Feldbusses)
			31	0/1	Error	Error	Achse ist im Fehlerzustand
2	UINT32	4-7	-	≥0	nErrorCode	ErrorCode	Fehlercode Achse
3	UINT32	8-11	-	ENUM	nAxisState	AxisState	Bewegungszustand der Achse
4	UINT32	12-15	-	ENUM	nAxisModeCon	AxisModeConfirmation	Betriebsart der Achse (Rückmeldung der NC)

Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
5	UINT32	16-19	-	ENUM	nCalibrationState	HomingState	Referenziers tatus der Achse ("Eichstatus")
6	UINT32	20-23	-	ENUM	nCoupleState	CoupleState	Koppelstatu s der Achse
7	UINT32	24-27	-	≥0	nSvbEntries	SvbEntries	SVB-Einträge/ Aufträge
8	UINT32	28-31	-	≥0	nSafEntries	SafEntries	SAF-Einträge/ Aufträge (NC-Interpreter, Fifo-Gruppe)
9	UINT32	32-35	-	>0	nAxisId	AxisId	Achs-ID
10	UINT32	36-39	-	-	nOpModeDWord, bOpMode...	OpModeDWord, OpMode...	Achs-Betriebsart-Doppelwort
			0	0/1	...PosAreaMonitoring	...PosAreaMonitor ing	Positionsber eichsüberwa chung
			1	0/1	...TargetPosMonitori ng	...TargetPosMonit oring	Zielpositions fensterüber wachung
			2	0/1	...Loop	...Loop	Schleifenwe g
			3	0/1	...MotionMonitoring	...MotionMonitorin g	Physikalisch e Bewegungs überwachun g
			4	0/1	...PEHTimeMonitorin g	...PEHTimeMonito ring	PEH-Zeitüberwac hung
			5	0/1	...BacklashComp	...BacklashComp	Losekompen sation
			6	0/1	...DelayedErrorReact ion	...DelayedErrorRe action	Verzögerte Fehlerreakti on der NC
			7	0/1	...Modulo	...Modulo	Modulo-Achse (Modulo-Anzeige)
			8-15	0/1			RESERVE
			16	0/1	...PosLagMonitoring	...PosLagMonitori ng	Schleppabst andsüberwa chung Position
			17	0/1	...VeloLagMonitoring	...VeloLagMonitori ng	Schleppabst andsüberwa chung Geschwindig keit

Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
			18	0/1	...SoftLimitMinMonitoring	...SoftLimitMinMonitoring	Endlagenüberwachung Min.
			19	0/1	...SoftLimitMaxMonitoring	...SoftLimitMaxMonitoring	Endlagenüberwachung Max.
			20	0/1	...PosCorrection	...PosCorrection	Positionskorrektur ("Messsystemfehlerkompensation")
			21	0/1	...AllowSlaveCommands	...AllowSlaveCommands	Erlaube Bewegungskommandos an Slave-Achsen
			22	0/1			RESERVE
			23	0/1	ApplicationRequest	ApplicationRequest	Anforderungs-Bit für die Anwendersoftware (SPS Code) für z.B. einen "ApplicationHomingRequest" ab TwinCAT V2.11 Build 1546
			24-31	0/1			RESERVE
11	UINT16	40-41	-	≥0	nActiveControlLoopIndex	ActiveControlLoopIndex	Aktiver Achsregelkreis Index (identisch mit alter Variable "nCtrlLoopIndex") ab TwinCAT V2.10 Build 1311
12	UINT16	42-43	-	≥0	nControlLoopIndex	ControlLoopIndex	Achsregelkreis Index (0, 1, 2, … wenn mehrere Achsregelkreise verwendet werden) ab TwinCAT V2.10 Build 1311

Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
13	REAL64	44-51	-	±∞	fPosIst	ActPos	Istposition (verrechneter Absolutwert)
14	REAL64	52-59	-	>∞	fModuloPosIst	ModuloActPos	Modulo-Istposition (verrechneter Wert z.B. in Grad)
15	INT32	60-63	-	±∞	nModuloTurns	ModuloActTurns	Modulo-Ist-Umdrehungen
16	REAL64	64-71	-	±∞	fVeloIst	ActVelo	Istgeschwindigkeit (optional)
17	REAL64	72-79	-	±∞	fPosDiff	PosDiff	Schleppabstand (Position)
18	REAL64	80-87	-	±∞	fPosSoll	SetPos	Sollposition (verrechneter Absolutwert)
19	REAL64	88-95	-	±∞	fVeloSoll	SetVelo	Sollgeschwindigkeit
20	REAL64	96-103	-	±∞	fAccSoll	SetAcc	Sollbeschleunigung
21	REAL64	104-111	-	±∞	fPosTarget	TargetPos	Voraussichtliche Zielposition der Achse ab TwinCAT V2.10 Build 1311
22	REAL64	112-119	-	>∞	fModuloPosSoll	ModuloSetPos	Modulo Sollposition (verrechneter Wert z.B. in Grad) ab TwinCAT V2.10 Build 1311
23	INT32	120-123	-	±∞	nModuloTurnsSoll	ModuloSetTurns	Modulo Soll-Umdrehungen ab TwinCAT V2.10 Build 1311
24	UINT16	124-125	-	≥0	nCmdNo	CmdNo	Kommandonummer des aktiven Auftrags der Achse (s. Buffer Mode)

Nr.	Daten-typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
							ab TwinCAT V2.10 Build 1311
25	UINT16	126-127	-	≥0	nCmdState	CmdState	Kommando Statusinformation (s. Buffer Mode) ab TwinCAT V2.10 Build 1311



Beschreibung der Inhalte der einzelnen Felder:

Define	Referenzierstatus der Achse (*nCalibrationState* bzw. *HomingState*)
0	Referenziervorgang fertig (READY)
1	Endlosstart in Richtung der Referenziernocke (Hinweis: wenn Nocke zu Beginn belegt ist, dann wird direkt mit Referenzierstatus 3 begonnen)
2	Warten auf positive Flanke der Referenziernocke und Achsstopp einleiten
3	Warten bis Achse im Stillstand (Prüfung ob Nocke noch belegt ist) und dann Endlosstart vom Referenziernocken in Richtung Syncimpuls
4	Warten auf fallende Flanke der Referenziernocke
5	Latch aktivieren, warten bis Latch gültig geworden ist und dann den Achsstopp einleiten
6	Wenn Achse im Stillstand dann Istposition setzen (Istposition = Referenzposition + Bremsweg)

Siehe auch Anmerkungen zu MC_Home

Define	Koppelstatus der Achse (*nCoupleState* bzw. *CoupleState*)
0	Singleachse, die weder Master noch Slave ist (SINGLE)
1	Masterachse mit beliebiger Anzahl Slaves (MASTER)
2	Slaveachse, die Master eines anderen Slaves ist (MASTER-SLAVE)

Define	Koppelstatus der Achse (*nCoupleState* bzw. *CoupleState*)
3	Nur Slaveachse (SLAVE)

Define	Master: Bewegungszustand / Fahrphase der kontinuierlichen Masterachse (Servo) (*nAxisState* bzw. *AxisState*)
0	Sollwertgenerator nicht aktiv (INACTIVE)
1	Sollwertgenerator aktiv (RUNNING)
2	Geschwindigkeitsoverride ist Null (OVERRIDE_ZERO)
3	Konstante Geschwindigkeit (PHASE_VELOCONST)
4	Beschleunigungsphase (PHASE_ACCPOS)
5	Verzögerungsphase (PHASE_ACCNEG)

Define	Master: Bewegungszustand / Fahrphase der diskreten Masterachse (Eil/Schleich) (*nAxisState* bzw. *AxisState*)
0	Sollwertgenerator nicht aktiv
1	Fahrphase (Eilgang bzw. Schleichgang)
2	Umschaltverzögerung von Eil- auf Schleichgang
3	Schleichfahrt (innerhalb vom Schleichweg)
4	Bremszeit (beginnend ab Bremsweg vor dem Ziel)

Define	Slave: Bewegungszustand / Fahrphase der kontinuierlichen Slaveachse (Servo) (*nAxisState* bzw. *AxisState*) Anmerkung: vorerst nur für Slaves vom Typ *Synchronisierungsgenerator* !
0	Slavegenerator nicht aktiv (INACTIVE)
11	Slave befindet sich in einer Bewegungs-Vorphase (PREPHASE)
12	Slave ist am Aufsynchronisieren (SYNCHRONIZING)
13	Slave ist aufsynchronisiert und fährt synchron (SYNCHRON)

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
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
```

Für jede NC-Achse steht ein Datenblock von 128 Byte für den Datentransport NC -> SPS und ein ebenfalls 128 Byte großer Datenblock für den Datentransport SPS -> NC zur Verfügung. Der SPS-Programmierer muss für jede Richtung und jede Achse eine Variable erzeugen und sie im E/A-Bereich mit der AT-Anweisung auf den Input- und Outputbereich fixieren. Die Zuordnung zwischen den NC-Variablen und den SPS-Variablen geschieht mittels des TwinCAT System Managers.

Nr.	Daten typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
1	UINT32	0-3	-	0/1	nDeCtrlDWord	ControlDWord	Kontroll-Doppelwort:
			0	0/1	Enable	Enable	Reglerfreiga be
			1	0/1	FeedEnablePlus	FeedEnablePlus	Vorschubfrei gabe Plus
			2	0/1	FeedEnableMinus	FeedEnableMin us	Vorschubfrei gabe Minus
			3	0/1	-	-	RESERVE
			4	0/1	-	-	RESERVE
			5	0/1	HomingSensor	HomingSensor	Referenziern ocke bzw. Referenziers ensor
			6	0/1	-	-	RESERVE
			7	0/1	-	-	RESERVE
			8	0/1	AcceptBlockedDrive	AcceptBlockedD rive	Akzeptiere Sperre der Sollwertüber nahme des Drives (z.B. Hardware Endlagen) ab TwinCAT V2.10 Build 1311
			9	0/1	BlockedDriveDetected	BlockedDriveDe tected	Anwendersi gnal *Achse ist blockiert* (z.B. mechanisch er Festanschla g). Noch nicht freigegeben!
			10-29	0/1	-	-	RESERVE
			30	0/1	PlcDebugFlag	PlcDebugFlag	Debug-Funktion PLC. Nur für internen Gebrauch!
			31	0/1	NcDebugFlag	NcDebugFlag	Debug-Funktion NC. Nur für internen Gebrauch!

Nr.	Daten typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
2	UINT32	4-7	-	0...1000000	nOverride	Override	Geschwindigkeits-Override (0% bis 100%)
3	UINT32	8-11	-		nAxisModeReq	AxisModeRequest	Betriebsart der Achse. Nur für interne Verwendung vorgesehen !
4	UINT32	12-15	-		nAxisModeDWord	AxisModeDWord	Nur für interne Verwendung vorgesehen !
5	REAL64	16-23	-		fAxisModeLReal	AxisModeLReal	Nur für interne Verwendung vorgesehen !
6	REAL64	24-31	-		fActPosCorrection	PositionCorrection	Istpositionskorrekturwert
7	REAL64	32-39	-		fExtSetPos	ExtSetPos	Externe Sollposition
8	REAL64	40-47	-		fExtSetVelo	ExtSetVelo	Externe Sollgeschwindigkeit
9	REAL64	48-55	-		fExtSetAcc	ExtSetAcc	Externe Sollbeschleunigung
10	INT32	56-59	-		nExtSetDirection	ExtSetDirection	Externe Sollfahrrichtung [-1,0,1]
11	UINT32	60-63	-		nReserved1	-	RESERVE
12	REAL64	64-71	-		fExtCtrlOutput	ExtControllerOutput	Externe Regler Ausgabe. Noch nicht freigegeben!
13	REAL64	72-79	-	±∞	-	GearRatio1	Getriebefaktor (Koppelfaktor) 1
14	REAL64	80-87	-	±∞	-	GearRatio2	Getriebefaktor (Koppelfaktor) 2
15	REAL64	88-95	-	±∞	-	GearRatio3	Getriebefaktor (Koppelfaktor) 3

Nr.	Daten typ	Byte	Bit	Def. Bereich	Variable-Name	Variable-Name (ab 2.11 bzw. TcMc2)	Beschrei-bung
16	REAL64	96-103	-	±∞	-	GearRatio4	Getriebefaktor (Koppelfaktor) 4
17	-	104-127	-	-	nReserved	-	RESERVE

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
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
```

Dieser Datentyp wird in Verbindung mit der Funktion MC_NewPosAndVelo gebraucht, mit der die Zielposition und die Geschwindigkeit einer Achse während der Fahrt verändert werden kann. Dieser Typ definiert den Modus für diese Funktion:

CHANGE_POS: Die Zielposition der Fahrt wird instantan geändert, das heißt die Achse steuert sofort auf das neue Ziel zu.

CHANGE_VELO: Die Geschwindigkeit der Achse wird instantan geändert.

CHANGE_POSANDVELO: Beide Parameter, Zielposition und Geschwindigkeit werden instantan geändert.

CHANGE_POS_AT_SWITCHPOS: Die Zielposition der Fahrt wird verändert, sobald eine Umschaltposition erreicht wurde.

CHANGE_VELO_AT_SWITCHPOS: Die Geschwindigkeit der Fahrt wird verändert, sobald eine Umschaltposition erreicht wurde.

CHANGE_POSANDVELO_AT_SWITCHPOS: Beide Parameter, Zielposition und Geschwindigkeit werden geändert, sobald eine Umschaltposition erreicht wurde.

REACH_VELO_AT_POS : Die Geschwindigkeit wird geändert, so dass die neue Geschwindigkeit an der Umschaltposition erreicht wird. (Dieser Mode ist nur in Verbindung mit dem optimierten Sollwertgenerator einer Achse möglich, siehe globale Achsparameter ab TwinCAT 2.10 Build 1052).

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
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
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
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
```

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
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
```

Voraussetzungen

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 selec
ted working direction*)
   bModulo: BOOL := FALSE;
   eTableType: E_CompensationTableType:=TableType1DEquidistant;
END_STRUCT
END_TYPE
```

Voraussetzungen

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
```

Voraussetzungen

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
```

Voraussetzungen

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
```

Voraussetzungen

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 Anhang

6.1 Diskrete Eil-/Schleichachse (Two Speed)

Wertebereich Startgeschwindigkeit V	Interpretation der Startgeschwindigkeit bei Override 100% (geforderte Fahrgeschwindigkeit /Fahrstufe)
V > 50	Eilgang
0 < V ≤ 50	Schleichgang
V ≤ 0	FEHLER

Wertebereich Override X	Interpretation des Overridewertes (100% º 1.000.000)
X > 50% (500000)	Eilgang
0% < X ≤ 50% (500000)	Schleichgang
X = 0%	Stillstand (Toleranzfenster: <100 ≡ <0.01%)

Eine Overrideänderung (auch Override = 0) wird nur innerhalb der Hauptfahrphase wirksam. Wenn der Override innerhalb einer der Bremsphasen auf 0 gesetzt wird, wird die eingeleitete Bremsphase unbeeinflusst beendet.

6.2 Drive-Interface für Eil/Schleichachsen NC->IO (12 Byte)

Nr.	Datentyp	Byte	Bit	Def. Bereich	Variable-Name	Beschreibung
1	UINT32	0-3	-	-	nOutData1	Drive-Output Ausgabedaten 1 (NC->IO)
2	UINT32	4-7	-	-	nOutData2	Drive-Output Ausgabedaten 2 (NC->IO)
3	UINT8	8	-	-	nControlByte	Control-Byte
			0	0/1	bMinusHigh	Richtung: Negativ Geschwind.: Schnell
			1	0/1	bMinusLow	Richtung: Negativ Geschwind.: Langsam
			2	0/1	bPlusLow	Richtung: Positiv Geschwind.: Langsam
			3	0/1	bPlusHigh	Richtung: Positiv Geschwind.: Schnell
			4	0/1	-	RESERVE
			5	0/1	-	RESERVE
			6	0/1	bBreakInv	Inverses Bremsbit (0 ≡ AKTIV, 1 ≡ PASSIV)
			7	0/1	bBreak	Bremsbit (0 ≡ PASSIV, 1 ≡ AKTIV)
4	UINT8	9	-	-	nExtControlByte	Extended Control Byte
			0	0/1	bDirectionMinus	Richtung: Negativ
			1	0/1	bDirectionPlus	Richtung: Positiv
			2	0/1	bVeloLow	Geschwindigkeit: Langsam

Nr.	Datentyp	Byte	Bit	Def. Bereich	Variable-Name	Beschreibung
			3	0/1	bVeloHigh	Geschwindigkeit: Schnell
			4	0/1	-	RESERVE
			5	0/1	-	RESERVE
			6	0/1	bBreakInv	Inverses Bremsbit (0 ≡ AKTIV, 1 ≡ PASSIV)
			7	0/1	bBreak	Bremsbit (0 ≡ PASSIV, 1 ≡ AKTIV)
5	UINT16	10-11	-	-	nReserved	Reserve-Bytes

Ein Achsstart wird nur innerhalb einer Entfernung zum Zielpunkt ausgeführt, die echt größer als der parametrierte Bremsweg ist.

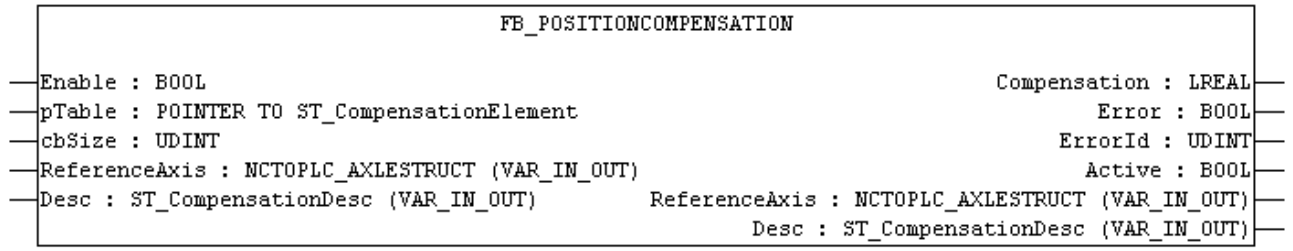
6.3 "Low Cost" Schrittmotorachse mit digitaler Ansteuerung (Stepper)

Drive-Interface für "Low Cost" Schrittmotorachse NC->IO (12 Byte)

Nr.	Datentyp	Byte	Bit	Def. Bereich	Variable-Name	Beschreibung
1	INT32	0-3	-	-	nOutData1	Drive-Output Ausgabedaten 1 (NC->IO)
2	INT32	4-7	-	-	nOutData2	Drive-Output Ausgabedaten 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 Invers
3.2	...	8	2	0/1	bPhaseB	Phase B
3.3	...	8	3	0/1	bPhaseBInv	Phase B Invers
3.4	...	8	4	0/1	-	RESERVE
3.5	...	8	5	0/1	-	RESERVE
3.6	...	8	6	0/1	bBreakInv	Inverses Bremsbit (0 ≡ AKTIV, 1 ≡ PASSIV)
3.7	...	8	7		bBreak	Bremsbit (1 ≡ AKTIV, 0 ≡ PASSIV)
4	UINT8	9	-	-	nExtControlByte	Extended Control Byte
4.0	...	9	0	0/1	bFrequency	Frequenz (Rechtecksignal)
4.1	...	9	1	0/1	bDirectionPlus	Richtung: Positiv
4.2	...	9	2	0/1	-	RESERVE
4.3	...	9	3	0/1	-	RESERVE
4.4	...	9	4	0/1	-	RESERVE
4.5	...	9	5	0/1	-	RESERVE
4.6	...	9	6	0/1	-	RESERVE
4.7	...	9	7	0/1	-	RESERVE
5	UINT16	10-11	-	-	nReserved	Reserve-Bytes

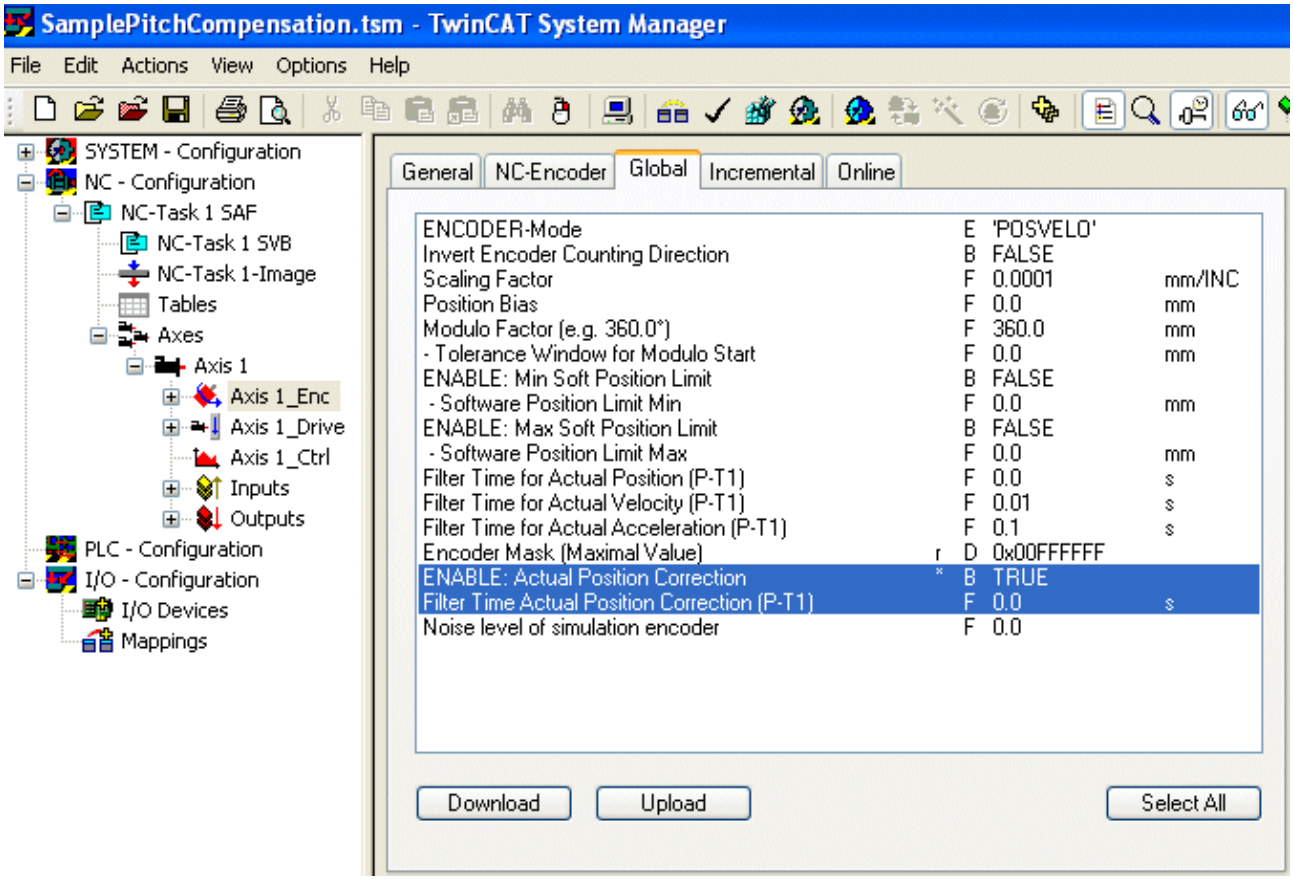
6.4 Beispiel Steigungskompensation

[This is preliminary documentation and subject to change.]



Dieses Beispiel zeigt, wie die Funktionsbausteine FB_PositionCompensation [▶ 28] und FB_WritePositionCorrection [▶ 27] zur Steigungskompensation einer Spindel verwendet werden.

Abhängig von der erforderlichen Genauigkeit wird empfohlen, die SPS Task mit der gleichen Zykluszeit wie die NC_SAF Task auszuführen. Ebenfalls ist es notwendig, die gegenwärtige Positionskorrektur der Achse zu aktivieren (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
```

Die Kompensationstabelle und -beschreibung ist in diesem Fall als konstant definiert. fPos hält die unkorrigierte absolute Position und für jede Position gibt es einen Kompensationswert fCompensation.

In annähernd allen Applikationen ist die Kompensation nur brauchbar wenn die Achse referenziert wurde (homing). FB_PositionCompensation sollte nur aktiviert werden, wenn diese Voraussetzung gegeben ist.

```
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=> );
```

Voraussetzungen

Entwicklungumgebung	Zielplattform	Einzubindende SPS Bibliotheken
from TwinCAT v2.10 Build 1314	PC (i386)	TcNc.Lib

Mehr Informationen:
www.beckhoff.de/tx1200

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.de
www.beckhoff.de