

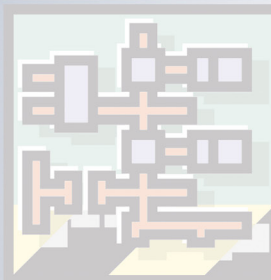
Handbuch | DE

# TS5810

TwinCAT 2 | PLC Hydraulic Positioning



Supplement | Motion





# Inhaltsverzeichnis

<b>1</b>	<b>Vorwort.....</b>	<b>7</b>
1.1	Hinweise zur Dokumentation .....	7
1.2	Zu Ihrer Sicherheit.....	8
1.3	Hinweise zur Informationssicherheit .....	9
<b>2</b>	<b>Einführung in die Hydraulik .....</b>	<b>10</b>
<b>3</b>	<b>Genereller Aufbau .....</b>	<b>16</b>
3.1	Gliederung der Dokumentation .....	16
3.2	Funktionen, Bausteine und Typen (ab V3.0).....	17
3.3	Die Hydraulikbibliothek.....	24
<b>4</b>	<b>PLCopen Motion Control .....</b>	<b>27</b>
4.1	Administrative.....	27
4.1.1	MC_Power_BkPlcMc (ab V3.0).....	27
4.1.2	MC_ReadActualPosition_BkPlcMc (ab V3.0) .....	29
4.1.3	MC_ReadActualTorque_BkPlcMc (ab V3.0).....	30
4.1.4	MC_ReadActualVelocity_BkPlcMc (ab V3.0).....	31
4.1.5	MC_ReadAxisError_BkPlcMc (ab V3.0) .....	32
4.1.6	MC_ReadBoolParameter_BkPlcMc (ab V3.0) .....	33
4.1.7	MC_ReadDigitalOutput_BkPlcMc (ab V3.0) .....	34
4.1.8	MC_ReadParameter_BkPlcMc (ab V3.0) .....	35
4.1.9	MC_ReadStatus_BkPlcMc (ab V3.0).....	37
4.1.10	MC_Reset_BkPlcMc (ab V3.0) .....	39
4.1.11	MC_ResetAndStop_BkPlcMc (ab V3.0).....	40
4.1.12	MC_SetOverride_BkPlcMc (ab V3.0).....	41
4.1.13	MC_SetPosition_BkPlcMc (ab V3.0).....	43
4.1.14	MC_SetReferenceFlag_BkPlcMc (ab V3.0).....	44
4.1.15	MC_WriteBoolParameter_BkPlcMc (ab V3.0) .....	45
4.1.16	MC_WriteDigitalOutput_BkPlcMc (ab V3.0).....	46
4.1.17	MC_WriteParameter_BkPlcMc (ab V3.0).....	47
4.2	Motion .....	48
4.2.1	MC_CamIn_BkPlcMc (ab V3.0) .....	48
4.2.2	MC_CamOut_BkPlcMc (ab V3.0) .....	50
4.2.3	MC_CamTableSelect_BkPlcMc (ab V3.0) .....	52
4.2.4	MC_DigitalCamSwitch_BkPlcMc (ab V3.0).....	53
4.2.5	MC_EmergencyStop_BkPlcMc (ab V3.0.5) .....	56
4.2.6	MC_FlyingGear_BkPlcMc.....	58
4.2.7	MC_GearIn_BkPlcMc (ab V3.0).....	62
4.2.8	MC_GearInPos_BkPlcMc (ab V3.0.33).....	64
4.2.9	MC_GearOut_BkPlcMc (ab V3.0).....	66
4.2.10	MC_Halt_BkPlcMc (ab V3.0) .....	67
4.2.11	MC_Home_BkPlcMc (ab V3.0) .....	68
4.2.12	MC_ImmediateStop_BkPlcMc (ab V3.0.5).....	71
4.2.13	MC_MoveAbsolute_BkPlcMc (ab V3.0).....	72
4.2.14	MC_MoveJoySticked_BkPlcMc (ab V3.0).....	74

4.2.15	MC_MoveRelative_BkPlcMc (ab V3.0).....	76
4.2.16	MC_MoveVelocity_BkPlcMc (ab V3.0) .....	78
4.2.17	MC_RampedStop_BkPlcMc.....	79
4.2.18	MC_Stop_BkPlcMc (ab V3.0) .....	81
4.2.19	MC_MoveJog_BkPlcMc.....	82
4.3	Datentypen.....	84
4.3.1	Axis_Ref_BkPlcMc (ab V3.0).....	84
4.3.2	E_TcPlcBufferedCmdType_BkPlcMc.....	86
4.3.3	E_TcMcCurrentStep (ab V3.0).....	88
4.3.4	E_TcMcDriveType (ab V3.0).....	90
4.3.5	E_TcMcEncoderType (ab V3.0).....	93
4.3.6	E_TcMCFbState (ab V3.0).....	96
4.3.7	E_TcMcHomingType (ab V3.0).....	96
4.3.8	E_TcMCPParameter (ab V3.0).....	97
4.3.9	E_TcMcProfileType (ab V3.0).....	110
4.3.10	E_TcMcPressureReadingMode (ab V3.0) .....	111
4.3.11	E_TcMcValveType .....	111
4.3.12	MC_BufferMode_BkPlcMc (ab V3.0) .....	112
4.3.13	MC_CAM_ID_BkPlcMc (ab V3.0) .....	113
4.3.14	MC_CAM_REF_BkPlcMc (ab V3.0).....	113
4.3.15	MC_CAMSWITCH_REF_BkPlcMc (ab V3.0) .....	114
4.3.16	MC_Direction_BkPlcMc (ab V3.0).....	115
4.3.17	MC_HomingMode_BkPlcMc (ab V3.0) .....	115
4.3.18	MC_StartMode_BkPlcMc (ab V3.0) .....	116
4.3.19	MC_TRACK_REF_BkPlcMc (ab V3.0) .....	116
4.3.20	OUTPUT_REF_BkPlcMc (ab V3.0) .....	117
4.3.21	ST_FunctionGeneratorFD_BkPlcMc (ab V3.0.31).....	117
4.3.22	ST_FunctionGeneratorTB_BkPlcMc (ab V3.0.31) .....	118
4.3.23	ST_TcMcAutoIdent (ab V3.0.4).....	118
4.3.24	ST_TcHydAxParam (ab V3.0).....	120
4.3.25	ST_TcHydAxRtData (ab V3.0).....	131
4.3.26	ST_TcMcAuxDataLabels (ab V3.0).....	137
4.3.27	ST_TcPlcDeviceInput (ab V3.0).....	138
4.3.28	ST_TcPlcDeviceOutput (ab V3.0).....	140
4.3.29	ST_TcPlcMcLogBuffer (ab V3.0) .....	142
4.3.30	ST_TcPlcMcLogEntry (ab V3.0).....	142
4.3.31	ST_TcPlcRegDataItem (ab V3.0.7).....	143
4.3.32	ST_TcPlcRegDataTable (ab V3.0.7).....	144
4.3.33	ST_TcHybridAxParam (ab V3.0.44).....	144
4.3.34	ST_TcHybridAxRtData (ab V3.0.44).....	146
4.3.35	ST_TcPlcInputAnalog (ab V3.0.44).....	148
4.3.36	ST_TcCtrlParam.....	148
4.3.37	MC_Ref_Signal_Ref_BkPlcMc .....	149
4.3.38	E_TcMcJogMode .....	150
4.4	System .....	151
4.4.1	Controller.....	151

4.4.2	Drive.....	172
4.4.3	Encoder.....	183
4.4.4	FunctionGenerator .....	211
4.4.5	TableFunctions.....	213
4.4.6	Generatoren .....	221
4.4.7	Runtime.....	228
4.4.8	Message Logging.....	240
4.4.9	Utilitys.....	245
4.5	Parameter .....	263
4.5.1	MC_AxAdsCommServer_BkPlcMc (ab V3.0) .....	263
4.5.2	MC_Communications_BkPlcMc.....	264
4.5.3	MC_AxAdsPtrArrCommServer_BkPlcMc.....	265
4.5.4	MC_AxAdsReadDecoder_BkPlcMc (ab V3.0) .....	266
4.5.5	MC_AxAdsWriteDecoder_BkPlcMc (ab V3.0) .....	268
4.5.6	MC_AiParamAuxLabelsLoad_BkPlcMc (ab V3.0) .....	269
4.5.7	MC_AiParamLoad_BkPlcMc (ab V3.0).....	270
4.5.8	MC_AiParamSave_BkPlcMc (ab V3.0).....	271
4.5.9	MC_AxUtiReadCoeDriveTerm_BkPlcMc (ab V3.0) .....	272
4.5.10	MC_AxUtiReadCoeEncTerm_BkPlcMc (ab V3.0) .....	274
4.5.11	MC_AxUtiReadRegDriveTerm_BkPlcMc (ab V3.0) .....	276
4.5.12	MC_AxUtiReadRegEncTerm_BkPlcMc (ab V3.0) .....	277
4.5.13	MC_AxUtiUpdateRegDriveTerm_BkPlcMc (ab V3.0.7) .....	278
4.5.14	MC_AxUtiUpdateRegEncTerm_BkPlcMc (ab V3.0.7) .....	280
4.5.15	MC_AxUtiWriteCoeDriveTerm_BkPlcMc (ab V3.0) .....	282
4.5.16	MC_AxUtiWriteCoeEncTerm_BkPlcMc (ab V3.0).....	283
4.5.17	MC_AxUtiWriteRegDriveTerm_BkPlcMc (ab V3.0) .....	285
4.5.18	MC_AxUtiWriteRegEncTerm_BkPlcMc (ab V3.0).....	286
4.6	Part 5 Homing .....	288
4.6.1	FinalizingFunctions .....	288
4.6.2	StepFunctions .....	290
<b>5</b>	<b>Knowledge Base .....</b>	<b>302</b>
5.1	FAQs (ab V3.0) .....	303
5.2	Globale Konstanten (ab V3.0).....	319
5.3	Ventil .....	328
5.4	Elektrisch-hydraulische Hybrid-Achsen.....	330
5.5	Bau einer Achse .....	343
5.5.1	FB_Power .....	349
5.6	Der PlcMcManager .....	349
5.7	Programm-Beispiele (ab V3.0).....	353
5.8	Inbetriebnahme .....	358
5.8.1	Grundeinstellungen .....	358
5.8.2	Temporärer Nullpunktgleich .....	360
5.8.3	Bewegungsrichtungen.....	360
5.8.4	Wegmesssystem.....	360
5.8.5	Kennlinienvermessung.....	361
5.8.6	Überdeckung.....	362

---

5.8.7	Referenzgeschwindigkeit/Geschwindigkeitsverhältnis .....	362
5.8.8	Referenzierungen.....	364
5.8.9	Dynamik/Zielanfahrt .....	364
<b>6</b>	<b>Support und Service .....</b>	<b>367</b>

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

### Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.

Weitergabe sowie Vervielfältigung dieses Dokuments, Verwertung und Mitteilung seines Inhalts sind verboten, soweit nicht ausdrücklich gestattet.

Zu widerhandlungen verpflichten zu Schadenersatz. Alle Rechte für den Fall der Patent-, Gebrauchsmuster- oder Geschmacksmustereintragung vorbehalten.

## 1.2 Zu Ihrer Sicherheit

### Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.  
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

### 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.

### Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

### Warnungen vor Personenschäden

#### **GEFAHR**

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

#### **WARNUNG**

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

#### **VORSICHT**

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

### Warnung vor Umwelt- oder Sachschäden

#### **HINWEIS**

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

### Information zum Umgang mit dem Produkt



Diese Information beinhaltet z. B.:  
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.



## 1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem <https://www.beckhoff.de/secguide>.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter <https://www.beckhoff.de/secinfo>.

## 2 Einführung in die Hydraulik

### Hydraulik vs. Elektromechanik: Ein Technologievergleich

Hydraulische Antriebe unterscheiden sich von elektrischen Antrieben durch einen grundlegend anderen Aufbau, der zu einem nur bedingt vergleichbaren Verhalten führt. Sowohl dieses spezielle Verhalten als auch das deutlich andere Einsatzfeld erfordern angepasste Steuerungs- und Kontrollmechanismen. Im Nachfolgenden werden diese Unterschiede im Überblick dargestellt.

Die von TwinCAT NC/NCI/CNC kontrollierten elektromechanischen Achsen bestehen typischerweise aus einem AX-Servoverstärker und einem AM-Synchronmotor mit integriertem Wegmesssystem. Die Unterschiede betreffen hier vor allem die Auslegung, denn auch Linearmotoren oder Asynchronmotoren lassen sich auf dieses Grundprinzip zurückführen. Der Servoverstärker erzeugt durch die von ihm kontrollierten Ströme ein rotierendes oder wanderndes Magnetfeld, dem der bewegliche Teil des Motors folgt. Dabei wird dieses Magnetfeld in seiner Stärke, Drehzahl und Winkel- bzw. Drehzahl-Differenz zum Rotor so geführt, dass die gewünschte Bewegung zustande kommt. Bei angemessener Auslegung entsteht eine Anordnung, die sich gut modellieren lässt. Da der grundsätzliche Aufbau konstant ist, gilt dies weitgehend auch für das Modell.

Hydraulische Achsen zeigen in ihrem Aufbau eine wesentlich stärkere Streuung. So stehen als Aktuatoren neben den diversen Varianten von linearen Zylindern (Plunger, Gleichlauf-, Differenzial-Zylinder, flächenumschaltbare Zylinder etc.) auch mehrere rotatorische Antriebe (Schwenkzylinder, Drehzylinder, diverse Bauarten von Hydromotoren) zur Verfügung. Die Geschwindigkeit kann durch Stetig-Ventile oder primär bzw. sekundär gesteuerte Pumpen definiert werden. Hinzu kommen diverse Hydraulik-Schaltungen, in denen weitere die Öl-Menge oder den Druck beeinflussende Komponenten ergänzt werden. Fast jede hat ein nichtlineares oder situationsabhängiges Verhalten.

Letztlich führen diese Unterschiede dazu, dass Anwendungen, die bereits durch eine genau vorgegebene und dann exakte ausgeführte Bewegung erfüllbar sind, heute weitgehend elektromechanisch realisiert werden. Die komplexeren, weniger standardisierten und schwerer handhabbaren hydraulischen Achsen werden für Aufgaben gewählt, in denen ihre besonderen Stärken genutzt werden sollen. So sind sie unter anderem hervorragend dazu geeignet, große Kräfte und Energien auch über längere Zeiten oder in begrenztem Bauraum einzusetzen. Dabei werden oft für elektromechanische Antriebe untypische Verhalten wie begrenzte oder ablösende Druck- oder Kraftregelung kommandiert. Als Beispiele sollen hier die Kunststoffindustrie und die Metallumformung genannt werden.

### Elektrisch-hydraulische Hybrid-Achsen

Sowohl elektromechanische Servo-Achsen als auch Hydraulik-Achsen bieten spezifische Vorteile. Durch die Kombination der Technologien entsteht ein hybrides System, das eine neue Mischung von positiven und negativen Eigenschaften beider Welten bietet. Auch wenn auf diesem Wege nicht unter Vermeidung aller Nachteile die Nutzung aller Vorteile möglich ist, kann durch geeignete Varianten ein deutlicher Vorteil erreicht werden. Im Nachfolgenden werden die Konzepte im Überblick dargestellt.

Bei proportionalventil-gesteuerten Hydraulik-Achsen stellt der im Vergleich mit Servo-Achsen niedrige Wirkungsgrad einen deutlichen Nachteil dar. Er wird durch das Prinzip der Drosselsteuerung begrenzt. Elektrische Antriebe werden seit Jahrzehnten nach dem PWM-Prinzip kontrolliert. Dies ist bei hydraulischen Achsen aus technischen Gründen (keine Schaltventile mit hohem Durchfluss und niedriger Schaltzeit  $\ll 1\text{ms}$ ) nicht vergleichbar möglich. Hybrid-Achsen steuern den Ölfluss nicht mit einer variablen Drossel, sondern in dem sie mit einem Servo-Antrieb die Drehzahl und möglicherweise die Drehrichtung einer Konstantstrom-Pumpe verändern. Theoretisch gibt es also keinen Druckabfall zwischen Pumpe und Zylinder. Die Pumpe kann hier als kraftschlüssiges, aber nicht formschlüssiges Getriebe verstanden werden, während der Zylinder die Rolle einer Spindel übernimmt.

Wenn die Möglichkeit geschaffen wird, durch situationsabhängige Umschaltung der Ölwege die wirksamen Zylinderflächen oder die pro Umdrehung geförderte Ölmenge zu verändern, steht eine wählbare Vorschubkonstante zur Verfügung. Es entsteht eine echte Getriebeschaltung, die für eine elektromechanische Achse nicht verfügbar ist. In Anwendungen, die abwechselnd hohe Geschwindigkeit und hohe Kraft benötigen, kann dies zu einer erheblichen Einsparung führen.

Durch Schaltventile kann eine einmal aufgebaute Kraft hydraulisch geklemmt und der elektrische Antrieb entlastet werden. Auf diesem Weg kann die Momentenreduktion einer elektromechanischen Achse vermieden werden.

Bis zu einer nicht unerheblichen Leistung können alle Komponenten der Hybrid-Achse als abgeschlossener Modul montiert werden. In diesem Fall sind alle hydraulischen Verbindungen im Inneren gekapselt und von außen nur elektrische Anschlüsse vorhanden. Die Achse ist montierbar und auch austauschbar wie eine elektromechanische Achse. Höhere Leistungen erfordern einen klassischen diskreten Aufbau. Allerdings sollte hier berücksichtigt werden, dass eine vergleichbare elektromechanische Achse ebenfalls alles andere als kompakt oder leicht ist.

Näheres zur Identifikation des verwendeten Aufbau-Konzepts und der Inbetriebnahme wird in der Knowledge Base (in Vorbereitung) dargestellt.

## Unterschiede im Überblick

Die oben beschriebenen Unterschiede im Aufbau erzeugen erhebliche Auswirkungen auf das Betriebsverhalten von hydraulischen und elektrischen Antrieben. Im Nachfolgenden werden diese Auswirkungen im Überblick dargestellt.

Typische Eigenfrequenzen von elektromechanischen Achsen liegen im Bereich  $>80$  Hz. Für hydraulische Achsen sind Werte bis unter 20 Hz nicht unüblich. In beiden Technologien sind Achsen mit  $>200$  Hz realisierbar, werden aber aus technischen und/oder kalkulatorischen Gründen nur bei Bedarf eingesetzt. Die Eigenfrequenz hat einen direkten Einfluss auf die Regelbarkeit, denn sie limitiert den nutzbaren  $k_P$  des Lagereglers. Standard-NCs setzen die Regelbarkeit der elektromechanischen Achsen voraus.

- Hydraulische Achsen setzen bevorzugt Differential-Zylinder mit nur einer Kolbenstange ein. Dadurch wird die Vorschubkonstante (hier als Weg pro Ölmenge definiert) richtungsabhängig. Ein solches Verhalten wird von Standard-NCs nicht berücksichtigt, weil es diesen Effekt bei elektromechanischen Achsen nicht gibt.
- Die asymmetrischen Arbeitsflächen eines Differential-Zylinders erfordern für einen Stillstand im Kraftgleichgewicht eine asymmetrische Druckverteilung auf den Flächen. Bei einem Achsstart in Gegenrichtung muss eine andere Druckverteilung hergestellt werden. Dazu muss eine Ölmenge durch die zunächst nur wenig geöffneten Ventile transportiert werden, ohne dass dabei eine Bewegung zustande kommt. Dies führt zu einem verzögerten Anfahren. Ein vergleichbares, aber deutlich heftigeres Phänomen tritt auf, wenn die Achse vorher eine Presskraft aufgebaut hat. Ein solches Verhalten wird von Standard-NCs nicht berücksichtigt, weil es diesen Effekt bei elektromechanischen Achsen nicht gibt.
- Hydraulische Aktoren sind auf Dichtungen angewiesen, um ihre Arbeitsräume voneinander und von der Umgebung zu trennen. Diese Dichtungen liegen mit z. T. langen umlaufenden Kanten an metallischen Flächen an und müssen darauf gleiten. Vor allem der Übergang vom Stillstand in die Bewegung ist von ausgeprägten Haft/Gleit-Reibungswechseln begleitet. Die vergleichbaren Effekte bei elektromechanischen Achsen sind um mehrere Größenordnungen kleiner und werden üblicherweise nicht berücksichtigt. Bei hydraulische Achsen bestimmen sie das Verhalten beim Anfahren, bei der Zielannäherung und beim Fahren mit niedrigen Geschwindigkeiten maßgeblich.
- Hydraulische Achsen benutzen stetig verstellbare Ventile oder Pumpen als Stellglied. Diese Komponenten sind immer mehr oder weniger nichtlinear. Die vom Regler zu berücksichtigende Streckenverstärkung und die von der Vorsteuerung zu verwendende Vorschubkonstante werden arbeitspunktabhängig. Kompromisse bei der Bewegungsführung können durch eine Linearisierung verringert, aber nicht vollständig vermieden werden. Ein solches Verhalten wird von Standard-NCs nicht berücksichtigt, weil es diesen Effekt bei elektromechanischen Achsen nicht gibt.
- Eine Totzone um den Nullpunkt von mehreren 10 % der Vollaussteuerung ist bei Ventilen nicht unüblich. Auch mit einer Linearisierung ist eine Lageregelung im Stillstand dann nur eingeschränkt möglich. Ein solches Verhalten wird von Standard-NCs nicht berücksichtigt, weil es diesen Effekt bei elektromechanischen Achsen nicht gibt.
- Der an das Ventil gesendete Ausgabewert definiert die Schieberposition und somit über eine nichtlineare mechanische Funktion die Öffnungen für den Ölstrom. Allerdings hat der Druckabfall über der Öffnung einen starken Einfluss auf die tatsächliche Ölmenge und somit auf die Zylindergeschwindigkeit. Schwankungen des Versorgungsdrucks oder des Zylinderdrucks (als Abbildung der Prozesskraft) haben einen starken Einfluss auf die Achsgeschwindigkeit.
- Die Verwendung eines I-Anteils im Regler ist nicht ohne weiteres möglich. Im Zusammenspiel mit den beschriebenen Haft/Gleit-Reibungswechseln kommt es leicht zu sehr typischen niederfrequenten Schwingungen, die kaum beherrschbar sind. Dabei pendelt der Zylinder periodisch um vom Arbeitszyklus bestimmte Positionen und beschädigt mittelfristig Dichtungen und Oberflächen.

Es ist nicht ausgeschlossen, dass sich eine hydraulische Achse mit einer Standard-NC betreiben lässt. Dies ist umso leichter möglich, je höherwertig die Komponenten ausgewählt und ausgelegt sind. Allerdings bieten dann die Erwartungen an das Verhalten wenig Spielraum für Kompromisse. Spätestens bei üblichen Auslegungen erfordern hydraulische Achsen angepasste Lösungen, die bei Beckhoff Automation in der Hydraulik-Library bereitgestellt werden.

### Motion Control einmal anders

Die zentrale Funktion einer Motion-Control-Lösung ist der Sollwertgenerator. Er berechnet oder beschließt momentane Sollwerte für Position, Geschwindigkeit, Beschleunigung und möglicherweise Ruck. Hier ist die zeitgesteuerte Arbeitsweise der NC gut bekannt. Allerdings gibt es hierzu eine oft übersehene Alternative, die gerade bei hydraulischen Achsen interessant ist. Ihre Herleitung und die Unterschiede sollen nachfolgend dargestellt werden.

Ein Sollwertgenerator kann entweder abhängig oder unabhängig von den Größen einer anderen Achse arbeiten. Das erstere ist gegeben, wenn die Werte bei einer Kurvenscheibenkopplung durch eine Tabelle oder bei einer Getriebekopplung durch eine Rechenformel von den Werten einer anderen Achse abgeleitet werden. Hierbei ist ein während der Bewegung aktiver Lageregler gefordert. Sowohl die Hydraulik-Library als auch vor allem die NC bieten hier diverse Möglichkeiten.

Werden die Werte unabhängig von anderen Achsen berechnet ist zwischen zeitgeführter und weggeführter Generierung zu unterscheiden. TwinCAT NC/NCI/CNC arbeitet wie praktisch alle aktuellen MC Systeme zeitgeführt. Die Kerntechnologie der Hydraulik-Library ist weggesteuert, aber auch hier ist ein zeitgesteuerter Betrieb möglich. Nachfolgend werden die Unterschiede dargestellt.

Eine zeitgesteuerte Motion Control Lösung verwendet Gleichungen, in denen die Zeit das Bewegungsprofil ablaufen lässt. Nachstehend wird dies für eine beschleunigte Bewegung dargestellt:

$$V := A * t$$

$$P := \frac{1}{2} * A * t^2$$

Wird die erste Gleichung quadriert und dann beide Gleichungen nach  $t^2$  aufgelöst und gleichgesetzt entsteht folgende Gleichung:

$$V := \text{SQRT}(2 * A * P)$$

Wird für P der Absolutwert der Reststrecke s zu einer Zielposition verwendet und das Vorzeichen wieder hergestellt, ergibt sich eine sich anpassende Bremsrampe.

$$V := \pm \text{SQRT}(2 * A * \text{ABS}(s))$$

Zu beachten ist, dass die Zeit als steuernde Größe durch den Weg ersetzt wurde. Wird diese Bremsrampe mit einer Rampe für die Beschleunigungsphase und einer Konstantphase kombiniert, ergibt sich die Grundlage für eine einfache aber besonders robuste Motion-Control-Lösung, die sich durch folgende Eigenschaften unterscheidet:

- Verzögerte Achsreaktionen beim Start einer Bewegung werden ignoriert. Das Ventil wird nicht durch einen Lageregler zunächst wirkungslos überbetont geöffnet, um dann beim Losbrechen des Zylinders möglicherweise bis zum Stillstand abzuregeln.
- Auch während der aktiven Bewegung findet keine Lageregelung statt. Fährt die Achse nicht mit genau der richtigen oder einer variierenden Geschwindigkeit wird dies automatisch durch ein vorzeitiges oder verzögertes Einleiten der Bremsphase kompensiert.
- Durch vom Prozess aufgebaute Gegenkräfte wird die Achse verlangsamt. Dies führt jedoch auch ohne eine Reaktion der Steuerung unvermeidlich zu einem Anstieg des Drucks, möglicherweise bis zum Versorgungsdruck und somit bis zur maximal verfügbaren Kraft. Sollte diese nicht für eine weitere Bewegung ausreichen, würde dies auch durch einen Regler nicht bewirkt. Auch ohne Lageregelung besteht nicht das Risiko, dass die Achse stehenbleibt.
- Bei Annäherung an die Zielposition wird die Geschwindigkeit entsprechend der restlichen Strecke angepasst. Diese Anpassung erfolgt kontinuierlich und kompensiert somit auch ein ungenaues Abbremsen.
- Nichtlinearitäten werden mitkompensiert. Allerdings können sie als störende Unregelmäßigkeiten in der Beschleunigung erkennbar werden. In diesem Fall kann das Verhalten über eine genauere Linearisierung verbessert werden.

- Der beim zeitgeführten Prinzip unvermeidliche ständig aktive Lageregler erhöht die Schwingneigung und erzeugt unerwünscht Änderungen der Geschwindigkeiten. Bei elektromechanischen Achsen ist dieser Effekt weniger ausgeprägt und wird toleriert. Auf hydraulische Achsen wirken erheblich mehr Anregungsquellen ein und sie sind niederfrequenter und weniger stark gedämpft. Der Effekt ist deutlich ausgeprägt und oft stark störend.
- Die Genauigkeit im Ziel ist nicht abhängig vom verwendeten Verfahren. Eine Abweichung des Achsverhaltens vom Ideal wird beim zeitgeführten Prinzip „vertikal“ durch eine aufaddierte Reglerausgabe beantwortet. Beim weggeführten Prinzip erfolgt die Reaktion durch ein „horizontales“ Strecken oder Stauchen des Profils.
- Beim zeitgeführten Prinzip werden sich zwei mit gleichen Parametern betriebene und gleichzeitig mit gleichen Kommandos gestartete Achsen bewegen als wären sie mechanisch verbunden. Beide Achsen sind zum richtigen Zeitpunkt am richtigen Ort mit der richtigen Geschwindigkeit unterwegs. Die Abweichung beschränkt sich auf die typischerweise kleinen Schleppfehler und werden nicht aufintegriert.
- Beim weggeführten Prinzip werden Einflüsse aus dem Prozess oder selbst Exemplarstreuungen der Komponenten nicht kompensiert. Abweichungen werden innerhalb einer Bewegung aufintegriert. Es entsteht kein sicher zu erwartender Zusammenhang zwischen zwei mit gleichen Parametern betriebene und gleichzeitig mit gleichen Kommandos gestartete Achsen. Sie werden mit der erreichbaren Genauigkeit in das Ziel positioniert, kommen dort aber nicht unbedingt genau gleichzeitig an.

### Aufbau der Library

Im Gegensatz zur NC arbeiten die Funktionalitäten der Library vollständig in der PLC Runtime. Dies hat mehrere Konsequenzen, die nachstehend aufgeführt sind.

- Auch interne Bausteine sind in der Regel sichtbar. Der Online View wird dadurch weniger übersichtlich. Andererseits sind lokale Variablen für eine Analyse nutzbar.
- Alle Parameter und selbst Laufzeitvariablen sind sichtbar und auch zugreifbar. Dadurch entstehen Möglichkeiten für spezifische Manipulationen. Es sollte leicht verständlich sein, dass dabei mit größter Vorsicht gearbeitet werden sollte.
- Es wird nichts getan ohne dass direkt oder indirekt ein entsprechender Baustein aufgerufen wird. Anders als bei der NC ist die innere Arbeitsweise des Motion Controls klar zu erkennen. Davon sind vor allem betroffen:
  - Laden und Speichern von Parametern.
  - Erfassung von Istwerten.
  - Sollwertgenerierung.
  - Regelung.
  - Ausgabe-Anpassung.
- Im Gegensatz zur NC gibt es keine „fertigen“ Achsen. Dies erhöht den Anfangsaufwand, bietet aber auch Möglichkeiten für die Realisierung von angepassten Eigenschaften.
- Da die Achse in der PLC-Applikation aufgebaut wird ergeben sich viele Möglichkeiten, unerwartete und schwer nachzuvollziehende Effekte durch eine inkorrekte Reihenfolge oder Kombination der aufgerufenen Bausteine zu erzeugen. Es ist unbedingt empfehlenswert, sich hier an den Beispielen zu orientieren.
- Da die Bausteine von der PLC aufgerufen werden arbeitet hier auch das Motion Control mit der Zykluszeit der PLC Task. Es ist eine Task mit einer eher NC-typischen Zykluszeit von deutlich unter 10 ms zu nutzen.

Um eine bessere Verständlichkeit der Projekte zu erreichen sind die wichtigsten Bausteine nach den Standards der PLCopen ausgeführt. Durch diese Norm wird unter anderem festgelegt, dass die Bausteine über eine AxisRef genannte Referenz mit einer Achse verbunden sind. Da es bei der Library keine verdeckte Task-Ebene gibt, sind alle für die Achse benötigten Daten (Parameter, Laufzeitwerte) in dieser Struktur integriert. Die Kommunikation der Bausteine einer Achse erfolgt durch gemeinsame Benutzung dieser Referenz. Ausgenommen sind hier nur die durch PLCopen festgelegten Signale. Der Eingang Execute kann z. B. vom Ausgang Done eines anderen Bausteins gesteuert werden, um einen gewünschten Ablauf zu erzeugen.

## Aufbau einer Applikation

In einer mit der Hydraulik-Library realisierten PLC-Applikation sind zunächst drei Themen zu unterscheiden:

- Auf die Gesamtheit der Achsen bezogene System-Bausteine. Hierzu gehört die Kommunikation mit dem IBN-Tool PlcMcManager oder die Handhabung der Meldungsaufzeichnung. Unabhängig von der Achszahl sind diese Bausteine genau einmal pro Projekt zu instanzieren und genau einmal pro Zyklus aufzurufen. Dies sollte naheliegender Weise aus dem Main() eines Programms heraus erfolgen.
- Für den Aufbau einer Achse verwendete Bausteine. Hierzu gehören z. B. der Encoder-Baustein, der Sollwertgenerator, usw. Von diesen Bausteinen ist genau eine Instanz pro Achse anzulegen. Der Aufruf sollte genau einmal pro Zyklus erfolgen.
- Auf eine Achse bezogene Bausteine. Hierzu gehören z. B. der MC\_MoveAbsolute\_BkPlcMc -Baustein, der MC\_Stop\_BkPlcMc-Baustein, usw. Von diesen Bausteinen kann mehr als eine Instanz pro Achse angelegt werden. Der Aufruf hat in der Regel genau einmal pro Zyklus zu erfolgen.

Sollte die Applikation nur eine Achse besitzen, wird dieser Unterschied weniger deutlich, ist aber trotzdem zu beachten.

## Systembausteine

Zu den Systembausteinen gehören:

- MC\_AxAdsCommServer\_BkPlcMc()

Dieser Baustein stellt für die Summe der Achsen eine ADS-Verbindung für den PlcMcManager zu Verfügung. Wird dieser Baustein nicht zyklisch aufgerufen kommt keine Verbindung zu Stande.

- MC\_AxRtLoggerSpool\_BkPlcMc() oder MC\_AxRtLoggerDespool\_BkPlcMc

Dieser Baustein verwaltet den Meldungs-Puffer. Wird nicht genau einer dieser Bausteine zyklisch aufgerufen läuft der Meldungs-Puffer über und spätere Meldungen gehen verloren.

Die Systembausteine benötigen, wie leicht zu erkennen ist, den Zugriff auf alle betroffenen Strukturen. Gleichzeitig benötigen auch die achsbezogenen Bausteine Zugriff. Dies ist auf einfache Weise dadurch sicherzustellen, dass die Strukturen als VAR\_GLOBAL angelegt sind. Dies ist in den Beispielen gezeigt und gilt vor allem für:

- Die Achsreferenzen. Sie sind als ARRAY[1..Achsanzahl] OF Axis\_Ref\_BkPlcMc anzulegen.
  - Damit ist es nicht möglich, die Achsreferenzen in Module der Applikation zu verteilen.
  - Hier gibt es eine alternative Arbeitsweise, die mit POINTER-Listen arbeitet. Dabei ist eine besondere Sorgfalt erforderlich. Diese Methode ist somit nicht für den allgemeinen Gebrauch zu empfehlen.
- Der Meldungspuffer vom Typ ST\_TcPlcMcLogBuffer. Der Puffer wird von allen Achsen gemeinsam genutzt und somit kann der Verwaltungsbaustein nicht einer Achse zugeordnet werden.

## Bausteine für den Aufbau einer Achse

Hierzu gehören immer:

- Der Initialisierungs-Baustein MC\_AxUtiStandardInit\_BkPlcMc().
- Die Bausteine der Istwerterfassung. Das ist immer ein Baustein vom Typ MC\_AxRtEncoder\_BkPlcMc() sowie je nach Bedarf einer oder mehrere Bausteine zum Ermitteln von Drücken oder Kräften. Falls notwendig können Filterungen eingesetzt werden.
- Ein Baustein vom Typ MC\_AxRuntime\_BkPlcMc() für die Sollwertgenerierung. Dieser Baustein enthält einen üblichen Lageregler.
- Ein Baustein vom Typ MC\_AxRtFinish\_BkPlcMc() oder MC\_AxRtFinishLinear\_BkPlcMc. Hier werden diverse Ausgabe-Größen zusammengeführt und eine abschnittsweise oder kennliniengesteuerte Ausgabe-Linearisierung durchgeführt.
- Ein Baustein vom Typ MC\_AxRtDrive\_BkPlcMc(), der die Anpassung an die E/A-Variablen der Ausgabe-Hardware vornimmt.

Bei Bedarf ist dieser Minimal-Aufbau durch Bausteine zu ergänzen, die der Achse weitere Fähigkeiten verleihen. Hierher gehören z. B. Bausteine für die Regelung von Drücken oder Kräften, eine alternative Lageregelung oder für die automatische Vermessung von Kennlinien. Um wirksam zu sein, sind die Aufrufe dieser Bausteine an der richtigen Position zwischen den oben genannten Bausteinen einzufügen.

Die Übersichtlichkeit der Applikation kann dadurch verbessert werden, dass diese Bausteine zu einem Achsbaustein mit allgemeinen Schnittstellen zusammengefasst werden.

### **Achsbezogene Bausteine**

Hierher gehören die üblichen Bausteine für die Gestaltung des Arbeitszyklus einer Achse.

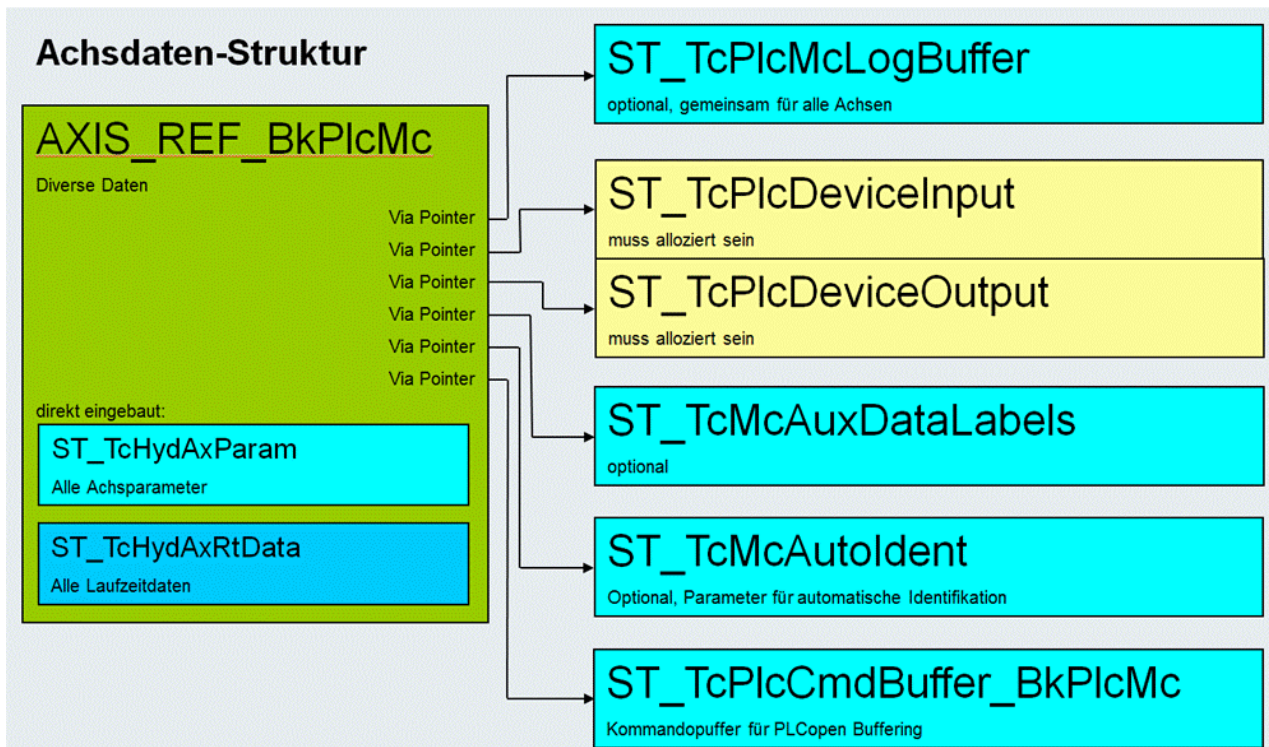
- MC\_Power\_BkPlcMc
- MC\_MoveAbsolute\_BkPlcMc
- MC\_Stop\_BkPlcMc
- MC\_Reset\_BkPlcMc
- MC\_Home\_BkPlcMc
- MC\_GearIn\_BkPlcMc
- MC\_GearOut\_BkPlcMc
- usw.

Da das Verhalten dieser Bausteine den Definitionen der PLCopen entspricht, können sie weitgehend wie die entsprechenden Bausteine der TC\_MC Bibliotheken verwendet werden. Allerdings senden die Bausteine dieser Bibliotheken nur Kommandos an den NC-Treiber und beobachten dessen Reaktionen und Rückmeldungen. Diverse Bausteine der Hydraulik-Library enthalten wesentliche Teile der Funktionalität und müssen kontinuierlich und in jedem Zyklus aufgerufen werden. Dies ist bei der Erstellung der Applikation zu berücksichtigen.

### 3 Genereller Aufbau

#### 3.1 Gliederung der Dokumentation

Jede Achse besteht aus einer Achsstruktur namens "Axis\_ref\_BkPlcMc", die aus verschiedenen ausgelagerten Strukturen zusammengesetzt ist. In dieser Achsstruktur befinden sich alle Daten (Laufzeitdaten sowie Parameterdaten) zu dieser Achse.



Bestimmte Bausteine müssen in jeder Applikation vorhanden sein um eine Achse überhaupt verfahren zu können. Zu diesen Bausteinen zählen:

- [MC\\_AxUtiStandardInit\\_BkPlcMc \[▶ 238\]](#) : Initialisierung und Überwachung verschiedener Bestandteile der Achse. Ein solcher FB sollte zyklisch aufgerufen werden. Erst nach einer erfolgreichen Initialisierung dürfen Bausteine wie **MC\_Power\_BkPlcMc** usw. aufgerufen werden.
- [MC\\_Power\\_BkPlcMc \[▶ 27\]](#) : Der Funktionsbaustein dient zum Ansteuern eines externen Stellgeräts. Der Baustein gibt Freigaben an z. B. Ventilendstufen oder Frequenzumrichter heraus.
- [MC\\_AxStandardBody\\_BkPlcMc \[▶ 237\]](#) : Der Funktionsbaustein ruft jeweils einen Baustein vom Typ [MC\\_AxRtEncoder\\_BkPlcMc \[▶ 183\]](#): Ermittlung der Istposition der Achse aus den Eingangsinformationen einer Hardware-Baugruppe.  
[MC\\_AxRuntime\\_BkPlcMc \[▶ 221\]](#): Übernimmt die Profilerzeugung.  
[MC\\_AxRtFinish\\_BkPlcMc \[▶ 230\]](#): Anpassung des Stellwertes an die Besonderheiten der Achse (Kennlinienlinearisierung)  
[MC\\_AxRtDrive\\_BkPlcMc \[▶ 172\]](#): Der Funktionsbaustein übernimmt die Aufbereitung des Stellwertes der Achse für die Ausgabe auf einer Hardware-Baugruppe.
- [MC\\_AxAdsCommServer\\_BkPlcMc \[▶ 263\]](#) : Stellt die Verbindung zu PlcMcManager her und überwacht diese. Dieser Baustein muss unabhängig von der Initialisierung aufgerufen werden. Nur so ist eine Inbetriebnahme ohne bereits vorhandene Parameter möglich.

Optional sinnvolle Bausteine sind:

- [MC\\_AxRtLoggerSpool\\_BkPlcMc \[▶ 244\]](#) : Der Funktionsbaustein stellt sicher, dass der LogBuffer der Library nicht überläuft.
- [MC\\_AxParamDelayedSave\\_BkPlcMc](#): Führt ein automatisches Speichern der Achsparameter durch.



Für die Inbetriebnahme wird der so genannte "PlcMcManager" bereitgestellt. Dieser soll die Inbetriebnahme der Anlage erleichtern, da wesentliche Einstellparameter in diesem Tool zusammengefasst dargestellt werden.

Für die ersten "Gehversuche" eignet sich das erste Beispiel hervorragend.

Funktionsgruppen	Beschreibung
<a href="#">Verwaltungsfunktionen [▶ 18]</a>	Funktionen für die Verwaltung und Kontrolle von Achsen, Zugriff auf Parameter und Zustände.
<a href="#">Einzelachsbewegungsfunktionen [▶ 19]</a>	Auslösen und Überwachen von aktiven Bewegungen für Einzelachsen.
<a href="#">Achsgruppenbewegungsfunktionen [▶ 20]</a>	Auslösen und Überwachen von aktiven Bewegungen für Achsgruppen.
<a href="#">Antriebsanpassungen [▶ 20]</a>	Bausteine zur Vorbereitung von Achsstellwerten für die Ausgabe auf Ausgabegeräte (Klemmen, Steller usw.) in der Peripherie.
<a href="#">Encoderanpassungen [▶ 20]</a>	Bausteine zur Auswertung von Istpositionsdaten, die von Eingabegeräte (Klemmen, Encoder usw.) in der Peripherie eingelesen wurden.
<a href="#">Parameterhandhabung [▶ 20]</a>	Bausteine für das Speichern, Lesen und Kommunizieren von Parametern.
<a href="#">Bewegungsgeneratoren [▶ 20]</a>	Stellwertgeneratoren für aktive Achsbewegungen
<a href="#">Regler [▶ 21]</a>	Regler für verschieden Zustandsgrößen: Position, Geschwindigkeit, Druck.
<a href="#">Tabellenfunktionen [▶ 21]</a>	Tabellenfunktionen für nichtlineare Abbildungen und Kurvenscheiben
<a href="#">Message Logging [▶ 22]</a>	Meldungsaufzeichnung.
<a href="#">Laufzeitfunktionen [▶ 22]</a>	Diverse Laufzeitfunktionen.
Datentypen	In der Bibliothek verwendete <a href="#">Aufzählungen [▶ 22]</a> und <a href="#">Strukturen [▶ 23]</a>

### 3.2 Funktionen, Bausteine und Typen (ab V3.0)



Hier werden alle vorhandenen Funktionen, Bausteine und Datentypen dieser Library aufgelistet.

Antworten zu häufig gestellten Fragen sowie Hinweise zum Einsatz der Library, zu Inbetriebnahme und Problemanalyse sowie Beispielprojekte finden Sie in der [Knowledge Base \[▶ 302\]](#).

Einige der hier aufgeführten Komponenten sind nicht für die Benutzung durch eine Applikation vorgesehen. Ihr Vorhandensein, Interface und Verhalten wird dementsprechend nicht garantiert. Da eine TwinCAT PLC Library jedoch strikt offen ist besteht keine Möglichkeit, diese internen Komponenten zu verbergen. Es sollte jedoch unbedingt darauf verzichtet werden, diese entsprechend mit (internal use only) oder (not recommended) gekennzeichneten Komponenten aus einer Applikation heraus direkt zu benutzen. Sollten eine dieser Komponenten für Sie von praktischem Nutzen sein nehmen Sie bitte Kontakt mit unserem Support auf. Wir werden dann die Möglichkeit prüfen, ob Ihnen ein Baustein unabhängig von der Library zur eigenverantwortlichen Verfügung gestellt werden kann.

Sollte die Library Bausteine, Typen oder Konstanten enthalten, die in der Dokumentation nicht aufgeführt werden handelt es sich um nicht freigegebene Elemente, die Gegenstand der aktuellen Softwarepflege und -entwicklung sind. Diese Elemente dürfen auf keinen Fall in einer Applikation direkt verwendet werden, da sie in der Regel noch nicht getestet sind.



Die Hydraulik Library bietet im Zusammenhang mit elektrischen Antrieben nur einen eingeschränkten Funktionsumfang. Ein erheblich weiteres Spektrum und eine umfassende Unterstützung bei Inbetriebnahme und Diagnose bieten TwinCAT NC PTP, NC I und CNC.



Es steht eine Reihe von Bibliotheken zur Verfügung, die auf eine typische Anordnung von Achsen oder spezielle Funktionalitäten ausgerichtet sind. Diese Bibliotheken setzen die TcPlcHydraulics Library voraus und sind getrennt zu bestellen.

Name	Beschreibung
TcPlcLibHydraulics_30_2R2Vgantry.LIB	in Vorbereitung
TcPlcLibHydraulics_30_4R3Vgantry.LIB	in Vorbereitung

**PLC open Motion Control**

Die hier aufgeführten Bausteine orientieren sich an:

Technical Specification

PLCopen - Technical Comittee 2 - Task Force

Function blocks for motion control

Part 1 Version 1.1 und Part 2 Version 0.99F (Definition noch nicht endgültig)

Die Namen dieser Bausteine beginnen mit MC\_ und enden mit \_BkPlcMc.



Teile der PLCopen Definitionen sind noch nicht endgültig. Es kann hierzu Änderungen in zukünftigen Versionen der Library geben.

Diese Änderungen können betreffen

- Namen, Verhalten oder sogar Existenz von Funktionen, Funktionsblöcken oder abgeleiteten Datentypen
- Namen, Verhalten, Typen oder Vorhandensein von Eingangs- oder Ausgangssignalen

**Administrative Function Blocks**

Name	Beschreibung
MC_CamTableSelect_BkPlcMc [ <a href="#">▶ 52</a> ]	Der Funktionsbaustein initialisiert eine Variable vom Typ ST_TcPlcMcCamId und bereitet dadurch eine Kurvenscheibe für die Kopplung von zwei Achsen vor.
MC_Power_BkPlcMc [ <a href="#">▶ 27</a> ]	Funktionsbaustein zum Ansteuern eines externen Stellgeräts.
MC_ReadActualPosition_BkPlcMc [ <a href="#">▶ 29</a> ]	Die Istposition einer Achse wird ermittelt.
MC_ReadActualTorque_BkPlcMc [ <a href="#">▶ 30</a> ]	Die Istkraft bzw. der Istdruck einer Achse wird ermittelt.
MC_ReadActualVelocity_BkPlcMc [ <a href="#">▶ 31</a> ]	Die Istgeschwindigkeit einer Achse wird ermittelt.

Name	Beschreibung
<a href="#">MC_ReadAxisError_BkPlcMc [▶ 32]</a>	Der aktuelle Errorcode einer Achse wird ermittelt.
<a href="#">MC_ReadBoolParameter_BkPlcMc [▶ 33]</a>	Die boolschen Parameter einer Achse werden ausgelesen.
<a href="#">MC_ReadDigitalOutput_BkPlcMc [▶ 34]</a>	Der aktuelle Zustand eines digitalen Ausgangs eines Nockenschaltwerks wird ermittelt.
<a href="#">MC_ReadParameter_BkPlcMc [▶ 35]</a>	Die nicht boolschen Parameter einer Achse werden ausgelesen.
<a href="#">MC_ReadStatus_BkPlcMc [▶ 37]</a>	Der Zustand der Achse wird dekodiert.
<a href="#">MC_Reset_BkPlcMc [▶ 39]</a>	Die Achse wird in einen betriebsbereiten Zustand versetzt.
<a href="#">MC_ResetAndStop_BkPlcMc [▶ 40]</a>	Die Achse wird in einen betriebsbereiten Zustand versetzt und befindet sich im Stillstand.
<a href="#">MC_SetOverride_BkPlcMc [▶ 41]</a>	Der Override der Achse wird gesetzt.
<a href="#">MC_SetPosition_BkPlcMc [▶ 43]</a>	Die Istposition der Achse wird gesetzt.
<a href="#">MC_SetReferenceFlag_BkPlcMc [▶ 44]</a>	Das Referenzier-Flag der Achse wird definiert. (Funktion ist nicht durch PLCopen definiert)
<a href="#">MC_WriteBoolParameter_BkPlcMc [▶ 45]</a>	Die boolschen Parameter einer Achse werden geschrieben.
<a href="#">MC_WriteDigitalOutput_BkPlcMc [▶ 46]</a>	Der aktuelle Zustand eines digitalen Ausgangs eines Nockenschaltwerks wird definiert.
<a href="#">MC_WriteParameter_BkPlcMc [▶ 47]</a>	Die nicht boolschen Parameter einer Achse werden geschrieben.

**Motion Function Blocks, Single Axis**

Name	Beschreibung
<a href="#">MC_DigitalCamSwitch_BkPlcMc [▶ 53]</a>	Erzeugung von Software-Nocken in Abhängigkeit von Position, Bewegungsrichtung und Geschwindigkeit einer Achse.
<a href="#">MC_EmergencyStop_BkPlcMc [▶ 56]</a>	Stoppen einer Bewegung ohne Erreichen der Zielposition. (Funktion ist nicht durch PLCopen definiert)
<a href="#">MC_Halt_BkPlcMc [▶ 67]</a>	Stoppen einer Bewegung ohne Erreichen der Zielposition.
<a href="#">MC_Home_BkPlcMc [▶ 68]</a>	Auslösung und Überwachung einer Referenzfahrt.
<a href="#">MC_ImmediateStop_BkPlcMc [▶ 71]</a>	Stoppen einer Bewegung ohne Erreichen der Zielposition. (Funktion ist nicht durch PLCopen definiert)
<a href="#">MC_MoveAbsolute_BkPlcMc [▶ 72]</a>	Start und Überwachung einer Positionierung mit angegebener Geschwindigkeit auf eine absolut angegebene Zielkoordinate.
<a href="#">MC_MoveJoySticked_BkPlcMc [▶ 74]</a>	Starten und Kontrollieren einer Achsbewegung mit einem proportionalen Steuergerät. (Funktion ist nicht durch PLCopen definiert)
<a href="#">MC_MoveRelative_BkPlcMc [▶ 76]</a>	Start und Überwachung einer Positionierung mit angegebener Geschwindigkeit um eine absolut angegebene Strecke.
<a href="#">MC_MoveVelocity_BkPlcMc [▶ 78]</a>	Start und Überwachung einer Positionierung mit angegebener Geschwindigkeit, aber ohne Zielangabe.
<a href="#">MC_RampedStop_BkPlcMc [▶ 79]</a>	Stoppen einer Bewegung mit einer reinen Zeitrampe.
<a href="#">MC_Stop_BkPlcMc [▶ 81]</a>	Stoppen einer Bewegung ohne Erreichen der Zielposition.

**Motion Function Blocks, Multiple Axis**

Name	Beschreibung
<a href="#">MC_CamIn_BkPlcMc [▶ 48]</a>	Der Funktionsbaustein startet und überwacht eine Kurvenscheiben-Kopplung zwischen zwei Achsen.
<a href="#">MC_CamOut_BkPlcMc [▶ 50]</a>	Der Funktionsbaustein löst eine Kurvenscheiben-Kopplung zwischen zwei Achsen.
<a href="#">MC_GearIn_BkPlcMc [▶ 62]</a>	Start und Überwachung der Getriebe-Kopplung von zwei Achsen.
<a href="#">MC_GearInPos_BkPlcMc [▶ 64]</a>	Fliegende Getriebe-Kopplung von zwei Achsen.
<a href="#">MC_GearOut_BkPlcMc [▶ 66]</a>	Auflösen der Getriebe-Kopplung von zwei Achsen.

**System Function Blocks**

Name	Beschreibung
<a href="#">MC_AxRtDrive_BkPlcMc [▶ 172]</a>	Aufbereitung des Stellwerts der Achse für die Ausgabe auf einer Hardware-Baugruppe, Mapping-Hinweise.
<a href="#">MC_AxRtEncoder_BkPlcMc [▶ 183]</a>	Ermittlung der Istposition der Achse aus den Eingangsinformationen einer Hardware-Baugruppe, Mapping-Hinweise.
<a href="#">MC_AxRtFinish_BkPlcMc [▶ 230]</a>	Anpassung des erzeugten Stellwerts an die Besonderheiten der Achse.
<a href="#">MC_AxRtFinishLinear_BkPlcMc [▶ 231]</a>	Anpassung des erzeugten Stellwerts an die Besonderheiten der Achse unter Berücksichtigung einer Kennlinie.
<a href="#">MC_AxRuntime_BkPlcMc [▶ 227]</a>	Stellwertgenerierung und Lageregelung der Achse.
<a href="#">MC_AxRtGenerator_BkPlcMc [▶ 221]</a>	Stellwertgenerierung der Achse.
<a href="#">MC_AxRtController_BkPlcMc [▶ 229]</a>	Lageregelung der Achse.

**System Function Blocks, andere Istwerte**

Name	Beschreibung
<a href="#">MC_AxRtReadForceDiff_BkPlcMc [▶ 199]</a>	Ermittlung der differentiellen Istkraft einer Achse.
<a href="#">MC_AxRtReadForceSingle_BkPlcMc [▶ 202]</a>	Ermittlung der einseitigen Istkraft einer Achse.
<a href="#">MC_AxRtReadPressureDiff_BkPlcMc [▶ 204]</a>	Ermittlung des differentiellen Istdrucks einer Achse.
<a href="#">MC_AxRtReadPressureSingle_BkPlcMc [▶ 207]</a>	Ermittlung des einseitigen Istdrucks einer Achse.

**System Function Blocks, Parameter**

Name	Beschreibung
<a href="#">MC_AxAdsCommServer_BkPlcMc [▶ 263]</a>	Die Applikation wird mit den Fähigkeiten eines ADS-Servers ausgestattet.
<a href="#">MC_AxAdsReadDecoder_BkPlcMc [▶ 266]</a>	Der Funktionsbaustein dekodiert ADS-Read-Zugriffe für einen ADS-Server.
<a href="#">MC_AxAdsWriteDecoder_BkPlcMc [▶ 268]</a>	Der Funktionsbaustein dekodiert ADS-Write-Zugriffe für einen ADS-Server.
<a href="#">MC_AxAdsPtrArrCommServer_BkPlcMc [▶ 265]</a>	Die Applikation wird mit den Fähigkeiten eines ADS-Servers ausgestattet.
<a href="#">MC_AiParamAuxLabelsLoad_BkPlcMc [▶ 269]</a>	Die Beschriftungstexte für die kundenspezifischen Achsparameter aus einer Datei laden.
<a href="#">MC_AiParamLoad_BkPlcMc [▶ 270]</a>	Parameter einer Achse aus einer Datei laden.
<a href="#">MC_AiParamSave_BkPlcMc [▶ 271]</a>	Parameter einer Achse in eine Datei schreiben.
<a href="#">MC_AiParamDelayedSave_BkPlcMc [▶ 245]</a>	Verzögertes Schreiben der Achs-Parameter.

Name	Beschreibung
<a href="#">MC_AxUtiReadCoeDriveTerm_BkPlcMc [▶ 272]</a>	Inhalt eines Registers aus der EL-Klemme lesen, die als Antriebsschnittstelle für die Achse dient.
<a href="#">MC_AxUtiReadCoeEncTerm_BkPlcMc [▶ 274]</a>	Inhalt eines Registers aus der EL-Klemme lesen, die als Encoderschnittstelle für die Achse dient.
<a href="#">MC_AxUtiReadRegDriveTerm_BkPlcMc [▶ 276]</a>	Inhalt eines Registers aus der KL-Klemme lesen, die als Antriebsschnittstelle für die Achse dient.
<a href="#">MC_AxUtiReadRegEncTerm_BkPlcMc [▶ 277]</a>	Inhalt eines Registers aus der KL-Klemme lesen, die als Encoderschnittstelle für die Achse dient.
<a href="#">MC_AxUtiUpdateRegDriveTerm_BkPlcMc [▶ 278]</a>	Einen Parametersatz in die Register einer KL-Klemme schreiben, die als Antriebsschnittstelle für die Achse dient.
<a href="#">MC_AxUtiUpdateRegEncTerm_BkPlcMc [▶ 280]</a>	Einen Parametersatz in die Register einer KL-Klemme schreiben, die als Encoderschnittstelle für die Achse dient.
<a href="#">MC_AxUtiWriteCoeDriveTerm_BkPlcMc [▶ 282]</a>	Inhalt eines Registers in der EL-Klemme schreiben, die als Antriebsschnittstelle für die Achse dient.
<a href="#">MC_AxUtiWriteCoeEncTerm_BkPlcMc [▶ 283]</a>	Inhalt eines Registers in der EL-Klemme schreiben, die als Encoderschnittstelle für die Achse dient.
<a href="#">MC_AxUtiWriteRegDriveTerm_BkPlcMc [▶ 285]</a>	Inhalt eines Registers in der KL-Klemme schreiben, die als Antriebsschnittstelle für die Achse dient.
<a href="#">MC_AxUtiWriteRegEncTerm_BkPlcMc [▶ 286]</a>	Inhalt eines Registers in der KL-Klemme schreiben, die als Encoderschnittstelle für die Achse dient.
<a href="#">MC_LinTableExportToAsciiFile_BkPlcMc [▶ 259]</a>	Der Funktionsbaustein exportiert eine Linearisierungstabelle in eine Datei im ASCII Format.
<a href="#">MC_LinTableExportToBinFile_BkPlcMc [▶ 260]</a>	Der Funktionsbaustein exportiert eine Linearisierungstabelle in eine Datei im Binär-Format.
<a href="#">MC_LinTableImportFromAsciiFile_BkPlcMc [▶ 261]</a>	Der Funktionsbaustein importiert eine Linearisierungstabelle aus einer Datei im ASCII Format.
<a href="#">MC_LinTableImportFromBinFile_BkPlcMc [▶ 262]</a>	Der Funktionsbaustein importiert eine Linearisierungstabelle aus einer Datei im Binär-Format.

**System Function Blocks, Controllers**

Name	Beschreibung
<a href="#">MC_AxCtrlAutoZero_BkPlcMc [▶ 151]</a>	Automatischer Nullpunktgleich.
<a href="#">MC_AxCtrlPressure_BkPlcMc [▶ 156]</a>	Regler für eine aufbauend wirkende Druckregelung.
<a href="#">MC_AxCtrlPressureFF_Ex_BkPlcMc [▶ 160]</a>	Erweiterter Regler für eine aufbauend wirkende Druckregelung.
<a href="#">MC_AxCtrlPullbackOnPressure_BkPlcMc</a>	Regler für eine verdrängend wirkende Druckregelung.
<a href="#">MC_AxCtrlSlowDownOnPressure_BkPlcMc [▶ 162]</a>	Regler für eine ablösende Druckregelung.
<a href="#">MC_AxCtrlStepperDeStall_BkPlcMc [▶ 167]</a>	Überwachung der Bewegung einer Schrittmotorachse.
<a href="#">MC_AxCtrlVelocity_BkPlcMc</a>	Regler für die Achs-Geschwindigkeit.
<a href="#">MC_AxCtrlVeloMoving_BkPlcMc</a>	Regler für die Achs-Geschwindigkeit.

**System Function Blocks, TableFunctions**

Name	Beschreibung
<a href="#">MC_AxTableFromAsciiFile_BkPlcMc [▶ 213]</a>	Lesen des Inhalts einer Tabelle aus einer Textdatei.
<a href="#">MC_AxTableFromBinFile_BkPlcMc [▶ 215]</a>	Lesen des Inhalts einer Tabelle aus einer binäre Datei.
<a href="#">MC_AxTableReadOutNonCyclic_BkPlcMc [▶ 216]</a>	Funktionsbaustein zur Ermittlung der einem Master-Wert zugeordneten Slave-Werte mit Hilfe einer Tabelle.
<a href="#">MC_AxTableToAsciiFile_BkPlcMc [▶ 218]</a>	Schreiben des Inhalts einer Tabelle in eine Textdatei.
<a href="#">MC_AxTableToBinFile_BkPlcMc [▶ 219]</a>	Schreiben des Inhalts einer Tabelle in eine binäre Datei.

**System Function Blocks, Message Logging**

Name	Beschreibung
<a href="#">MC_AxRtLogAxisEntry_BkPlcMc [► 240]</a>	Eine achsbezogene Meldung wird in den LogBuffer der Library eingetragen.
<a href="#">MC_AxRtLogClear_BkPlcMc [► 241]</a>	Alle Einträge im LogBuffer löschen und initialisieren.
<a href="#">MC_AxRtLogEntry_BkPlcMc [► 242]</a>	Eine Meldung wird in den LogBuffer der Library eingetragen.
<a href="#">MC_AxRtLoggerDespool_BkPlcMc [► 242]</a>	Minimalanzahl von freien Meldungen im LogBuffer der Library sicherstellen.
<a href="#">MC_AxRtLoggerRead_BkPlcMc [► 243]</a>	Lesen einer Meldung aus dem LogBuffer der Library.
<a href="#">MC_AxRtLoggerSpool_BkPlcMc [► 244]</a>	Übertragung von Meldungen aus dem LogBuffer der Library in die Ereignisanzeige von Windows.

**System Function Blocks, Laufzeitfunktionen**

Name	Beschreibung
<a href="#">MC_AxRtCheckSyncDistance_BkPlcMc [► 228]</a>	Überwachung der Entfernung zwischen Referenznocken und Nullimpuls.
<a href="#">MC_AxRtCmdBufferExecute_BkPlcMc [► 240]</a>	Abarbeitung des Kommando-Puffers.
<a href="#">MC_AxRtCommandsLocked_BkPlcMc [► 246]</a>	Die Funktion erleichtert das Setzen und Löschen einer Schutzfunktion im Status-Doppelwort einer Achse.
<a href="#">MC_AxRtGoErrorState_BkPlcMc [► 233]</a>	(not recommended) Die Achse wird in einen Störzustand versetzt.
<a href="#">MC_AxRtMoveChecking_BkPlcMc [► 234]</a>	Überwachung der Bewegung einer Achse.
<a href="#">MC_AxRtSetDirectOutput_BkPlcMc [► 235]</a>	Direkte Ausgabe eines Stellwerts.
<a href="#">MC_AxRtSetExtGenValues_BkPlcMc [► 236]</a>	Versorgung einer Achse mit Führungsgrößen, die nicht aus dem achseigenen Generator stammen.
<a href="#">MC_AxStandardBody_BkPlcMc [► 237]</a>	Aufruf der üblichen Unterbestandteile einer Achse (Encoder, Generator, Finish, Drive).
<a href="#">MC_AxUtiAutoldent_BkPlcMc [► 254]</a>	Automatische Ermittlung von Parametern der Achse.
<a href="#">MC_AxUtiAutoldentSlave_BkPlcMc</a>	in Vorbereitung: Automatische Ermittlung von Parametern einer Slave-Achse.
<a href="#">MC_AxUtiAverageDerivative_BkPlcMc [► 249]</a>	Ermittlung der Ableitung eines Wertes durch numerische Differentiation über mehr als einen Zyklus.
<a href="#">MC_AxUtiPT1_BkPlcMc [► 250]</a>	Berechnung eines Tiefpass 1. Ordnung.
<a href="#">MC_AxUtiPT2_BkPlcMc [► 251]</a>	Berechnung eines Tiefpass 2. Ordnung.
<a href="#">MC_AxUtiSlewRateLimiter_BkPlcMc [► 252]</a>	Erzeugung einer anstiegsbegrenzten Rampe.
<a href="#">MC_AxUtiSlidingAverage_BkPlcMc [► 253]</a>	Ermittlung eines gleitenden Mittelwerts.
<a href="#">MC_AxUtiStandardInit_BkPlcMc [► 238]</a>	Initialisierung und Überwachung der Bestandteile einer Achse.
<a href="#">MC_FunctionGeneratorFD_BkPlcMc [► 211]</a>	Ein Funktionsgenerator.
<a href="#">MC_FunctionGeneratorSetFrq_BkPlcMc [► 212]</a>	Aktualisiert die Arbeitsfrequenz einer Zeitbasis für einen oder mehrere Funktionsgeneratoren.
<a href="#">MC_FunctionGeneratorTB_BkPlcMc [► 212]</a>	Aktualisiert eine Zeitbasis für einen oder mehrere Funktionsgeneratoren.

**Datentypen: Enumerationen**

Name	Beschreibung
<a href="#">E_TcMcCurrentStep [► 88]</a>	Diese Enumeration liefert Codes für die internen Zustände der Stellwertgeneratoren.

Name	Beschreibung
<a href="#">E_TcMcDriveType [► 90]</a>	Die Konstanten in dieser Enumeration werden zur Kennzeichnung von für die Stellwertausgabe einer Achse verwendeter Hardware benutzt.
<a href="#">E_TcMcEncoderType [► 93]</a>	Die Konstanten in dieser Enumeration werden zur Kennzeichnung von für die Istwerterfassung einer Achse verwendeter Hardware benutzt.
<a href="#">E_TcMCFbState [► 96]</a>	Diese Enumeration liefert Codes für den aktuellen Zustand einer Achse.
<a href="#">E_TcMcHomingType [► 96]</a>	Diese Enumeration liefert Codes für die Referenziermethode einer Achse.
<a href="#">E_TcMCPParameter [► 97]</a>	Die Konstanten in dieser Auflistung werden zur Parameternummerierung benutzt.
<a href="#">E_TcMcPressureReadingMode [► 111]</a>	Die Konstanten in dieser Auflistung legen fest, welcher Istwert in der <code>ST_TcHydAxRtData</code> Struktur der Achse mit dem Ergebnis einer Druck- oder Krafterfassung zu aktualisieren ist.
<a href="#">E_TcMcProfileType [► 110]</a>	Die Konstanten in dieser Auflistung werden zur Kennzeichnung von Stellwertgeneratoren benutzt.
<a href="#">E_TcPlcBufferedCmdType_BkPlcMc [► 86]</a>	In Vorbereitung: Die Konstanten in dieser Auflistung werden zur Kennzeichnung von bepufferten Achskommandos.
<a href="#">MC_BufferMode_BkPlcMc [► 112]</a>	Die Konstanten in dieser Auflistung werden zur Steuerung des Blendings nach PLC Open verwendet.
<a href="#">MC_Direction_BkPlcMc [► 115]</a>	Diese Enumeration liefert Codes für die Richtung einer Bewegung, wenn diese Information nicht in anderen Daten enthalten oder aus der Situation heraus feststellbar ist.
<a href="#">MC_HomingMode_BkPlcMc [► 115]</a>	Diese Enumeration liefert Codes für die Festlegung der Referenziermethode.
<a href="#">MC_StartMode_BkPlcMc [► 116]</a>	Die Konstanten in dieser Auflistung werden zur Kennzeichnung der Modi beim Starten von Achsen benutzt.

**Datentypen: Strukturen**

Name	Beschreibung
<a href="#">Axis_Ref_BkPlcMc [► 84]</a>	Eine Variable von diesem Typ enthält alle nötigen Variablen oder Pointer auf Variablen, die zu einer Achse gehören.
<a href="#">CAMSWITCH_REF_BkPlcMc [► 114]</a>	Eine Variable von diesem Typ wird an einen <code>MC_DigitalCamSwitch_BkPlcMc [► 53]</code> Baustein übergeben.
<a href="#">MC_CAM_ID_BkPlcMc [► 113]</a>	Eine Variable von diesem Typ enthält die Beschreibung einer zur Kopplung vorbereiteten Kurvenscheibe.
<a href="#">MC_CAM_REF_BkPlcMc [► 113]</a>	Eine Variable von diesem Typ enthält die Beschreibung einer bereitgestellten Kurvenscheibe.
<a href="#">OUTPUT_REF_BkPlcMc [► 117]</a>	Eine Variable von diesem Typ enthält Ausgabedaten eines <code>MC_DigitalCamSwitch_BkPlcMc [► 53]</code> Bausteins.
<a href="#">ST_FunctionGeneratorFD_BkPlcMc [► 117]</a>	Eine Variable von diesem Typ enthält Parameter für die Definition der Ausgangssignale eines Funktionsgenerators.
<a href="#">ST_FunctionGeneratorTB_BkPlcMc [► 118]</a>	Eine Variable von diesem Typ enthält Parameter für die Definition einer Zeitbasis für einen Funktionsgenerator.
<a href="#">ST_TcMcAutoldent [► 118]</a>	Eine Variable von diesem Typ enthält die Parameter für einen <code>MC_AxUtiAutoldent_BkPlcMc [► 254]</code> Baustein.

Name	Beschreibung
<a href="#">ST_TcMcAuxDataLabels [▶ 137]</a>	Eine Variable von diesem Typ enthält Beschriftungstexte der kundenspezifischen Achsparameter.
<a href="#">ST_TcHydAxParam [▶ 120]</a>	Eine Variable von diesem Typ enthält alle Parameter einer Achse.
<a href="#">ST_TcHydAxRtData [▶ 131]</a>	Eine Variable von diesem Typ enthält die Laufzeitdaten einer Achse.
<a href="#">ST_TcPlcMcLogBuffer [▶ 142]</a>	Eine Variable mit dieser Struktur bildet den LogBuffer der Library.
<a href="#">ST_TcPlcMcLogEntry [▶ 142]</a>	Eine Variable mit dieser Struktur enthält eine Meldung des LogBuffer der Library.
<a href="#">ST_TcPlcDeviceInput [▶ 138]</a>	Diese Struktur enthält die Eingangsabbild-Variablen einer Achse.
<a href="#">ST_TcPlcDeviceOutput [▶ 140]</a>	Diese Struktur enthält die Ausgangsabbild-Variablen einer Achse.
<a href="#">ST_TcPlcRegDataItem [▶ 143]</a>	Diese Struktur enthält einen Parametersatz für eine KL-Klemme.
<a href="#">ST_TcPlcRegDataTable [▶ 144]</a>	Diese Struktur enthält einen Parameter für eine KL-Klemme.
<a href="#">TRACK_REF_BkPlcMc [▶ 116]</a>	In Vorbereitung.

### 3.3 Die Hydraulikbibliothek

Um den Anforderungen der Hydraulik gerecht zu werden, sind für die Regelung spezielle Algorithmen notwendig. Die SPS Bibliotheken TcPlcHydraulics\_30 (für TC2) und TC2\_Hydraulics (für TC3) fassen eine Reihe von Bausteinen und Funktionen zum Thema hydraulische Achsen sowie die darin verwendeten Datentypen zusammen. Sie erweitern die Unterstützung dieser Antriebstechnologie indem sie den Betrieb von Achsen ermöglichen, die durch ihre Eigenschaften (Grenzfrequenz, stark streuendes Verhalten) nicht zur Lageregelung geeignet sind oder deren Aufgaben sich von dem bei elektrischen Servo-Achsen stark unterscheiden.

Das hier vorgestellte Produkt umfasst:

- die Softwarebibliothek "TcPlcHydraulics.lib" bzw. "Tc2\_Hydraulics.compiled-library"
- das Inbetriebnahme Tool "PlcMcManager.exe"

Um die Nutzung der Bibliothek zu vereinfachen sind die Bausteine entsprechend den Vorgaben der Nutzerorganisation der IEC61131 Anwender (PLCopen) ausgeführt und entsprechend zertifiziert.



Die Dokumentation der Version V2.1 wird auch weiterhin zur Verfügung stehen.



**Themen der Bibliothek:**

- Auswertung von [Encodern](#) [▶ 183]
- Auswertung von Druckmessdosen
- diverse Filterfunktionen
  - Pt1-Filter
  - [Gleitender-Mittelwert](#) [▶ 253]
  - [Anstiegsbegrenzung](#) [▶ 252]
- voller Zugriff auf interne Größen
- Bewegungssteuerung
- Regler für
  - Druck/Kraft
  - Position
  - Geschwindigkeit
  - Möglichkeit der eigenen Reglerentwicklung
- Synchronisierung von hydraulischen sowie elektrischen Achsen
- Anpassung der Stellwerte an Ausgabegeräte
- Vollständige Handhabung komplexer Geräte
- Message Logging
- Parameterhandhabung
  - Speicher- und Laderoutinen
  - Autosave
- Kennlinien-Linearisierung
  - abschnittsweise
  - [Kompensations-Kennlinie](#) [▶ 254]

**Folgende Bewegungssteuerungen werden unterstützt:**

1. Zeitgeführte Bewegungssteuerung:
  - Die steuernde Größe der Profilerzeugung ist die Zeit.
  - Der Generator „kennt“ die Achse nicht.
  - Nur der vorgesteuerte Lageregler stellt die Verbindung her.
2. Weggeführte Bewegungssteuerung:
  - Die steuernde Größe der Profilerzeugung ist der Restweg.
  - Der Generator „kennt“ die Achse.
  - Während der Bewegung ist keine Lageregelung möglich/nötig.
3. Abhängige Bewegungssteuerung:
  - Die Sollwerte werden über eine Abbildungsvorschrift (Getriebeformel, Kurventabelle) aus den Werten einer anderen Achse berechnet.
  - Der Generator „kennt“ die Achse nicht.
  - Nur der vorgesteuerte Lageregler stellt die Verbindung her.

**Weggeführte und zeitgeführte Bewegungssteuerung:**

Die zeitgeführte Bewegungssteuerung benutzt die Zeit als Bezugsvariable. Die Grundgleichungen sind

$$v=a*t \text{ und}$$

$$s=0.5*a*t*t.$$

Der Sollwertgenerator gibt eine Geschwindigkeit und eine Position vor, diese wird vom Lage- und Geschwindigkeitsregler ausgewertet und mit der aktuellen Position verrechnet.

Bei der weggesteuerten Positionierung wird im Gegensatz zur zeitgesteuerten der Stellwert der Achse als Funktion des Restwegs berechnet. Durch Umstellen der obigen Gleichungen erhält man

$$v = \sqrt{2 \cdot a \cdot s}$$

Beide Verfahren haben Vor- und Nachteile.

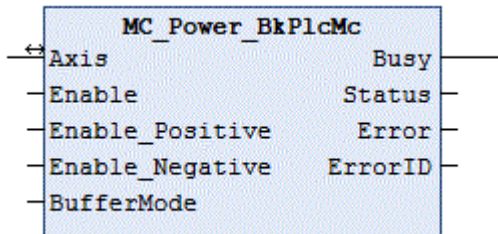
- Zeitgesteuerte Achsen benötigen gerade für Beschleunigungs- und Verzögerungsvorgänge einen geschlossenen Regelkreis. Nur über die Rückkopplung ist es dem Geschwindigkeitsregler möglich die richtige Ausgabegröße zu generieren. Ein solcher Regelkreis reagiert aber stark auf Stick/Slip-Effekte oder Versorgungsdruckschwankungen, wodurch das System leicht zum Schwingen angeregt wird.
- Weggesteuerte Achsen müssen nicht in einem geschlossenen Regelkreis betrieben werden. Damit ist dieses Verfahren wesentlich robuster gegenüber äußeren Störeinflüssen.
- Da die weggesteuerten Achsen nicht die Zeit sondern den Weg als Basis haben, wird eine Geschwindigkeit gestellt, aber nicht nachgeregelt. Aus diesem Grund ist die Positionierung von hydraulischen Achsen sehr robust.

Beide Verfahren werden von der Hydraulik-Bibliothek unterstützt und können auch kombiniert verwendet werden.

## 4 PLCopen Motion Control

### 4.1 Administrative

#### 4.1.1 MC\_Power\_BkPlcMc (ab V3.0)



Der Funktionsbaustein dient zum Ansteuern eines externen Stellgeräts. Weitere Informationen zu diesem Thema finden Sie unter [FAQ #9](#) [▶ 310].

#### Eingänge

```
VAR_INPUT
  Enable:          BOOL;
  Enable_Positive: BOOL;
  Enable_Negative: BOOL;
  BufferMode:      MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc; (ab/from V3.0.8)
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein TRUE an diesem Eingang schaltet ein externes Stellgerät einer Achse aktiv.
Enable_Positive	BOOL	Ein TRUE an diesem Eingang schaltet die richtungsbezogene Freigabe eines externen Stellgeräts einer Achse für Bewegungen in positiver Richtung aktiv.
Enable_Negative	BOOL	Ein TRUE an diesem Eingang schaltet die richtungsbezogene Freigabe eines externen Stellgeräts einer Achse für Bewegungen in negativer Richtung aktiv.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [▶ 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:    BOOL;
  Status:  BOOL;
```

```

Error:      BOOL;
ErrorID:    UDINT;
END_VAR

```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Status	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.

## Verhalten des Bausteins

Dieser Baustein dient zur Steuerung von externen Stellgeräten. Dabei kann es sich um Baugruppen zur Ventilansteuerung (Onboard Endstufe des Ventils oder Schaltschrank-Baugruppe), um Frequenzumrichter oder Servoverstärker handeln. Diese Geräte benötigen in der Regel ein digitales Signal zur Freigabe der Energieabgabe durch eine Leistungsstufe. Je nach Ausführung des Geräts können zusätzlich die Bewegungsrichtungen "Positiv" und "Negativ" gezielt aktiv geschaltet werden.

Die Eingangssignale des Bausteins werden in die Interfaces zum Peripherie-Gerät weitergereicht. Zusätzlich wird durch **Enable** eine Fehlerüberwachung aktiviert.

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

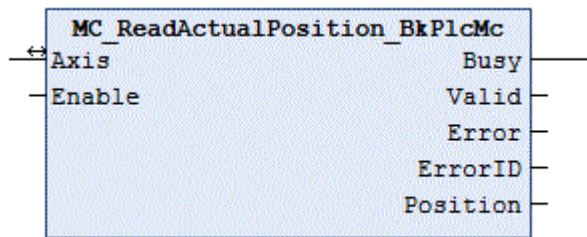
- Ist in nDrive\_Type in pStAxParams der Wert iTcMc\_DriveAx2000\_XXXXX eingestellt, wird wie folgt weiter verfahren:
  - Ist einer der Pointer pStDeviceOutput und pStDeviceInput in [Axis\\_Ref\\_BkPlcMc \[► 84\]](#) nicht initialisiert reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcDriveIn** bzw. **dwTcHydErrCdPtrPlcDriveOut**. **Status** ist dann FALSE.
  - Wenn in der Kommunikation mit dem AX-Gerät eine Störung erkannt wird oder im pStDeviceInput Interface eine Störmeldung des AX-Geräts auftritt reagiert der Baustein mit **Error** und einem **ErrorID**, der in den [Globalen Konstanten \[► 326\]](#) der Bibliothek definiert ist. **Status** ist dann FALSE und die Achse wird in einen Störzustand mit dem Achsfehler **dwTcHydErrCdDriveNotReady** versetzt.
  - Andernfalls wird am **Status** der Wert von **Enable** zurückgegeben.
- Ist in nDrive\_Type in pStAxParams der Wert iTcMc\_DriveKL2531 oder iTcMc\_DriveKL2541 eingestellt, wird wie folgt weiter verfahren:
  - Die Pointer pStDeviceOutput und pStDeviceInput in [Axis\\_Ref\\_BkPlcMc \[► 84\]](#) werden überprüft. Sind diese Pointer nicht initialisiert reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcDriveIn** bzw. **dwTcHydErrCdPtrPlcDriveOut**. **Status** ist dann FALSE.
  - Wenn in der Kommunikation mit der E/A-Klemme eine Störung erkannt wird oder im pStDeviceInput Interface eine Störmeldung der Klemme auftritt reagiert der Baustein mit **Error** und einem **ErrorID**, der in den [Globalen Konstanten \[► 326\]](#) der Bibliothek definiert ist. **Status** ist dann FALSE und die Achse wird in einen Störzustand mit dem Achsfehler **dwTcHydErrCdDriveNotReady** versetzt.
  - **Enable** wird benutzt um die Klemmenendstufe durch ein Bit in pStDeviceOutput.bTerminalCtrl zu aktivieren. Das Bereit-Signal in bTerminalCtrl.bTerminalState wird als **Status** zurückgegeben.
  - Der Wert von **Enable\_Positive** wird bei fehlerfreiem Zustand des Antriebsinterfaces mit der Maske **dwTcHydDcDwFdPosEna** in nDeCtrlDWord von pStAxRtData eingetragen.
  - Der Wert von **Enable\_Negative** wird bei fehlerfreiem Zustand des Antriebsinterfaces mit der Maske **dwTcHydDcDwFdNegEna** in nDeCtrlDWord von pStAxRtData eingetragen.
- Andernfalls wird eine Überprüfung der Pointer pStDeviceInput und pStDeviceOutput in [Axis\\_Ref\\_BkPlcMc \[► 84\]](#) vorgenommen. Sind diese Pointer nicht initialisiert reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcDriveIn** bzw. **dwTcHydErrCdPtrPlcDriveOut**. **Status** ist dann FALSE.
  - Andernfalls wird am **Status** der Wert von bPowerOk aus pStDeviceInput zurückgegeben.

- Der Wert von **Enable** wird bei fehlerfreiem Zustand des Antriebsinterfaces mit der Maske dwTcHydDcDwCtrlEnable in nDeCtrlDWord von pStAxRtData eingetragen.
- Der Wert von **Enable\_Positive** wird bei fehlerfreiem Zustand des Antriebsinterfaces mit der Maske dwTcHydDcDwFdPosEna in nDeCtrlDWord von pStAxRtData eingetragen.
- Der Wert von **Enable\_Negative** wird bei fehlerfreiem Zustand des Antriebsinterfaces mit der Maske dwTcHydDcDwFdNegEna in nDeCtrlDWord von pStAxRtData eingetragen.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

### 4.1.2 MC\_ReadActualPosition\_BkPlcMc (ab V3.0)



Der Funktionsbaustein ermittelt die aktuelle Position einer Achse.

#### Eingänge

```
VAR_INPUT
    Enable:    BOOL;
END_VAR
```

Nanme	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst eine Aktualisierung des Positionswertes aus.

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Nanme	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    Busy:      BOOL;
    Valid:     BOOL;
    Error:     BOOL;
    ErrorID:   UDINT;
    Position:  LREAL;
END_VAR
```

Nanme	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Valid	BOOL	Hier wird die erfolgreiche Ermittlung der Istposition signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Position	LREAL	[mm] Die Istposition.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Enable** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Befindet sich die Achse in einem gestörten Zustand und ist die Ursache ein Encoder-Problem wird mit **Error** und **ErrorID:=Errorcode** des Encoders reagiert.

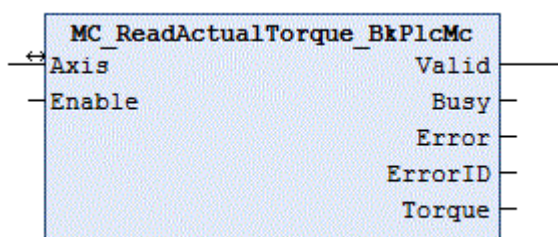
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Istposition ermittelt und **Valid** gemeldet.

Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

**4.1.3 MC\_ReadActualTorque\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein ermittelt die aktuelle Istkraft bzw. den Istdruck einer Achse.

**Eingänge**

```
VAR_INPUT
  Enable:      BOOL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst eine Aktualisierung des Istwertes aus.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Valid:      BOOL;
  Busy:      BOOL;
  Error:      BOOL;
  ErrorID:    UDINT;
  Torque:     LREAL;
END_VAR
```

Name	Typ	Beschreibung
Valid	BOOL	Hier wird die erfolgreiche Ermittlung des Istwertes signalisiert.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.

Name	Typ	Beschreibung
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Torque	LREAL	Die Istkraft bzw. der Istdruck.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Enable** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Befindet sich die Achse in einem gestörten Zustand und ist die Ursache ein Encoder-Problem wird mit **Error** und **ErrorID:=Errorcode** des Encoders reagiert.

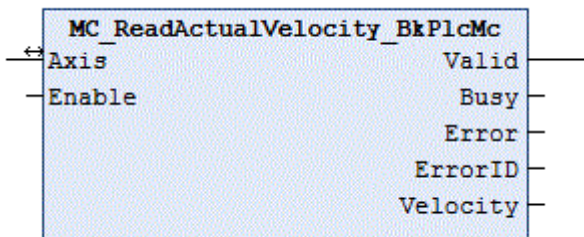
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Istkraft bzw. der Istdruck ermittelt und **Valid** gemeldet.

Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

**4.1.4 MC\_ReadActualVelocity\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein ermittelt die aktuelle Geschwindigkeit einer Achse.

**Eingänge**

```
VAR_INPUT
  Enable:      BOOL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst eine Aktualisierung des Geschwindigkeitswertes aus.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Valid:      BOOL;
  Busy:      BOOL;
  Error:      BOOL;
```

```
ErrorID:    UDINT;
Velocity:  LREAL;
END_VAR
```

Name	Typ	Beschreibung
Valid	BOOL	Hier wird die erfolgreiche Ermittlung der Geschwindigkeit signalisiert.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Velocity	LREAL	[mm/s] Die Istgeschwindigkeit.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Enable** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Befindet sich die Achse in einem gestörten Zustand und ist die Ursache ein Encoder-Problem wird mit **Error** und **ErrorID:=**Errorcode des Encoders reagiert.

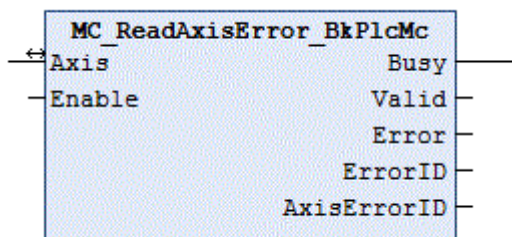
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Geschwindigkeit ermittelt und **Valid** gemeldet.

Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

**4.1.5 MC\_ReadAxisError\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein ermittelt den aktuellen Errorcode einer Achse.

**Eingänge**

```
VAR_INPUT
  Enable:    BOOL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein TRUE an diesem Eingang löst eine Aktualisierung des Errorcodes aus.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> <a href="#">▶ 84</a> zu übergeben.



**Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  AxisErrorID:  UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Ermittlung der Istposition signalisiert.
Error	BOOL	Hier wird TRUE signalisiert, wenn der Baustein die gewünschte Funktion nicht ausführen konnte.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt, wenn der Baustein die gewünschte Funktion nicht ausführen konnte.
AxisErrorID	UDINT	Hier wird der aktuelle Fehlercode [▶ 321] der Achse bereitgestellt.

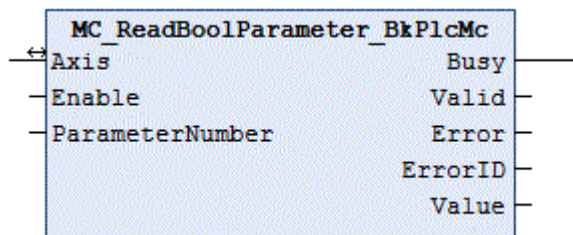
**Verhalten des Bausteins**

Wenn **Enable** TRUE ist untersucht der Baustein das übergebene Achsinterface. Der aktuelle Fehlercode wird als **AxisErrorID** gemeldet. Ist **Enable** FALSE löscht der Baustein alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit und keine Vorbedingung zur Durchführung seiner Aufgaben. Die Ausgänge Error und Busy werden zu keinem Zeitpunkt den Wert TRUE annehmen und sind nur aus Kompatibilitätsgründen vorhanden.

**4.1.6 MC\_ReadBoolParameter\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein liest die boolschen Parameter einer Achse aus. Für nicht boolsche Parameter steht der Baustein MC\_ReadParameter\_BkPlcMc [▶ 35] zur Verfügung.

**Eingänge**

```
VAR_INPUT
  Enable:          BOOL;
  ParameterNumber: INT;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst einen Lesevorgang aus.
ParameterNumber	INT	Diese Kennnummer legt den auszulesenden Parameter fest. Es sollten nur benannte Konstanten aus <u>E_TcMCPParameter</u> [▶ 97] verwendet werden.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  Value:         BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Value	BOOL	Hier wird der Wert des Parameters zur Verfügung gestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Enable** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn an **ParameterNumber** ein nicht unterstützter Wert angelegt wurde wird mit **Error** und **ErrorID:=dwTcHydErrCdNotSupport** reagiert.

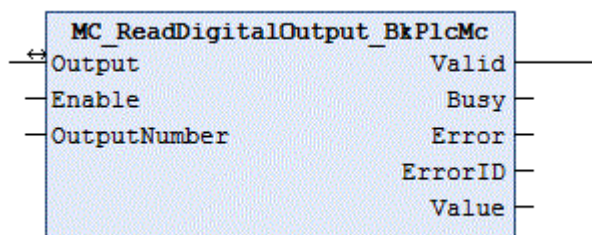
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten steht an **Value** der gewünschte Parameterwert zur Verfügung und **Done** wird gemeldet.

Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

**4.1.7 MC\_ReadDigitalOutput\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein ermittelt den aktuellen Zustand eines digitalen Ausgangs eines Nockenschaltwerks.

**Eingänge**

```
VAR_INPUT
  Enable:        BOOL;
  OutputNumber:  INT;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst eine Aktualisierung des Zustands aus.

Name	Typ	Beschreibung
OutputNumber	INT	Die Nummer des zu ermittelnden Ausgangs.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Output:      OUTPUT_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Output	OUTPUT_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ OUTPUT_REF_BkPlcMc [► 117] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Done:      BOOL;
  Busy:      BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
  Value:     BOOL;
END_VAR
```

Name	Typ	Beschreibung
Valid	BOOL	Hier wird die erfolgreiche Ermittlung des Zustands signalisiert.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Value	BOOL	Der Zustand des digitalen Ausgangs.

**Verhalten des Bausteins**

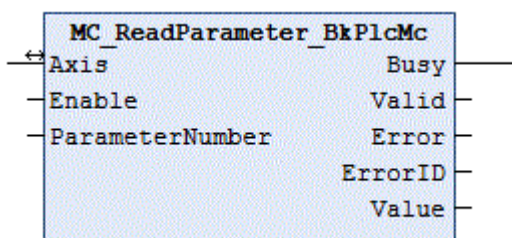
Bei einem TRUE an **Enable** untersucht der Baustein die übergebenen Parameter. Dabei kann ein Problem erkannt und gemeldet werden:

- Wenn der Wert von **OutputNumber** nicht im zulässigen Bereich [0..31] liegt wird mit **Error** und **ErrorID:=dwTchHydErrCdIllegalOutputNumber** reagiert.

Wenn diese Überprüfungen ohne Probleme durchgeführt werden konnten wird der Zustand des digitalen Ausgangs ermittelt und **Valid** gemeldet.

Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.

**4.1.8 MC\_ReadParameter\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein liest die nicht boolschen Parameter einer Achse aus. Für boolsche Parameter steht der Baustein MC\_ReadBoolParameter\_BkPlcMc [► 33] zur Verfügung.

## Eingänge

```
VAR_INPUT
  Enable:          BOOL;
  ParameterNumber: INT;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst einen Lesevorgang aus.
ParameterNumber	INT	Diese Kennnummer legt den auszulesenden Parameter fest. Es sollten nur benannte Konstanten aus <a href="#">E_TcMCPParameter [► 97]</a> verwendet werden.

## Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [► 84]</a> zu übergeben.

## Ausgänge

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  Value:         LREAL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Value	LREAL	Hier wird der Wert des Parameters zur Verfügung gestellt.

## Verhalten des Bausteins

Auf eine steigende Flanke an **Enable** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn an **ParameterNumber** ein nicht unterstützter Wert angelegt wurde wird mit **Error** und **ErrorID:=dwTcHydErrCdNotSupport** reagiert.

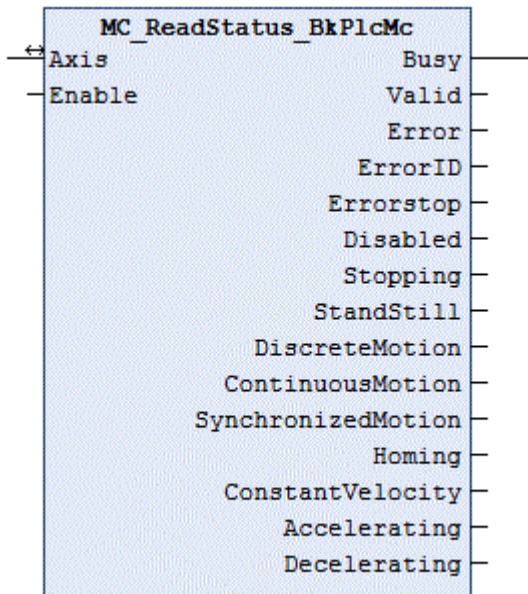
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten steht an **Value** der gewünschte Parameterwert zur Verfügung und **Done** wird gemeldet.

Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

### 4.1.9 MC\_ReadStatus\_BkPlcMc (ab V3.0)



Der Funktionsbaustein ermittelt den aktuellen Zustand einer Achse.

#### Eingänge

```
VAR_INPUT
  Enable:          BOOL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein TRUE Zustand an diesem Eingang löst eine Aktualisierung des Bausteins aus.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:           BOOL;
  Done:           BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
  Errorstop:      BOOL;
  Disabled:       BOOL;
  Stopping:       BOOL;
  StandStill:     BOOL;
  DiscreteMotion: BOOL;
  ContinousMotion: BOOL;
  SynchronizedMotion: BOOL;
  Homing:         BOOL;
  ConstantVelocity: BOOL;
  Accelerating:   BOOL;
  Decelerating:   BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Ermittlung der Istposition signalisiert.
Error	BOOL	Dieser Ausgang meldet Probleme bei der Ausführung der Funktion des Bausteins.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt, wenn der Baustein die gewünschte Funktion nicht ausführen konnte.
Errorstop	BOOL	Dieses Signal zeigt an, dass die Achse eines Fehlers in einen nicht betriebsbereiten Zustand versetzt wurde. Dieser Zustand ist nur durch die Aktivierung eines <a href="#">MC_Reset_BkPlcMc [▶ 39]</a> oder <a href="#">MC_ResetAndStop_BkPlcMc [▶ 40]</a> Bausteins aufhebbar.
Disabled	BOOL	Dieses Signal zeigt an, ob die Achse durch ihren <a href="#">MC_Power_BkPlcMc [▶ 27]</a> Baustein enabled oder disabled wurde.
Stopping	BOOL	Dieses Signal zeigt an, dass eine aktive Bewegung der Achse durch einen <a href="#">MC_Stop_BkPlcMc [▶ 81]</a> oder <a href="#">MC_ResetAndStop_BkPlcMc [▶ 40]</a> Baustein beendet wird. Sobald die Achse den Stillstand erreicht hat, wird dieses Signal gelöscht.
StandStill	BOOL	Dieses Signal zeigt an, dass die Achse weder gestört noch aktiv ist.
DiscreteMotion	BOOL	Dieses Signal zeigt an, dass die Achse eine eigenständige (nicht durch eine Kopplung zustande gekommene) Bewegung mit definiertem Ziel ausführt.
ContinuousMotion	BOOL	Dieses Signal zeigt an, dass die Achse eine eigenständige (nicht durch eine Kopplung zustande gekommene) Bewegung mit definierter Geschwindigkeit, aber ohne festgelegtes Ziel ausführt.
SynchronizedMotion	BOOL	Dieses Signal zeigt an, dass die Achse durch eine Getriebekopplung kontrolliert wird.
Homing	BOOL	Dieses Signal zeigt an, dass die Achse eine Referenzfahrt ausführt.
ConstantVelocity	BOOL	Dieses Signal zeigt an, dass die Achse mit konstanter Geschwindigkeit bewegt wird.
Accelerating	BOOL	Dieses Signal zeigt an, dass die Achse ihre Geschwindigkeit einem vorgesehenen Wert annähert.  <b>Dies bedeutet nicht immer eine Betragserhöhung der Geschwindigkeit: beim Starten einer bereits fahrenden Achse kann es vorkommen, dass die Achse entgegen dem aktuellen Geschwindigkeitsvorzeichen beschleunigt, um eine vorgesehene Geschwindigkeit in Gegenrichtung aufzubauen. Dies ist aus Sicht der ursprünglichen Bewegung eine Verzögerung, aus Sicht der aktuellen (neuen) Bewegung jedoch eine Beschleunigung.</b>
Decelerating	BOOL	Dieses Signal zeigt an, dass die Achse ihre Geschwindigkeit reduziert, um eine Bewegung mit einer geringeren als der aktuellen Geschwindigkeit fortzusetzen oder zu beenden.

### Verhalten des Bausteins

Wenn an **Enable** TRUE anliegt untersucht der Baustein das übergebene Achsinterface und dekodiert die internen Zustandsinformationen. Ein FALSE Zustand an **Enable** löscht alle anstehenden Ausgangssignale.

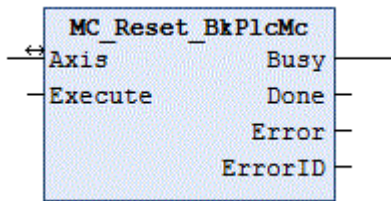
**i** Dieser Baustein benötigt keine Zeit und keine Vorbedingung zur Durchführung seiner Aufgaben. Die Ausgänge Error und Busy werden zu keinem Zeitpunkt den Wert TRUE annehmen und sind nur aus Kompatibilitätsgründen vorhanden.

#### **i** Ausgänge beachten

Die Ausgänge **Error** und **ErrorID** geben den Zustand des Bausteins und **nicht** den der Achse wieder.

Um den aktuellen Fehlercode der Achse auszulesen ist ein `MC_ReadAxisError_BkPlcMc()` [▶ 32] Baustein zu verwenden.

### 4.1.10 MC\_Reset\_BkPlcMc (ab V3.0)



Der Funktionsbaustein beseitigt einen Störzustand und versetzt die Achse in einen betriebsbereiten Zustand.

#### Eingänge

```
VAR_INPUT
    Execute:    BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang löst einen Achsreset aus.

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [▶ 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    Busy:      BOOL;
    Done:      BOOL;
    Error:      BOOL;
    ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung des Achsreset signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

#### Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin wird der Achsreset durchgeführt. Dadurch wird die Achse soweit möglich in einem betriebsbereiten Zustand versetzt und **Done** gemeldet. Ist dies nicht möglich wird mit **Error** und **ErrorID:=Achs-ErrorCode** reagiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale.



Bei einigen Antriebstypen ist zur Behebung bestimmter Fehler ein Signalaustausch mit einem externen Gerät erforderlich. Während der dafür benötigten Zeit kann der Baustein kein endgültiges Ergebnis (Done oder Error) melden, sondern signalisiert mit Busy die andauernde Bearbeitung der Funktion.

### 4.1.11 MC\_ResetAndStop\_BkPlcMc (ab V3.0)



Der Funktionsbaustein versetzt eine gestörte Achse in einen betriebsbereiten Zustand. Wenn die Achse einen Fahrauftrag abarbeitet wird dieser abgebrochen und die dazu nötige Stopp-Operation wird überwacht.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  Deceleration: LREAL; (ab/from V3.0.5)
  Jerk:         LREAL; (ab/from V3.0.5)
  RampTime:    LREAL; (ab/from V3.0.5)
  BufferMode:   MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc; (ab/from V3.0.8)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang löst einen Achsreset und eine Stopp-Operation aus.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die anzuwendende Verzögerung.
Jerk	LREAL	[mm/s <sup>3</sup> ] Der anzuwendende Ruck.
RampTime	LREAL	[s] Die geforderte Anhaltezeit.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung des Achsreset signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.



**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Ist ein anstehender Stöorzustand der Achse durch die Reset-Operation nicht erfolgreich zu beheben wird mit **Error** und **ErrorID:=Achs-ErrorCode** reagiert.
- Wird die Achse im Verlauf einer eventuell erforderlichen Stopp-Operation in einen Stöorzustand versetzt wird mit **Error** und **ErrorID:=Achs-ErrorCode** reagiert.

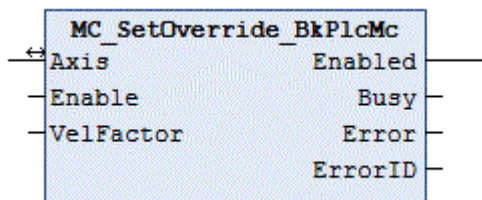
Nach erfolgreichem Abschluss beider Operationen wird **Done** gemeldet. Die Achse ist dann störungsfrei und im Stillstand.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale.



Wenn die Achse eine Bewegung ausführt wird sie bis zum Stillstand abgebremst. Zusätzlich ist bei einigen Antriebstypen zur Behebung bestimmter Fehler ein Signalaustausch mit einem externen Gerät erforderlich. Während der dafür benötigten Zeit kann der Baustein kein endgültiges Ergebnis (Done oder Error) melden, sondern signalisiert mit Busy die andauernde Bearbeitung der Funktion.

**4.1.12 MC\_SetOverride\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein setzt den Override einer Achse.



Dieser Baustein wirkt nur bei Verwendung des Profiltyps iTcMc\_ProfileCtrlBased.

**Eingänge**

```
VAR_INPUT
  Enable:      BOOL;
  VelFactor:   LREAL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein aktiver Zustand an diesem Eingang setzt den Override der Achse.
VelFactor	LREAL	[1] Der neue Override der Achse.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Enabled:  BOOL;
  Busy:    BOOL;
```

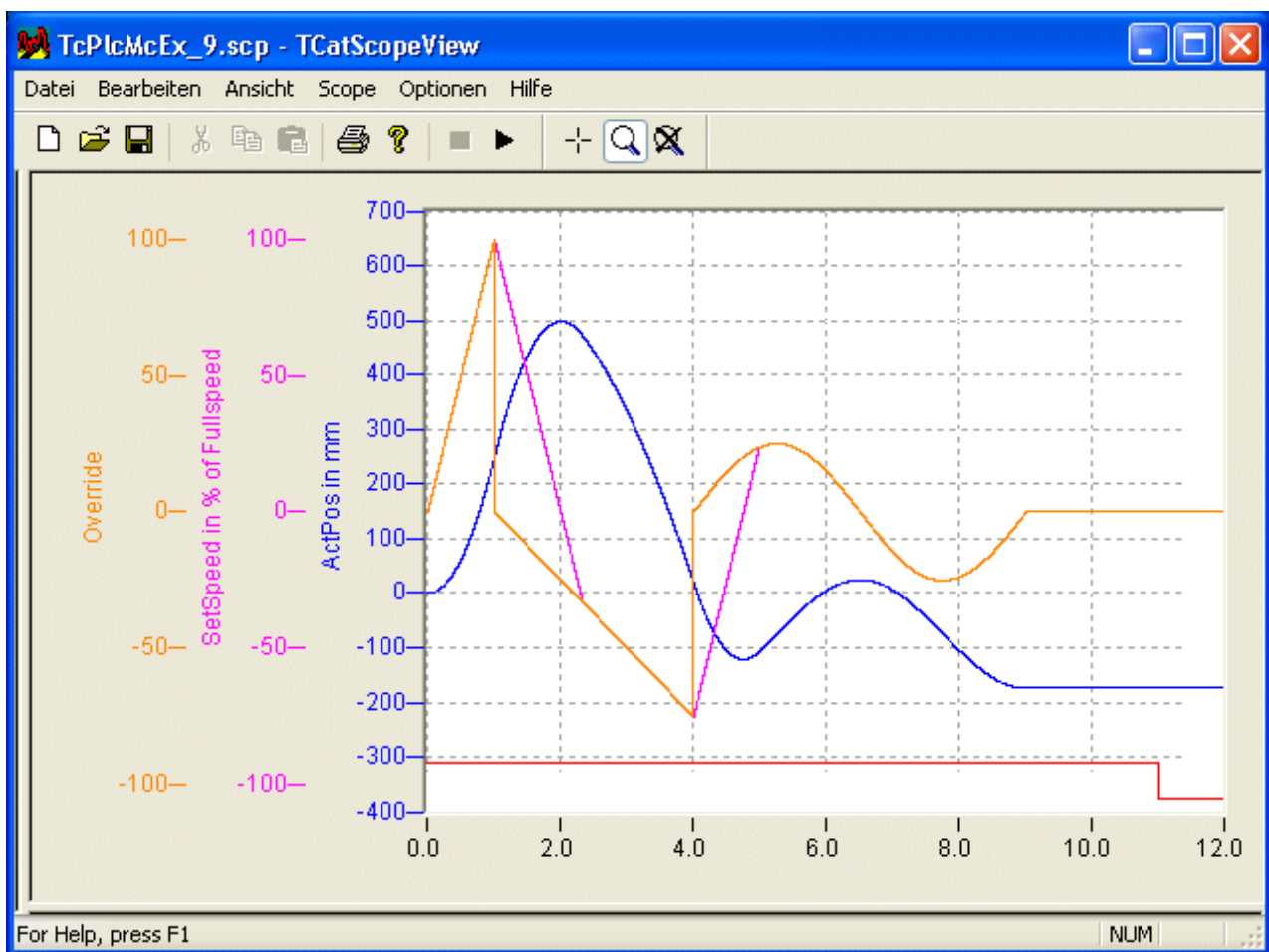
```
Error:      BOOL;
ErrorID:    UDINT;
END_VAR
```

Name	Typ	Beschreibung
Enabled	BOOL	Hier wird der aktive Zustand des Bausteins signalisiert.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

### Verhalten des Bausteins

Bei einem aktiven Zustand an **Enable** wird der als **VelOverride** übergebene Wert auf den Bereich 0.0 bis 1.0 begrenzt und in **Axis.pStAxParams^.fOverride** eingetragen. **Enabled** wird auf TRUE gesetzt.

Eine fallende Flanke an **Enable** löscht alle Ausgänge.

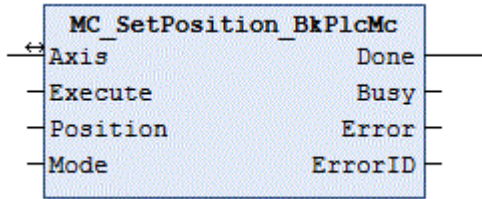


Alle durch eine Override-Änderung hervorgerufenen Geschwindigkeitsänderungen werden entsprechend der maximal zulässigen Beschleunigungen und Verzögerungen begrenzt.



Um ein reproduzierbares Verhalten bei der Zielfahrt zu garantieren wird durch den Override die Fahrgeschwindigkeit nur bis auf `pStAxParams.fCreepSpeed` reduziert. Dadurch ist es nicht möglich, die Achse durch einen Override von 0.0 in der Bewegung zu stoppen.

### 4.1.13 MC\_SetPosition\_BkPlcMc (ab V3.0)



Der Funktionsbaustein setzt die Istposition einer Achse.

#### Eingänge

```

VAR_INPUT
  Execute:    BOOL;
  Position:   LREAL;
  Mode:       BOOL;
END_VAR
  
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang setzt die Istposition der Achse.
Position	LREAL	[mm] Die neue Istposition der Achse.
Mode	BOOL	Dieser Parameter legt den Arbeitsmodus fest. Bei <b>Mode</b> =TRUE wird die Istposition um <b>Position</b> geändert, bei <b>Mode</b> =FALSE wird die Istposition auf <b>Position</b> gesetzt.

#### Ein-/Ausgänge

```

VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
  
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```

VAR_OUTPUT
  Done:      BOOL;
  Busy:      BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
END_VAR
  
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung des Befehls signalisiert.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

#### Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

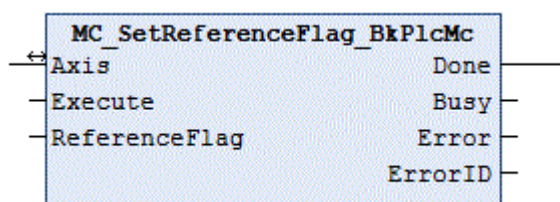
- In Abhängigkeit vom in **Axis.pStAxParams^.nEnc\_Type** festgelegten Encodertyp wird entweder **ST\_TcHydAxRtData.fEnc\_RefShift** oder **ST\_TcHydAxParam.fEnc\_ZeroShift** so aktualisiert, dass die Istposition der Achse den geforderten Wert annimmt. Ist der Encodertyp nicht bekannt oder der Encoder lässt ein Setzen des Istwerts nicht zu wird mit **Error** und **ErrorID:=dwTcHydErrCdEncType** reagiert.

- Wird ST\_TcHydAxParam.fEnc\_ZeroShift dabei erkennbar verändert wird Axis\_Ref\_BkPlcMc [▶ 84].ST\_TcHydAxRtData [▶ 131].bParamsUnsave gesetzt.

**i** Durch diesen Baustein können die Istposition und/oder die Zielposition einer aktuell abgearbeiteten Bewegung hinter einen aktiven Software-Endschalter verschoben werden. Dies wird nicht durch den Baustein überwacht.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten werden alle ebenfalls betroffenen Elemente in ST\_TcHydAxRtData automatisch aktualisiert. Dadurch ist dieser Baustein auch bei Achsen aktivierbar, die eine aktive Bewegung ausführen. Die erfolgreiche Ausführung der Funktion wird mir **Done** signalisiert. Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale.

### 4.1.14 MC\_SetReferenceFlag\_BkPlcMc (ab V3.0)



(Funktion ist nicht durch PLCopen definiert) Der Funktionsbaustein definiert das Referenzier-Flag einer Achse.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  ReferenceFlag:  BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang setzt das Referenzier-Flag der Achse.
ReferenceFlag	BOOL	Der neue Zustand des Referenzier-Flags der Achse.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [▶ 84] zu übergeben.

#### Ausgänge

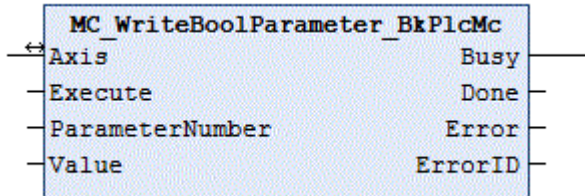
```
VAR_OUTPUT
  Done:      BOOL;
  Busy:      BOOL;
  Error:      BOOL;
  ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung des Befehls signalisiert.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin wird das Referenzier-Flag in ST\_TcHydAxRtData.nStateDWord [▶ 320] aktualisiert. Dazu wird abhängig von **ReferenceFlag** das entsprechende Bit mit `dwTcHydNsDwReferenced` gelöscht oder gesetzt. Die erfolgreiche Ausführung der Funktion wird mir **Done** signalisiert. Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale.

**4.1.15 MC\_WriteBoolParameter\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein schreibt die boolschen Parameter einer Achse aus. Für nicht boolsche Parameter steht der Baustein MC\_WriteParameter\_BkPlcMc [▶ 47] zur Verfügung.

**Eingänge**

```

VAR_INPUT
  Enable:          BOOL;
  ParameterNumber: INT;
  Value:           BOOL;
END_VAR
  
```

Name	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst einen Schreibvorgang aus.
ParameterNumber	INT	Diese Kennnummer legt den auszulesenden Parameter fest. Es sollten nur benannte Konstanten aus <u>E_TcMCPParameter</u> [▶ 97] verwendet werden.
Value	BOOL	Hier ist der Wert des Parameters bereit zu stellen.

**Ein-/Ausgänge**

```

VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
  
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc zu übergeben.

**Ausgänge**

```

VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
END_VAR
  
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung des Schreibvorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Enable** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn an **ParameterNumber** ein nicht unterstützter Wert angelegt wurde wird mit **Error** und **ErrorID:=dwTcHydErrCdNotSupport** reagiert.

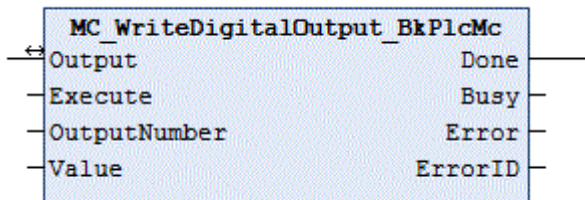
Wenn diese Überprüfungen ohne Probleme durchgeführt werden konnten wird **Value** in den gewünschten Parameterwert eingetragen und **Done** wird gemeldet. Wird der Parameter dabei verändert wird Axis\_Ref\_BkPlcMc [▶ 84].ST\_TcHydAxRtData [▶ 131].bParamsUnsave gesetzt.

Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

**4.1.16 MC\_WriteDigitalOutput\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein definiert den Zustand eines digitalen Ausgangs eines Nockenschaltwerks.

**Eingänge**

```
VAR_INPUT
    Execute:      BOOL;
    OutputNumber: INT;
    Value:        BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang löst eine Aktualisierung des Zustands aus.
OutputNumber	INT	Die Nummer des zu ermittelnden Ausgangs.
Value	BOOL	Der Zustand des digitalen Ausgangs.

**Ein-/Ausgänge**

```
VAR_INOUT
    Output:      OUTPUT_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Output	OUTPUT_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ OUTPUT_REF_BkPlcMc zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
    Done:      BOOL;
    Busy:      BOOL;
    Error:     BOOL;
    ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Ermittlung des Zustands signalisiert.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

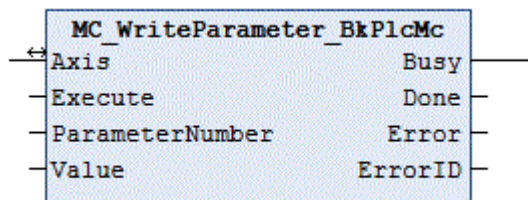
Bei einer steigenden Flanke an **Execute** untersucht der Baustein die übergebenen Parameter. Dabei kann ein Problem erkannt und gemeldet werden:

- Wenn der Wert von **OutputNumber** nicht im zulässigen Bereich [0..31] liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdIllegalOutputNumber** reagiert.

Wenn diese Überprüfungen ohne Probleme durchgeführt werden konnten wird der Zustand des digitalen Ausgangs entsprechend dem Wert von **Value** definiert und **Done** gemeldet.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale.

**4.1.17 MC\_WriteParameter\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein schreibt die nicht boolschen Parameter einer Achse aus. Für boolsche Parameter steht der Baustein MC\_WriteBoolParameter\_BkPlcMc [▶ 45] zur Verfügung.

**Eingänge**

```
VAR_INPUT
  Enable:          BOOL;
  ParameterNumber: INT;
  Value:           LREAL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Eine steigende Flanke an diesem Eingang löst einen Schreibvorgang aus.
ParameterNumber	INT	Diese Kennnummer legt den auszulesenden Parameter fest. Es sollten nur benannte Konstanten aus <u>E_TcMCPParameter</u> [▶ 97] verwendet werden.
Value	LREAL	Hier ist der Wert des Parameters bereit zu stellen.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [▶ 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung des Schreibvorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Enable** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn an **ParameterNumber** ein nicht unterstützter Wert angelegt wurde wird mit **Error** und **ErrorID:=dwTcHydErrCdNotSupport** reagiert.

Wenn diese Überprüfungen ohne Probleme durchgeführt werden konnten wird **Value** in den gewünschten Parameterwert eingetragen und **Done** wird gemeldet. Wird der Parameter dabei erkennbar verändert wird `Axis_Ref_BkPlcMc [▶ 84].ST TcHydAxRtData [▶ 131].bParamsUnsave` gesetzt.

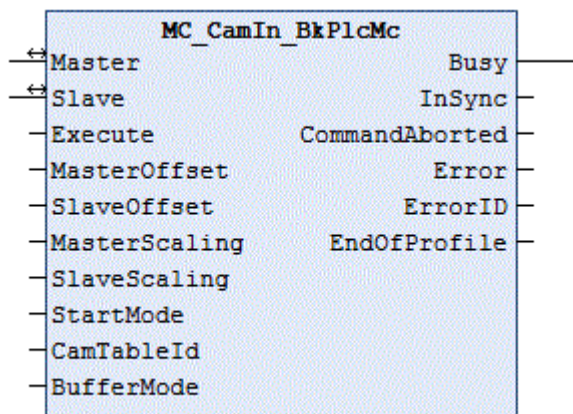
Eine fallende Flanke an **Enable** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

## 4.2 Motion

### 4.2.1 MC\_CamIn\_BkPlcMc (ab V3.0)



Der Funktionsbaustein startet und überwacht eine Kurvenscheiben-Kopplung zwischen zwei Achsen. Zum Lösen der Kopplung ist ein `MC_CamOut_BkPlcMc [▶ 50]` Baustein zu verwenden.

**Eingänge**

```
VAR_INPUT
  Execute:          BOOL;
  MasterOffset:     LREAL:=0.0;
  SlaveOffset:      LREAL:=0.0;
  MasterScaling:    LREAL:=0.0;
```



```
SlaveScaling: LREAL:=0.0;
StartMode: MC_StartMode_BkPlcMc:=MC_StartMode_Absolute;
CamTableId: MC_CAM_ID_BkPlcMc;
BufferMode: MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc; (ab/from V3.0.8)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Kopplung.
MasterOffset	LREAL	[mm, 1] Dieser Wert wird mit der Istposition des Masters verrechnet, bevor der sich ergebende Wert in der Master-Spalte der Tabelle gesucht wird.
SlaveOffset	LREAL	[mm, 1] Dieser Wert wird mit der Slave-Position aus der Tabelle verrechnet.
MasterScaling	LREAL	[mm, 1] Dieser Wert wird mit der Istposition des Masters verrechnet, bevor der sich ergebende Wert in der Master-Spalte der Tabelle gesucht wird.
SlaveScaling	LREAL	[mm, 1] Dieser Wert wird mit der Slave-Position aus der Tabelle verrechnet.
StartMode	MC_StartMode_BkPlcMc	Ein Wert aus <code>MC_StartMode_BkPlcMc</code> [► 116], der das Verhalten der Slave-Achse beim Aktivieren der Kopplung festlegt.
CamTableId	MC_CAM_ID_BkPlcMc	Hier ist eine Variable des Typs <code>MC_CAM_ID_BkPlcMc</code> [► 113] zu übergeben, die von einem Baustein des Typs <code>MC_CamTableSelect_BkPlcMc</code> [► 52] initialisiert wurde.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten <code>Aborting_BkPlcMc</code> belegt werden. (ab V3.0.8)

 **Ein-/Ausgänge**

```
VAR_INOUT
Master: Axis_Ref_BkPlcMc;
Slave: Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Master	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> zu übergeben.
Slave	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
Busy: BOOL;
InSync: BOOL;
CommandAborted: BOOL;
Error: BOOL;
ErrorID: UDINT;
EndOfProfile: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
InSync	BOOL	Hier wird das erstmalige erfolgreiche Aufsynchronisieren der Achsen signalisiert. Das Signal bleibt anschließend auch dann anstehen, wenn die Synchronisierung zu einem späteren Zeitpunkt zeitweise oder bleibend aussetzt.
CommandAborted	BOOL	Hier wird ein Abbruch der Kopplung signalisiert.

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
EndOfProfile	BOOL	Hier wird signalisiert, ob der Master das Ende des definierten Bereichs erreicht hat.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **CamTableId.bValidated** nicht durch einen Baustein vom Typ MC\_CamTableSelect\_BkPlcMc gesetzt wurde wird mit **Error** und **ErrorID:=dwTcHydErrCdTblNoInit** reagiert.
- Wenn sich entweder Master oder Slave nicht im Ruhezustand befinden wird mit **Error** und **ErrorID:=dwTcHydErrCdNotStartable** reagiert.
- Wenn als **StartMode** der Wert MC\_StartMode\_Rampln vorgegeben wird reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdNotSupport** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Kopplung initiiert. In Abhängigkeit von **StartMode** wird die Bezugsposition für **Slave** entweder auf den Wert 0.0 oder mit der aktuellen Istposition von **Slave** festgelegt. Die Achse befindet sich ab jetzt im Zustand McState\_Synchronizedmotion [▶ 96] und der Baustein beginnt mit der Berechnung und Überwachung der Kopplung.

Sollposition und Sollgeschwindigkeit von **Slave** werden in Abhängigkeit von Istposition und Sollgeschwindigkeit des Masters und der Tabelle berechnet.

Wenn bei aktiver Kopplung die Slaveachse erstmalig die durch die Kopplung geforderte Geschwindigkeit erreicht, wird dies am Ausgang InGear signalisiert. Da die Kopplung derzeit nur im Stillstand aktiviert werden kann ist dies unmittelbar der Fall. Sollte während aktiver Kopplung die Slaveachse aus einem beliebigen Grund den Vorgaben nicht folgen können bleibt InGear unverändert.

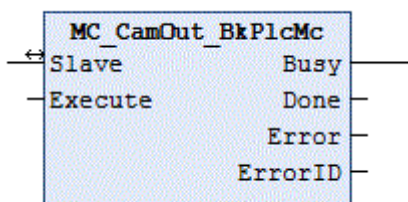
Tritt bei aktiver Kopplung im Bewegungsgenerator ein Fehlercode auf wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.

Eine fallende Flanke an **Execute** beendet weder die Berechnung noch die Überwachung der Kopplung. Dies kann nur durch die Kopplung durch einen MC\_CamOut\_BkPlcMc Baustein oder das Auftreten eines Fehlers bewirkt werden. Erst dann werden alle anstehenden Ausgangssignale gelöscht.

**i** Dieser Baustein übernimmt zeitweise die Aufgabe der Sollwertgenerierung. Um dies zu signalisieren wird Busy nicht nur bis zum Übergang in die Synchronität TRUE, sondern bleibt bis zum Auflösen der Kopplung anstehen.

**i** **Bausteinaufruf**  
Es ist zwingend erforderlich, diesen Baustein zyklisch aufzurufen wenn **Busy** auf TRUE steht. Anschließend ist der Baustein noch mindestens einmal mit **Execute:=FALSE** aufzurufen.

**4.2.2 MC\_CamOut\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein löst eine durch einen MC\_CamIn\_BkPlcMc [▶ 48] Baustein gestartete Kurvenscheiben-Kopplung zwischen zwei Achsen.

 **Eingänge**

```
VAR_INPUT
  Execute:          BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Kopplung.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Slave:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Slave	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:           BOOL;
  Done:           BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

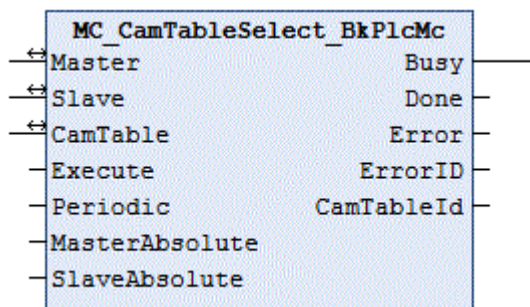
- Wenn der Pointer pStAxParams in Axis\_Ref\_BkPlcMc [[▶ 84](#)] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.
- Wenn der Pointer pStAxRtData in Axis\_Ref\_BkPlcMc [[▶ 84](#)] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrMcPlc** reagiert.
- Wenn sich die Achse nicht gekoppelt ist reagiert der Baustein ohne weitere Prüfungen oder Aktivitäten mit **Done**.
- Ist die aktuelle Sollgeschwindigkeit der Achse kleiner als die von pStAxParams.fCreepSpeed festgelegte Geschwindigkeit geht die Achse unmittelbar in McState\_Standstill über und baut die Restgeschwindigkeit ab. Es wird **Done** signalisiert und alle weiteren Überprüfungen oder Aktivitäten werden unterlassen.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten und nicht bereits aus einem der genannten Gründe **Done** signalisiert wird erfolgt eine Umwandlung der von der Kurvenscheibenkopplung kontrollierten Bewegung in eine vom Master unabhängige endlose Bewegung mit gleicher Geschwindigkeit und Richtung. Wenn diese Umwandlung erfolgreich ausgeführt wurde wird **Done** signalisiert, andernfalls wird mit **Error** und **ErrorID:=Fehlercode** reagiert.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

### 4.2.3 MC\_CamTableSelect\_BkPlcMc (ab V3.0)



Der Funktionsbaustein initialisiert eine Variable vom Typ MC\_CAM\_ID\_BkPlcMc [► 113] und bereitet dadurch eine Kurvenscheibe für die Kopplung von zwei Achsen vor.

#### Eingänge

```
VAR_INPUT
    Execute:          BOOL;
    Periodic:         BOOL;
    MasterAbsolute:   BOOL;
    SlaveAbsolute:   BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet das Kommando.
Periodic	BOOL	Nicht unterstützt: Hier ist derzeit FALSE zu übergeben.
MasterAbsolute	BOOL	Nicht unterstützt: Hier ist derzeit TRUE zu übergeben.
SlaveAbsolute	BOOL	Nicht unterstützt: Hier ist derzeit TRUE zu übergeben.

#### Ein-/Ausgänge

```
VAR_INOUT
    Master:           Axis_Ref_BkPlcMc;
    Slave:           Axis_Ref_BkPlcMc;
    CamTable:       MC_CAM_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Master	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [► 84] zu übergeben.
Slave	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [► 84] zu übergeben.
CamTable	MC_CAM_REF_BkPlcMc	Hier ist eine Variable des Typs <u>MC_CAM_REF_BkPlcMc</u> [► 113] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    Busy:           BOOL;
    Done:          BOOL;
    Error:         BOOL;
    ErrorID:       UDINT;
    CamTableId:   MC_CAM_ID_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Initialisierung von CamTableId signalisiert.

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
CamTableId	MC_CAM_ID_BkPlcMc	Hier wird eine Variable des Typs <code>MC_CAM_ID_BkPlcMc</code> [ <a href="#">▶ 113</a> ] zurückgegeben, die an einen Baustein des Typs <code>MC_CamIn_BkPlcMc</code> [ <a href="#">▶ 48</a> ] weitergegeben werden kann.

### Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **CamTable.pTable** nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.
- Wenn **CamTable.nLastIdx** nicht grösser als **CamTable.nFirstIdx** ist wird mit **Error** und **ErrorID:=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **CamTable.nFirstIdx** und **CamTable.nLastIdx** eine Tabelle mit mehr als 100 Zeilen definieren wird mit **Error** und **ErrorID:=dwTcHydErrCdTblLineCount** reagiert.
- Wenn **MasterAbsolute** oder **SlaveAbsolute** nicht gesetzt oder **Periodic** gesetzt ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotSupport** reagiert.

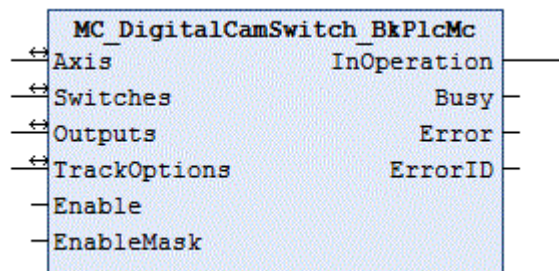
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird **CamTableId** initialisiert. Dazu werden die Daten aus **CamTable** und die Eingangsdaten des Bausteins übernommen. **CamTableId** wird als gültig und verändert markiert. Mit **Done** wird die Abarbeitung des Kommandos gemeldet.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale.



Dieser Baustein benötigt keine Zeit zur Durchführung seiner Aufgaben. Der Ausgang Busy wird zu keinem Zeitpunkt den Wert TRUE annehmen und ist nur aus Kompatibilitätsgründen vorhanden.

## 4.2.4 MC\_DigitalCamSwitch\_BkPlcMc (ab V3.0)



Der Funktionsbaustein erzeugt in Abhängigkeit von Position, Bewegungsrichtung und Geschwindigkeit einer Achse Software-Nocken.

### Eingänge

```
VAR_INPUT
    Enable:          BOOL;
    EnableMask:     DWORD;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Dieser Eingang kontrolliert alle Aktivitäten des Bausteins.
EnableMask	DWORD	Eine Maske, deren Bits die Freigabe der Ausgänge in <b>Outputs</b> festlegen.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
  Switches:     CAMSWITCH_REF_BkPlcMc;
  Outputs:     OUTPUT_REF_BkPlcMc;
  TrackOptions: TRACK_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.
Switches	CAMSWITCH_REF_BkPlcMc	Hier ist ein Array vom Typ <u>CAMSWITCH REF BkPlcMc</u> [ <a href="#">▶ 114</a> ] zu übergeben.
Outputs	OUTPUT_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>OUTPUT_REF BkPlcMc</u> [ <a href="#">▶ 117</a> ] zu übergeben.
TrackOptions	TRACK_REF_BkPlcMc	Hier ist ein Array vom Typ <u>TRACK_REF BkPlcMc</u> [ <a href="#">▶ 116</a> ] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  InOperation:  BOOL;
  Busy:        BOOL;
  Error:       BOOL;
  ErrorID:     UDINT;
END_VAR
```

Name	Typ	Beschreibung
InOperation	BOOL	Hier wird signalisiert, ob der Baustein aktiv ist.
Busy	BOOL	Dieser Ausgang ist für die Dauer der Abarbeitung des Kommandos auf TRUE.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Gesteuert durch die Istposition einer Achse werden Nockensignale (Switches) geschaltet. Es stehen positionsgesteuerte (mit Anfangs- und Endposition) und zeitgesteuerte (mit Triggerposition und Dauer) zur Verfügung. Dabei kann die Bewegungsrichtung der Achse berücksichtigt werden.

Die Nockensignale werden Spuren (Tracks) mit parametrierbaren Eigenschaften zugeordnet. Durch Ein- und Ausschaltverzögerung kann das Zeitverhalten festgelegt werden. Durch negative Werte kann hier auch eine vorausschauende Signalgabe erreicht werden. Eine Hysterese ermöglicht die Unterdrückung von unerwünschten Signalgaben, wenn die Achse in der Nähe eines Schaltpunkts steht und die Istposition nicht völlig konstant ist.

**Beispiel**

Verwendete CAMSWITCH\_REF BkPlcMc [[▶ 114](#)]:

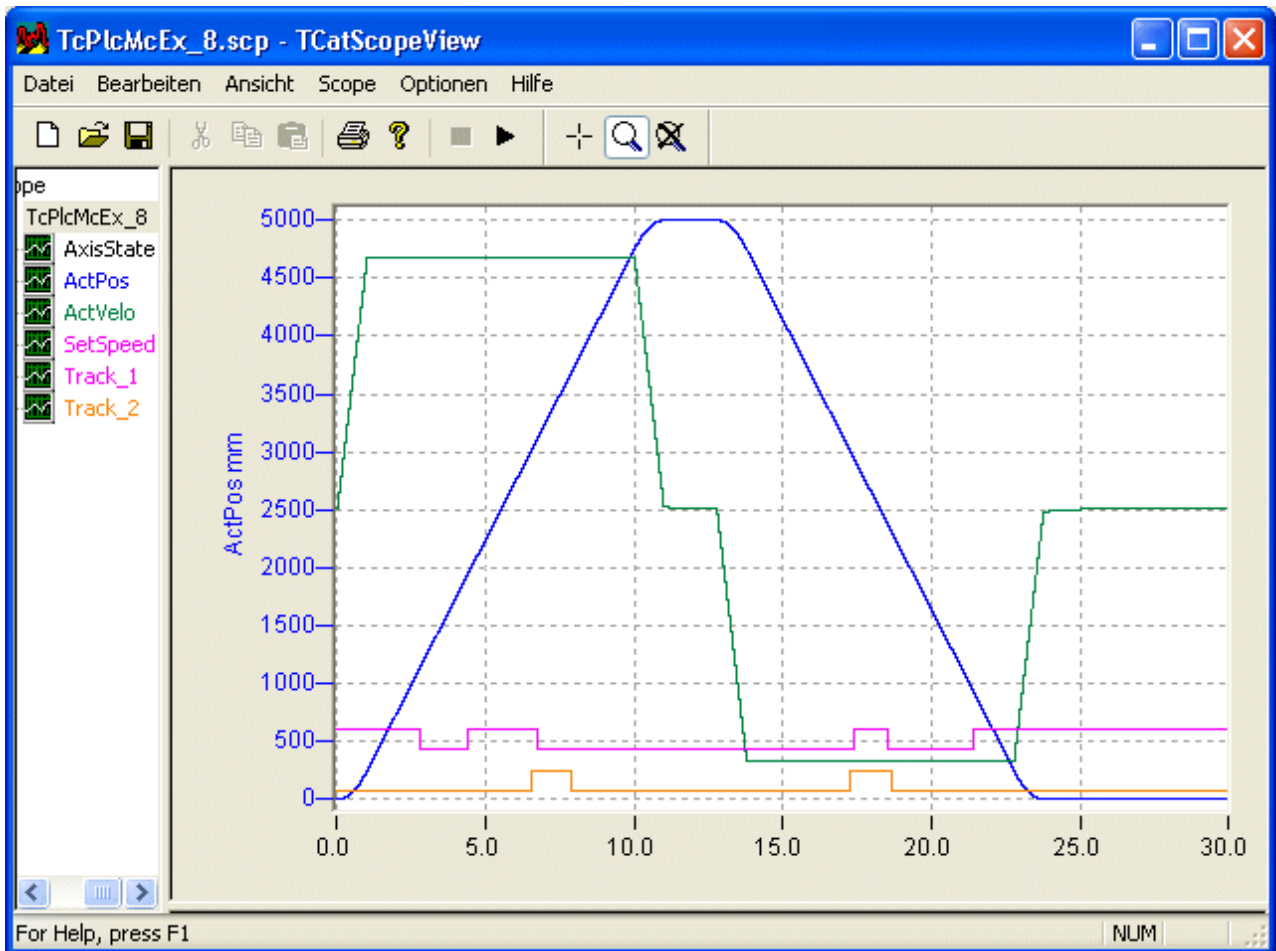
Parameter	Switch[1]	Switch[2]	Switch[3]	Switch[4]	...	Switch[n]
TrackNumber	1	1	1	2		
FirstOnPosition	2000.0	2500.0	-1000.0	3000.0		
LastOnPosition	3000.0	3000.0	1000.0			
AxisDirection	1	2	0	0		
CamSwitchMode	0	0	0	1		
Duration				1.35		

Parameter	Switch[1]	Switch[2]	Switch[3]	Switch[4]	...	Switch[n]
.....						

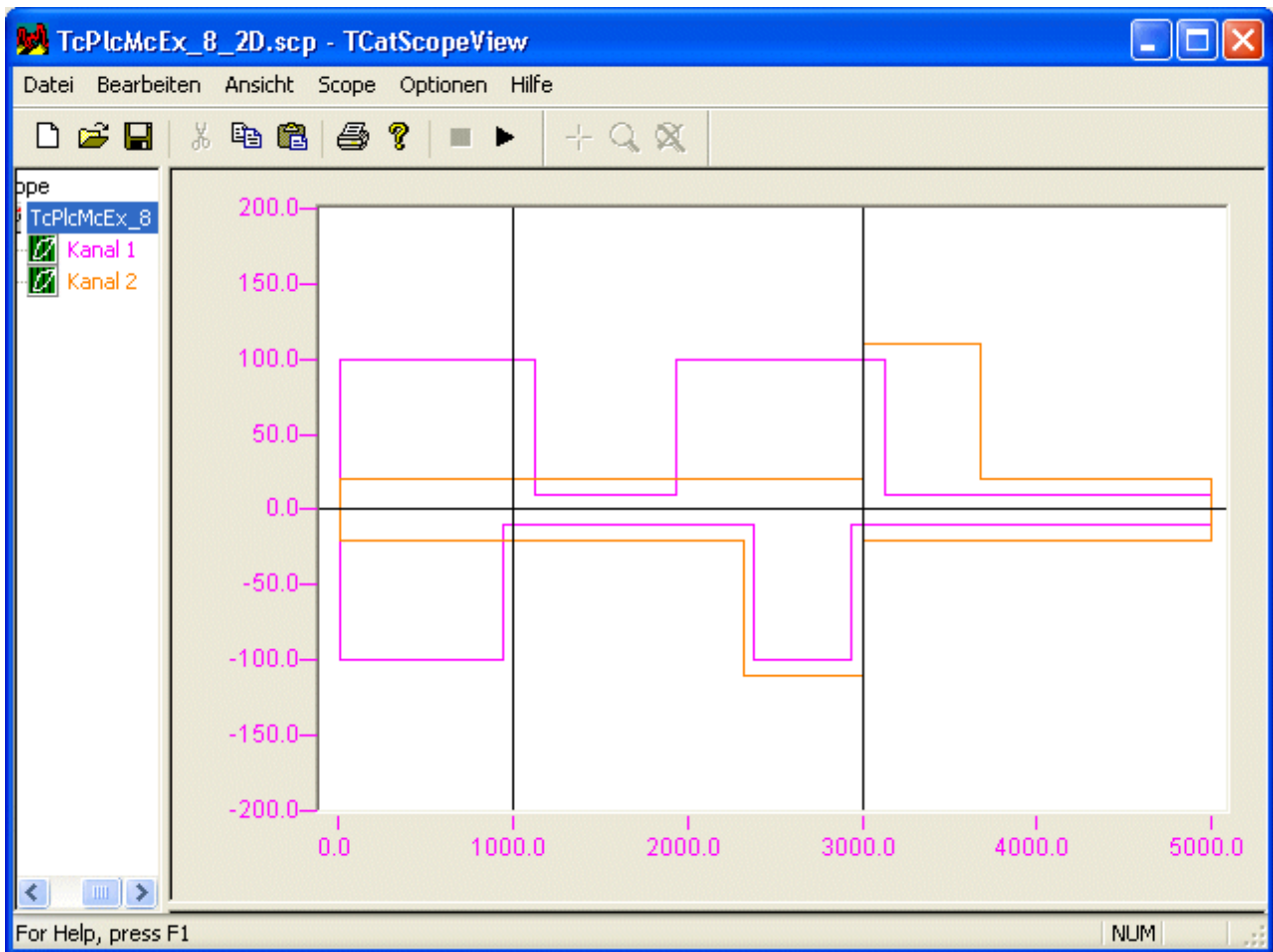
Verwendete TRACK\_REF BkPlcMc [▶ 116]:

Parameter	Track[1]	Track[2]	...	Track[n]
OnCompensation	-0.125	0.0		
OffCompensation	0.250	0.0		
Hysteresis	0.0	0.0		

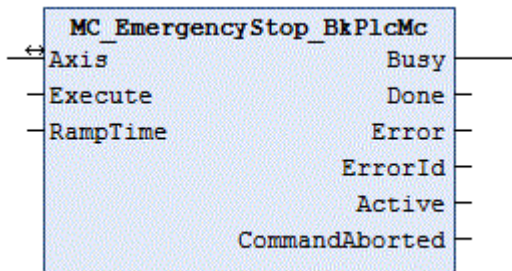
Verlauf der Signale bei einer Fahrbewegung der Achse von 0.0 nach 5000.0 und zurück:



Die nachfolgende Darstellung zeigt den Verlauf der Signale über die Position. Dabei werden die Signale bei positiver Bewegungsrichtung normal (nach oben), bei negativer Bewegungsrichtung jedoch negativ = 'nach unten' dargestellt. Die senkrechten Cursor-Linien markieren die Positionen 1000 bzw. 3000 mm.



### 4.2.5 MC\_EmergencyStop\_BkPlcMc (ab V3.0.5)



Der Funktionsbaustein bricht eine aktuell ausgeführte Bewegung einer Achse ab und überwacht die Notstopp-Operation.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  RampTime:    LREAL;  (ab/from V3.0.5)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang beendet eine Bewegung der Achse.
RampTime	LREAL	[s] Die geforderte Anhaltezeit.



 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  Active:        BOOL;
  CommandAborted: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Operation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Active	BOOL	Hier wird angezeigt, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird angezeigt, dass die Abarbeitung dieses Kommandos durch ein anderes Kommando abgebrochen wurde.

**Verhalten des Bausteins**

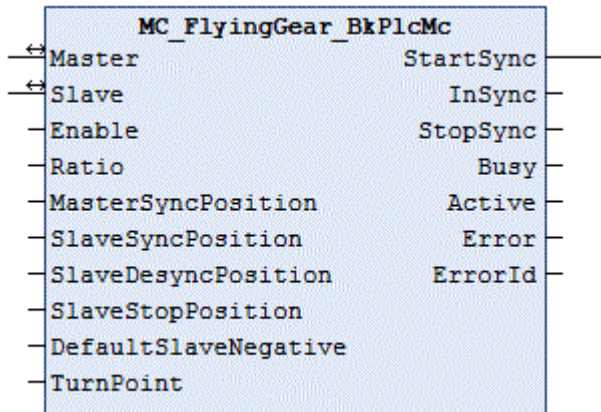
Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Ein Stopp kann nur ausgeführt werden, wenn die Achse eine aktive Bewegung ausführt. Befindet sie sich im Stillstand meldet der Baustein sofort **Done**.
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Befindet sich die Achse in einem Zustand, in dem sie durch eine Kopplung mit einer anderen Achse oder einen vergleichbaren Mechanismus kontrolliert wird reagiert sie mit **Error** und **ErrorID:=dwTcHydErrCdNotReady**.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Stopp-Operation initiiert. Dabei wird **RampTime** verwendet, um unter Berücksichtigung der Bezugsgeschwindigkeit eine Verzögerung zu berechnen. Ist ein ruckbegrenzender Stellwert-Generator ausgewählt wird **MaxJerk** verwendet. Ist für **RampTime** kein Wert vorgegeben, der erkennbar größer als 0 ist wird der Achsparameter fEmergencyRamp verwendet.

Zum Abbremsen der Achse wird intern ein MC\_Stop\_BkPlcMc [► 81] Baustein verwendet. Nachdem die Stellwertausgabe auf 0 abgebaut ist werden alle Ausgaben von Steuer- oder Regelspannungen unterdrückt, solange **Execute** auf TRUE gesetzt ist.

## 4.2.6 MC\_FlyingGear\_BkPlcMc



Der Funktionsbaustein übernimmt die Steuerung einer fliegenden Getriebekopplung.

### Eingänge

```
VAR_INPUT
  Enable:          BOOL;
  Ratio:           LREAL;
  MasterSyncPosition: LREAL;
  SlaveSyncPosition: LREAL;
  SlaveDesyncPosition: LREAL;
  SlaveStopPosition: LREAL;
  DefaultSlaveNegative: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Dieses Signal erlaubt dem Baustein, aktiv zu werden, sobald die Position des Masters die MasterSyncPosition in der vorgesehenen Richtung passiert.
Ratio	LREAL	Dieser Parameter gibt das Getriebeverhältnis zwischen Master und Slave für den vollständig gekoppelten Fall vor.
MasterSyncPosition	LREAL	Dieser Parameter gibt an, an welcher Position in der dafür vorgesehenen Richtung des Masters die Kopplung vollständig aufgebaut sein soll.
SlaveSyncPosition	LREAL	Dieser Parameter gibt an, an welcher Position in der dafür vorgesehenen Richtung des Slaves die Kopplung vollständig aufgebaut sein soll.
SlaveDesyncPosition	LREAL	Dieser Parameter gibt an, an welcher Position in der dafür vorgesehenen Richtung des Slaves der Abbau der Kopplung begonnen werden soll.
SlaveStopPosition	LREAL	Dieser Parameter gibt an, an welcher Position in der dafür vorgesehenen Richtung des Slaves die Kopplung vollständig abgebaut sein soll.
DefaultSlaveNegative	BOOL	In einigen Fällen kann der Baustein die vorgesehene Arbeitsrichtung nicht aus den übergebenen Parametern ermitteln. Dann wird eine zusätzliche Information der Applikation benötigt.

### Ein-/Ausgänge

```
VAR_IN_OUT
  Master:          AXIS_REF_BkPlcMc;
  Slave:           AXIS_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Master	AXIS_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">AxisRef BkPlcMc [► 84]</a> zu übergeben.
Slave	AXIS_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">AxisRef BkPlcMc [► 84]</a> zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  StartSync:          BOOL;
  InSync:             BOOL;
  Busy:               BOOL;
  Active:             BOOL;
  Error:              BOOL;
  ErrorId:            BOOL;
END_VAR
```

Name	Typ	Beschreibung
StartSync	BOOL	Dieses Signal ist TRUE, wenn sich der Master innerhalb der Strecke für den Aufbau der Kopplung befindet.
InSync	BOOL	Dieses Signal ist TRUE, wenn die Kopplung vollständig aktiv ist.
Busy	BOOL	Dieses Signal wird TRUE, wenn Enable TRUE, Master und Slave bereit und nicht gestört und die übergebenen Parameter geeignet sind.
Active	BOOL	Dieses Signal wird TRUE, sobald der Master die MasterSyncPosition in der dafür vorgesehenen Richtung passiert. Es wird FALSE, wenn in Gegenrichtung der Slave die SlaveStopPosition erreicht.
Error	BOOL	Dieses Signal wird TRUE, wenn bei einer steigenden Flanke an Enable ungeeignete Parameter vorliegen oder wenn bei einem TRUE an Enable der Master oder der Slave in einem Fehlerzustand sind.
ErrorId	BOOL	Eine numerisch kodierte Angabe einer Fehlerursache.

**Verhalten des Bausteins**

Der Baustein stellt eine fliegend auf- und abgebaute Kopplung zwischen einer Master- und einer Slave-Achse her. Durch die Parametrierung und die Freigabe kann das Verhalten in weiten Bereichen variiert werden. In Kombination mit Positionierbefehlen können eine Vielzahl von Bewegungsabläufen realisiert werden. Hier werden nur einige Beispiele gezeigt.

**Hinweis** Die Scopes sind mit dem Beispiel [S106\\_FlyingGear](#) erzeugt worden. Dabei entspricht die Nummer des Beispiels der gewählten nSequence.

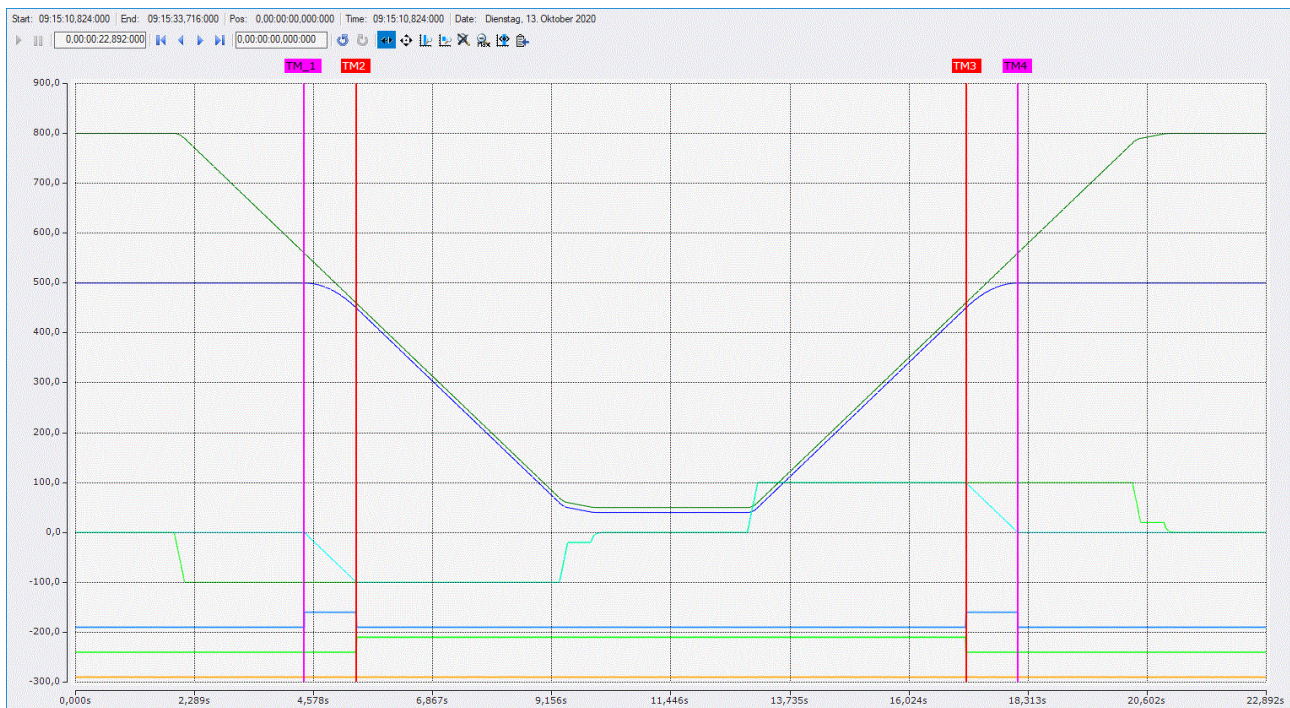
**Bei der Inbetriebnahme**

Bei der Positionserfassung des Masters sollte der Nullpunkt so verschoben werden, dass die Istposition eine nachvollziehbare Information über die Situation in der Maschine darstellt.

Für den Slave ist die Nullpunktverschiebung so zu wählen, dass die Istposition bei Berücksichtigung der Abmessungen von Material, Werkzeugen und anderen Einbauten mit der des Masters übereinstimmt, wenn der verbleibende Spalt gerade 0 geworden ist. Dabei sollte keine Kraft aufgebaut worden sein, um eine elastische Verformung zu vermeiden.

**Beispiel #1**

Hier bewegen sich Master und Slave in gleicher Richtung. Beide kehren zu ihrer Ausgangsposition zurück.



### Vorbereitung:

- Der Master wurde auf 800.0mm positioniert.
- Der Slave wurde auf 500.0mm positioniert.
- Damit ist der Ort festgelegt, an dem der Slave mit dem Aufbau der Synchronisation beginnt.
- Der endgültige Koppelfaktor (Ratio) wurde hier zum besseren Verständnis auf 1.0 eingestellt. Im praktischen Einsatz wird dieser Faktor oft etwas kleiner gewählt, um einen Übergang in eine Druck- oder Kraftregelung auszulösen.
- Der Punkt für das Erreichen der Synchronisation wurde für den Master (MasterSyncPosition) auf 460.0mm und für den Slave (SlaveSyncPosition) auf 450.0mm festgelegt.
- Die Differenz von 10.0mm zwischen Master und Slave stellt den für Material, Werkzeug und andere Einbauten benötigten Abstand dar und ist hier so gewählt, dass die Darstellung das Verhalten verdeutlicht.
- Der Weg für das Aufsynchronisieren des Slaves errechnet sich aus der Ausgangsposition des Slaves und der SlaveSyncPosition und ist hier  $500.0 - 450.0 \Rightarrow 50\text{mm}$ .
- Bei  $\text{Ratio}=1.0$  ergibt sich für den Master ein Weg für die Aufsynchronisation von  $2.0 * 50\text{mm} \Rightarrow 100\text{mm}$ . Also wird die Aufsynchronisation des Masters bei  $460.0\text{mm} + 100\text{mm} \Rightarrow 560\text{mm}$  (MasterSyncPosition plus Weg) beginnen. Wenn der Master beim Freigeben der Kopplung unterhalb dieser Position steht wird ein Fehler gemeldet.
- Der Punkt für das Verlassen der Synchronisation wurde für den Slave (SlaveDesyncPosition) auf 450.0mm festgelegt.
- Der Punkt für den vollständige Abbau der Kopplung (SlaveStopPosition) wurde auf 500.0mm festgelegt. Somit ist der dafür vorgesehene Weg  $500.0\text{mm} - 450\text{mm} \Rightarrow 50\text{mm}$ .
- Bei  $\text{Ratio}=1.0$  wird der Master dabei ebenfalls einen Weg von  $2.0 * 50\text{mm} \Rightarrow 100\text{mm}$  zurücklegen.
- Wege, die der Master vor TM\_1 und nach TM\_4 im obigen Scope zurücklegt betreffen den Slave nicht.
- Wenn der Master seine Geschwindigkeit zwischen TM\_1 und TM\_2 bzw. TM\_3 und TM\_4 ändert wird der Slave dies berücksichtigen. Dann kann sein Geschwindigkeitsverlauf weniger übersichtlich sein als in obigem Scope.

### Beispiel #2

Hier bewegen sich Master und Slave in entgegengesetzter Richtung. Beide kehren zu ihrer Ausgangsposition zurück.



- Hier ist die Ausgangsposition nicht um 50mm über der SlaveSyncPosition, sondern um den gleichen Betrag darunter.

**Beispiel #3**

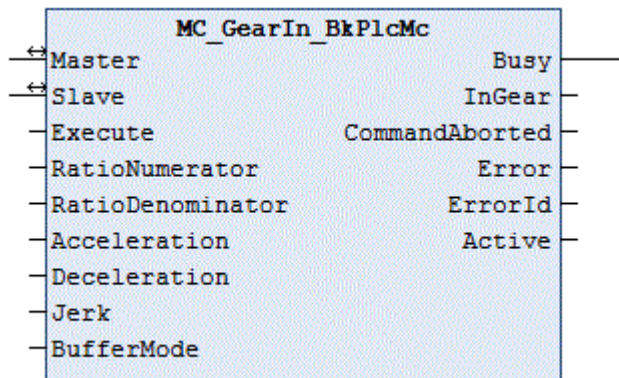
Hier bewegen sich Master und Slave in gleicher Richtung. Allerdings kehrt der Slave erst durch einen MC\_MoveAbsolut\_BkPlcMc() in seine Ausgangsposition zurück.



- Hier ist die Ausgangssituation die gleiche wie in Beispiel #1.
- Allerdings ist hier die SlaveStopPosition eine andere. Der Slave stoppt entsprechend früher.

- Der Abstand zwischen SlaveDesyncPosition und SlaveStopPosition ist hier 100mm. Dadurch erzeugt der Abbau der Kopplung ein anderes Profil als der Aufbau.
- Nachdem der Master seine Bewegung abgeschlossen hat wird der Baustein deaktiviert und der Slave mit einem eigenen Befehl zur Ausgangsposition gefahren.

### 4.2.7 MC\_GearIn\_BkPlcMc (ab V3.0)



Der Funktionsbaustein startet und überwacht eine Kopplung zwischen zwei Achsen. Zum Lösen der Kopplung ist ein [MC\\_GearOut\\_BkPlcMc](#) [▶ 66] Baustein zu verwenden.

#### Eingänge

```

VAR_INPUT
  Execute:          BOOL;
  RatioNumerator:  INT;
  RatioDenominator: INT;
  Acceleration:    LREAL;
  Deceleration:    LREAL;
  Jerk:            LREAL; (ab/from V3.0.5)
  BufferMode:      MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc; (ab/from V3.0.8)
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Kopplung.
RatioNumerator	INT	[1, 1] Diese Parameter beschreiben den Kopplungsfaktor in der Form eines Getriebes.
RatioDenominator	INT	[1, 1] Diese Parameter beschreiben den Kopplungsfaktor in der Form eines Getriebes.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die für das Aufsynchronisieren zulässige Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die für das Aufsynchronisieren zulässige Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Jerk	LREAL	[mm/s <sup>3</sup> ] Der anzuwendende Ruck.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

#### Ein-/Ausgänge

```

VAR_INOUT
  Master:      Axis_Ref_BkPlcMc;
  Slave:      Axis_Ref_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Master	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.
Slave	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  InGear:        BOOL;
  CommandAborted: BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  Active:        BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
InGear	BOOL	Hier wird das erstmalige erfolgreiche Aufsynchronisieren der Achsen signalisiert. Das Signal bleibt anschließend auch dann anstehen, wenn die Synchronisierung zu einem späteren Zeitpunkt zeitweise oder bleibend aussetzt.
CommandAborted	BOOL	Hier wird ein Abbruch der Kopplung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Active	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Als nächstes wird überprüft, ob **RatioDenominator** gleich 0 ist. In diesem Fall wird mit **Error** und **ErrorID:=dwTcHydErrCdIllegalGearFactor** reagiert.
- Derzeit kann die Kopplung nur dann aktiviert werden, wenn sowohl der Master als auch der Slave im Stillstand sind. Andernfalls wird mit **Error** und **ErrorID:=dwTcHydErrCdNotStartable** reagiert.
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn der Bewegungsalgorithmus bereits einen Fehlercode signalisiert wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Kopplung initiiert. Die Achse befindet sich ab jetzt im Zustand McState\_Synchronizedmotion [► 96] und der Baustein beginnt mit der Überwachung der Kopplung.

Wenn bei aktiver Kopplung die Slaveachse erstmalig die durch die Kopplung geforderte Geschwindigkeit erreicht wird dies am Ausgang **InGear** signalisiert. Da die Kopplung derzeit nur im Stillstand aktiviert werden kann ist dies unmittelbar der Fall. Sollte während aktiver Kopplung die Slaveachse aus einem beliebigen Grund den Vorgaben nicht folgen können bleibt **InGear** unverändert.

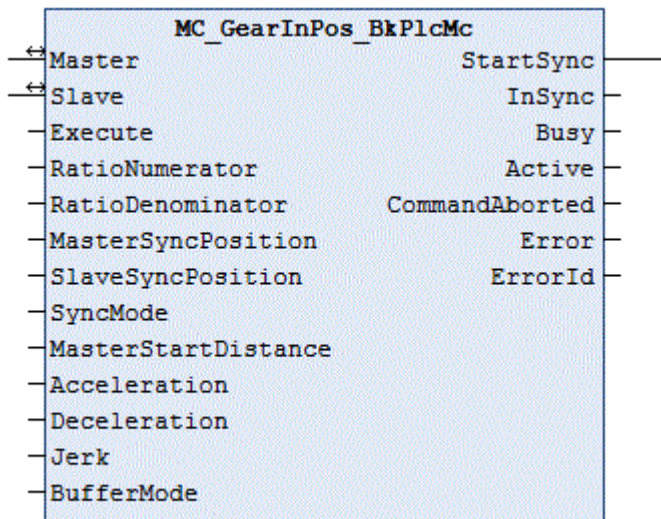
Tritt bei aktiver Kopplung im Bewegungsgenerator ein Fehlercode auf wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Kopplung auf FALSE gesetzt bleibt die bestehende Kopplung unbeeinflusst wirksam.



Der Ausgang **Active** ist derzeit mit dem Ausgang **Busy** identisch.

### 4.2.8 MC\_GearInPos\_BkPlcMc (ab V3.0.33)



Der Funktionsbaustein startet und überwacht eine fliegende Kopplung zwischen zwei Achsen. Zum Lösen der Kopplung ist ein [MC\\_GearOut\\_BkPlcMc](#) [66] Baustein zu verwenden.

#### Eingänge

```

VAR_INPUT
  Execute:          BOOL;
  RatioNumerator:  INT;
  RatioDenominator: INT;
  MasterSyncPosition: LREAL;
  SlaveSyncPosition: LREAL;
  SyncMode:        INT;
  MasterStartDistance: LREAL;
  Acceleration:    LREAL;
  Deceleration:    LREAL;
  Jerk:            LREAL;      (ab/from V3.0.5)
  BufferMode:      MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Kopplung.
RatioNumerator	INT	[1, 1] Diese Parameter beschreiben den Kopplungsfaktor in der Form eines Getriebes.
RatioDenominator	INT	[1, 1] Diese Parameter beschreiben den Kopplungsfaktor in der Form eines Getriebes.
MasterSyncPosition	LREAL	[mm] Ab dieser Master-Position ist die Kopplung vollständig wirksam.
SlaveSyncPosition	LREAL	[mm] Ab dieser Slave-Position ist die Kopplung vollständig wirksam.
SyncMode	INT	derzeit nicht unterstützt.
MasterStartDistance	LREAL	[mm] Über diesen Weg des Masters wird die Kopplung aufgebaut.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die für das Aufsynchronisieren zulässige Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die für das Aufsynchronisieren zulässige Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Jerk	LREAL	[mm/s <sup>3</sup> ] Der anzuwendende Ruck.



Name	Typ	Beschreibung
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

 **Ein-/Ausgänge**

```
VAR_INOUT
  Master:      Axis_Ref_BkPlcMc;
  Slave:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Master	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.
Slave	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  StartSync:      BOOL;
  InSync:         BOOL;
  Busy:           BOOL;
  Active:         BOOL;
  CommandAborted: BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
StartSync	BOOL	Hier wird die Übergangsphase zwischen Ruhezustand und vollständig wirksamer Kopplung signalisiert.
InSync	BOOL	Hier wird das erstmalige erfolgreiche Aufsynchronisieren der Achsen signalisiert. Das Signal bleibt anschließend auch dann anstehen, wenn die Synchronisierung zu einem späteren Zeitpunkt zeitweise oder bleibend aussetzt.
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird ein Abbruch der Kopplung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Als nächstes wird überprüft, ob **RatioDenominator** gleich 0 ist. In diesem Fall wird mit **Error** und **ErrorID:=dwTcHydErrCdIllegalGearFactor** reagiert.
- Ist der **RatioDenominator** kleiner als 0 wird mit **Error** und **ErrorID:=dwTcHydErrCdNotSupport** reagiert.
- Die Kopplung kann nur dann aktiviert werden, wenn der Slave im Stillstand ist. Andernfalls wird mit **Error** und **ErrorID:=dwTcHydErrCdNotStartable** reagiert.
- Wenn der Absolutwert des **MasterStartDistance** zu klein ist wird mit **Error** und **ErrorID:=dwTcHydErrCdCannotSynchronize** reagiert.
- Wenn die Istposition des Masters nicht zwischen **MasterSyncPosition** und dem von **MasterStartDistance** festgelegten Ende der Synchronisationsstrecke liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdCannotSynchronize** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Kopplung initiiert. Die Slave-Achse befindet sich zunächst weiter im Zustand McState\_Standstill [▶ 96]. Erst wenn die Master-Achse erstmalig den Anfang der Synchronisationsstrecke erreicht meldet die Slave-Achse McState\_Synchronizedmotion [▶ 96], signalisiert **StartSync** und der Baustein beginnt mit der Überwachung der Kopplung. Sobald die Achse erstmalig das Ende der Synchronisationsstrecke erreicht signalisiert die Slave-Achse **InSync**. Sollte die Master-Achse zu einem späteren Zeitpunkt den Anfang der Synchronisationsstrecke rückwärts passieren, wird die Kopplung nicht wieder gelöst.

Tritt bei aktiver Kopplung im Bewegungsgenerator ein Fehlercode auf wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Kopplung auf FALSE gesetzt bleibt die bestehende Kopplung unbeeinflusst wirksam.

Es steht unter [#103](#) [▶ 353] ein Beispiel zur Verfügung.

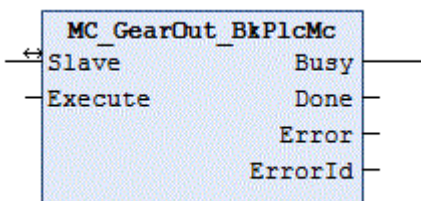


Der Baustein unterstützt nicht den Funktionsumfang der TwinCAT NC.



Der Ausgang Active ist derzeit mit dem Ausgang Busy identisch.

### 4.2.9 MC\_GearOut\_BkPlcMc (ab V3.0)



Der Funktionsbaustein löst eine Kopplung zwischen zwei Achsen. Diese Kopplung muss zuvor mit einem MC GearIn BkPlcMc [▶ 62] oder einem MC GearInPos BkPlcMc [▶ 64] Baustein hergestellt worden sein.

#### Eingänge

```

VAR_INPUT
  Execute:      BOOL;
END_VAR
  
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang löst die Kopplung.

#### Ein-/Ausgänge

```

VAR_INOUT
  Slave:      Axis_Ref_BkPlcMc;
END_VAR
  
```

Name	Typ	Beschreibung
Slave	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [▶ 84] zu übergeben.

#### Ausgänge

```

VAR_OUTPUT
  Busy:      BOOL;
  Done:     BOOL;
  
```

```
Error:          BOOL;
ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Bewegung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

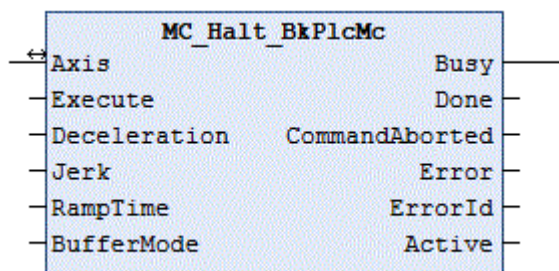
**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn die Achse nicht in einer Getriebekopplung betrieben wird signalisiert der Baustein unmittelbar **Done** und unterlässt alle weiteren Überprüfungen oder Aktivitäten.
- Ist die aktuelle Sollgeschwindigkeit der Achse kleiner als die von pStAxParams.fCreepSpeed festgelegte Geschwindigkeit geht die Achse unmittelbar in McState\_Standstill über und baut die Restgeschwindigkeit ab. Es wird **Done** signalisiert und alle weiteren Überprüfungen oder Aktivitäten werden unterlassen.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten und nicht bereits aus einem der genannten Gründe **Done** signalisiert wird erfolgt eine Umwandlung der von der Getriebekopplung kontrollierten Bewegung in eine vom Master unabhängige endlose Bewegung mit gleicher Geschwindigkeit und Richtung. Wenn diese Umwandlung erfolgreich ausgeführt wurde wird **Done** signalisiert, andernfalls wird mit **Error** und **ErrorID:=Fehlercode** reagiert.

**4.2.10 MC\_Halt\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein bricht eine aktuell ausgeführte Bewegung einer Achse ab und überwacht die Stopp-Operation.

**i** Die von diesem Baustein eingeleitete Stopp-Operation ist durch andere Bausteine unterbrechbar. Soll eine Achse während der Stopp-Operation nicht durchstartbar sein ist ein MC\_Stop\_BkPlcMc Baustein zu verwenden.

**Eingänge**

```
VAR_INPUT
Execute:          BOOL;
Deceleration:    LREAL; (ab/from V3.0.5)
Jerk:            LREAL; (ab/from V3.0.5)
RampTime:        LREAL; (ab/from V3.0.5)
BufferMode:      MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc; (ab/from V3.0.8)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang beendet eine Bewegung der Achse.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die anzuwendende Verzögerung.
Jerk	LREAL	[mm/s <sup>3</sup> ] Der anzuwendende Ruck.

Name	Typ	Beschreibung
RampTime	LREAL	[s] Die geforderte Anhaltezeit.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc zu übergeben.

 **Ausgänge**

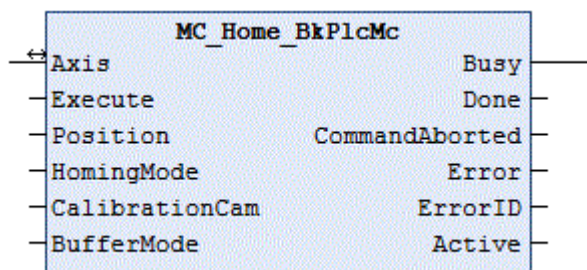
```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  Active:        BOOL;
  CommandAborted: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Operation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Active	BOOL	Hier wird angezeigt, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird angezeigt, dass die Abarbeitung dieses Kommandos durch ein anderes Kommando abgebrochen wurde.

**Verhalten des Bausteins**

Das Verhalten des Bausteins ist identisch mit dem des MC\_Stop BkPlcMc [▶ 81]() Bausteins. Der einzige Unterschied ist, dass die Abarbeitung des Kommandos durch andere Bausteine abbrechbar ist.

**4.2.11 MC\_Home\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein startet und überwacht die Referenzfahrt einer Achse.

 **Eingänge**

```
VAR_INPUT
  Execute:       BOOL;
  Position:      LREAL;
  HomingMode:    MC_HomingMode_BkPlcMc;
```

```

CalibrationCam: BOOL;
BufferMode: MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc; (ab/from V3.0.8)
END_VAR

```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Referenzfahrt.
Position	LREAL	[mm] Die Referenzposition.
HomingMode	MC_HomingMode_BkPlcMc	Hier wird die zu verwendende <u>Methode</u> [► 115] festgelegt.
CalibrationCam	BOOL	Hier kann der Referenzierindex (Nocken) direkt übergeben werden.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

 Ein-/Ausgänge

```

VAR_INOUT
Axis: Axis_Ref_BkPlcMc;
END_VAR

```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [► 84] zu übergeben.

 Ausgänge

```

VAR_OUTPUT
Busy: BOOL;
Done: BOOL;
CommandAborted: BOOL;
Error: BOOL;
ErrorID: UDINT;
END_VAR

```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Referenzfahrt signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch der Referenzfahrt signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Eine Referenzfahrt kann nur aus einem fehlerfreien Stillstand begonnen werden. Ist dies nicht der Fall wird mit **Error** und **ErrorID:=dwTcHydErrCdNotStartable** bzw. dem vorliegenden Errorcode reagiert.
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn eine der in den Achs-Parametern angegebene Geschwindigkeiten zu klein (weniger als 1% der Referenzgeschwindigkeit) ist wird mit **Error** und **ErrorID:=dwTcHydErrCdSetVelo** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Referenzfahrt initiiert. Der genaue Ablauf wird von HomingMode [► 115] festgelegt. Sollte während der Abarbeitung der Referenzfahrt vom Bewegungsalgorithmus ein Fehlercode signalisiert wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert. Wird die vollständige Referenzfahrt durch die Aktivität eines anderen Bausteins verhindert, wird mit **CommandAborted** reagiert. Ein erfolgreicher Abschluss der Referenzfahrt wird mit **Done** gemeldet.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Referenzfahrt auf FALSE gesetzt wird die eingeleitete Referenzfahrt unbeeinflusst weiter bearbeitet. Die Signale am Ende der Bewegung (**Error**, **ErrorID**, **CommandAborted**, **Done**) werden für einen Zyklus gegeben.

**Hinweis** Wenn die Applikation nicht selbst eine Referenzposition festlegt, sondern ein mit den Maschinendaten abgespeicherter und geladener Wert Verwendung finden soll ist die Benutzung von **fEnc\_DefaultHomePosition** in **pStAxParams** vorgesehen. Sollten situationsabhängig verschiedene Werte benötigt werden, sollte auf **fCustomerData[]** in **pStAxParams** zurückgegriffen werden.

Ist als Encoder-Typ **iTcMc\_EncoderSim** eingestellt, wird unabhängig von **HomingMode** und **Axis\_Ref\_BkPlcMc** [► 84].**stAxParams.nEnc\_HomingType** der Modus **MC\_Direct\_BkPlcMc** wirksam.

### MC\_DefaultHomingMode\_BkPlcMc

Die Referenziermethode wird nicht durch die Applikation sondern durch **Axis\_Ref\_BkPlcMc** [► 84].**stAxParams.nEnc\_HomingType** festgelegt. Dabei gelten die folgenden Regeln:

<b>nEnc_HomingType</b>	<b>MC_HomingMode_BkPlcMc</b>
<b>iTcMc_HomingOnBlock</b>	<b>MC_Block_BkPlcMc</b>
<b>iTcMc_HomingOnIndex</b>	<b>MC_AbsSwitch_BkPlcMc</b>
<b>iTcMc_HomingOnSync</b>	<b>MC_RefPulse_BkPlcMc</b>
<b>iTcMc_HomingOnExec</b>	<b>MC_Direct_BkPlcMc</b>

### MC\_AbsSwitch\_BkPlcMc

Die Achse wird mit **Axis\_Ref\_BkPlcMc** [► 84].**stAxParams.fEnc\_RefIndexVelo** in der durch **bEnc\_RefIndexPositive** festgelegten Richtung bewegt. Nimmt **CalibrationCam** den Zustand TRUE an oder wird in **Axis\_Ref\_BkPlcMc** [► 84].**stAxRtData.nDeCtrlDWord** die Referenznocke (Bit 5, **dwTcHydDcDwRefIndex**) erkannt stoppt die Achse. Anschließend wird sie mit **fEnc\_RefSyncVelo** in der durch **bEnc\_RefSyncPositive** festgelegten Richtung bewegt, bis die Referenznocke wieder verlassen wird. Der Istwert der Achse wird auf den Wert der Referenzposition gesetzt.

### MC\_LimitSwitch\_BkPlcMc

Derzeit nicht unterstützt.

### MC\_RefPulse\_BkPlcMc

Die Achse wird mit **Axis\_Ref\_BkPlcMc** [► 84].**stAxParams.fEnc\_RefIndexVelo** in der durch **bEnc\_RefIndexPositive** festgelegten Richtung bewegt. Nimmt **CalibrationCam** den Zustand TRUE an oder wird in **Axis\_Ref\_BkPlcMc** [► 84].**stAxRtData.nDeCtrlDWord** die Referenznocke (Bit 5, **dwTcHydDcDwRefIndex**) erkannt stoppt die Achse. Anschließend wird sie mit **fEnc\_RefSyncVelo** in der durch **bEnc\_RefSyncPositive** festgelegten Richtung bewegt, bis die Referenznocke wieder verlassen wird. Dann wird das Hardware-Latch des Encoders aktiviert und die Achse so lange weiter bewegt, bis das Latch gültig wird. Nach dem Stoppen der Achse wird der Istwert der Achse auf einen Wert gesetzt, der aus der Referenzposition und der seit dem Sync-Puls des Encoders zurückgelegten Strecke errechnet wird.

### MC\_Direct\_BkPlcMc

Der Istwert der Achse wird unmittelbar auf den Wert der Referenzposition gesetzt.

### MC\_Absolute\_BkPlcMc

Derzeit nicht unterstützt.

**MC\_Block\_BkPlcMc**

Die Achse wird mit `Axis_Ref_BkPlcMc [▶ 84].stAxParams.fEnc_RefIndexVelo` in der durch `bEnc_RefIndexPositive` festgelegten Richtung bewegt. Wird für eine Zeitdauer von 2 Sekunden keine Bewegung festgestellt, gilt der Festanschlag (Block) als erreicht. Der Istwert der Achse wird auf den Wert der Referenzposition gesetzt.

Ab einer Version 3.0.41 vom 12.10.2017 ist es möglich, die Zeitdauer für die Erkennung des Blocks zu verändern. Siehe hierzu `ST_TcHydAxRtData [▶ 131].fBlockDetectDelay`.

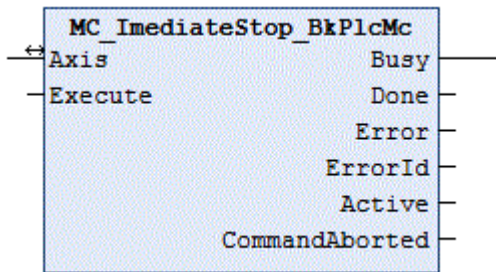
**MC\_FlyingSwitch\_BkPlcMc**

Derzeit nicht unterstützt.

**MC\_FlyingRefPulse\_BkPlcMc**

Derzeit nicht unterstützt.

**4.2.12 MC\_ImmediateStop\_BkPlcMc (ab V3.0.5)**



Der Funktionsbaustein bricht eine aktuell ausgeführte Bewegung einer Achse ab.

**Eingänge**

```
VAR_INPUT
    Execute:      BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang beendet eine Bewegung der Achse.

**Ein-/Ausgänge**

```
VAR_INOUT
    Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc [▶ 84]</code> zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
    Busy:      BOOL;
    Done:      BOOL;
    Error:      BOOL;
    ErrorID:    UDINT;
    Active:     BOOL;
    CommandAborted: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Operation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Active	BOOL	Hier wird angezeigt, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird angezeigt, dass die Abarbeitung dieses Kommandos durch ein anderes Kommando abgebrochen wurde.

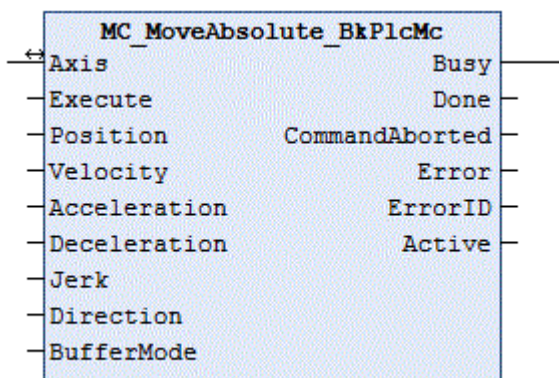
**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Ein Stopp kann nur ausgeführt werden, wenn die Achse eine aktive Bewegung ausführt. Befindet sie sich im Stillstand meldet der Baustein sofort **Done** .
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stop-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Befindet sich die Achse in einem Zustand, in dem sie durch eine Kopplung mit einer anderen Achse oder einen vergleichbaren Mechanismus kontrolliert wird reagiert sie mit **Error** und **ErrorID:=dwTcHydErrCdNotReady**.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Stop-Operation initiiert. Dabei wird der Stellwert der Achse unmittelbar und ohne jede Rampe auf 0 gesetzt. Anschließend werden alle Ausgaben von Steuer- oder Regelspannungen unterdrückt, solange **Execute** auf TRUE gesetzt ist.

**4.2.13 MC\_MoveAbsolute\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein startet und überwacht die Bewegung einer Achse.

**Eingänge**

```

VAR_INPUT
  Execute:          BOOL;
  Position:         LREAL;
  Velocity:         LREAL;
  Acceleration:    LREAL;
  Deceleration:    LREAL;
  Jerk:            LREAL;
  Direction:       MC_Direction_BkPlcMc:=MC_Shortest_Way_BkPlcMc;    (ab/from V3.0.8)
  BufferMode:      MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;        (ab/from V3.0.8)
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Bewegung.



Name	Typ	Beschreibung
Position	LREAL	[mm] Die Zielposition der Bewegung in Istwert-Einheiten der Achse.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] reserviert.
Direction	MC_Direction_BkPlcMc	reserviert. Dieser Eingang wurde nur aus Kompatibilitätsgründen ergänzt und sollte nicht oder mit der Konstanten MC_Shortest_Way_BkPlcMc belegt werden. (ab V3.0.8)
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  CommandAborted: BOOL;
  Error:         BOOL;
  ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Bewegung signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch der Bewegung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Als nächstes wird überprüft, ob **Position** hinter einem aktiven Software-Endschalter liegt. In diesem Fall wird mit **Error** und **ErrorID:=dwTcHydErrCdSoftEnd** reagiert.
- In Abhängigkeit vom in **Axis.pStAxParams^.nProfile** festgelegten Bewegungsalgorithmus kann die Achse entweder nur aus dem Stillstand oder auch aus einer anderen noch nicht abgeschlossenen Bewegung heraus die hier gestartete Bewegung beginnen. Sollte sie derzeit nicht in der Lage sein diesen neuen Auftrag zu akzeptieren, wird mit **Error** und **ErrorID:=dwTcHydErrCdNotStartable** reagiert.

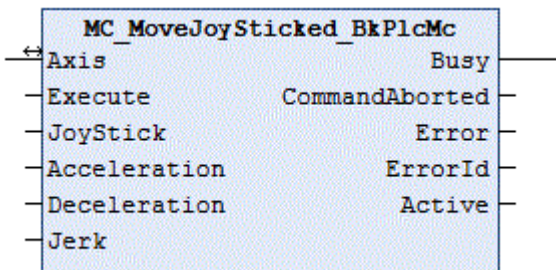
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Velocity** zu klein (weniger als 1% der Referenzgeschwindigkeit) ist, wird mit **Error** und **ErrorID:=dwTcHydErrCdSetVelo** reagiert.
- Wenn **Acceleration** zu klein ist (**Velocity** kann nicht innerhalb von 100 Sekunden erreicht werden), wird mit **Error** und **ErrorID:=dwTcHydErrCdAcc** reagiert.
- Wenn **Deceleration** zu klein ist (**Velocity** kann nicht innerhalb von 100 Sekunden abgebaut werden), wird mit **Error** und **ErrorID:=dwTcHydErrCdDec** reagiert.
- Wenn der Bewegungsalgorithmus bereits einen Fehlercode signalisiert, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten, wird die Bewegung initiiert. Dazu werden die Parameter **Position**, **Velocity**, **Acceleration** und **Deceleration** auf die maximal zulässigen Werte begrenzt und an den Bewegungsalgorithmus übergeben. Die Achse befindet sich ab jetzt im Zustand McState\_DiscreteMotion [► 96] und der Baustein beginnt mit der Überwachung der Bewegung.

Sollte während der Abarbeitung der Bewegung vom Bewegungsalgorithmus ein Fehlercode signalisiert werden, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert. Wird die vollständige Bewegung durch die Aktivität eines anderen Bausteins verhindert, wird mit **CommandAborted** reagiert. Erreicht der Bewegungsalgorithmus die Zielbedingungen der Achse, wird mit **Done** reagiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Bewegung auf FALSE gesetzt, wird die eingeleitete Bewegung unbeeinflusst weiter bearbeitet. Die Signale am Ende der Bewegung (**Error**, **ErrorID**, **CommandAborted**, **Done** ) werden für einen Zyklus gegeben.

#### 4.2.14 MC\_MoveJoySticked\_BkPlcMc (ab V3.0)



Der Funktionsbaustein startet und überwacht die Bewegung einer Achse.

**i** Diese Funktion wird derzeit nur durch Achsen unterstützt, die durch einen Baustein vom Typ MC\_AxRuntimeCtrlBased\_BkPlcMc kontrolliert werden (in Vorbereitung: MC\_AxRunTimeTimeRamp\_BkPlcMc). Die Auswahl eines solchen Bausteins erfolgt dadurch, dass in nProfileType in ST\_TcHydAxParam die entsprechende Konstante aus E\_TcMcProfileType vorgegeben wird.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  JoyStick:     LREAL;
  Acceleration: LREAL;
  Deceleration: LREAL;
  Jerk:        LREAL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Bewegung.
JoyStick	LREAL	[1] Die auf den Wertebereich ±1.0 normierte Geschwindigkeitsvorgabe durch das Kontrollgerät.

Name	Typ	Beschreibung
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Jerk	LREAL	[mm/s <sup>3</sup> ] reserviert.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  CommandAborted: BOOL;
  Error:         BOOL;
  ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird ein Abbruch der Bewegung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

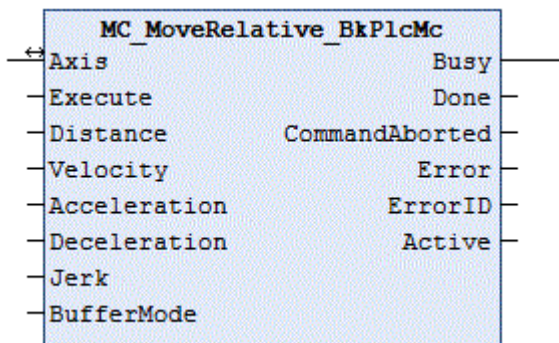
Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn der Bewegungsalgorithmus bereits einen Fehlercode signalisiert, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.
- Als nächstes wird überprüft, ob der Generator der Achse die geforderte Funktion unterstützt. Ist dies nicht der Fall, wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Bewegung initiiert. Dazu wird der Bewegungsalgorithmus in den Zustand iTcHydStateExtGenerated und die Achse in den Zustand McState\_Synchronizedmotion versetzt. Die Geschwindigkeit der Achse wird durch **JoyStick** und ST\_TcHydAxParam [► 120].fRefVelo festgelegt. Bei einer Änderung der Geschwindigkeit wird eine Rampenbegrenzung auf ST\_TcHydAxParam [► 120].fMaxAcc vorgenommen. Fährt die Achse in Richtung auf einen aktiven Software-Endschalter, wird die Geschwindigkeit in Abhängigkeit von der verbleibenden Entfernung so begrenzt, dass eine korrekte Zielfahrt auf den Endschalter zustande kommt.

Eine fallende Flanke an **Execute** versetzt den Bewegungsalgorithmus in den Zustand iTcHydStateTcDecP oder iTcHydStateTcDecM und die Achse in den Zustand McState\_Standstill. Falls sich die Achse zu diesem Zeitpunkt bewegt wird sie mit einer Stopprampe abgebremst und geht in den Zustand iTcHydStateIdle über.

### 4.2.15 MC\_MoveRelative\_BkPlcMc (ab V3.0)



Der Funktionsbaustein startet und überwacht die Bewegung einer Achse.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  Distance:     LREAL;
  Velocity:     LREAL;
  Acceleration: LREAL;
  Deceleration: LREAL;
  Jerk:        LREAL;
  BufferMode:   MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;    (ab/from V3.0.8)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Bewegung.
Distance	LREAL	[mm] Die Entfernung zur Zielposition der Bewegung in Istwert-Einheiten der Achse.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Jerk	LREAL	[mm/s <sup>3</sup> ] reserviert.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  CommandAborted: BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Bewegung signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch der Bewegung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

### Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

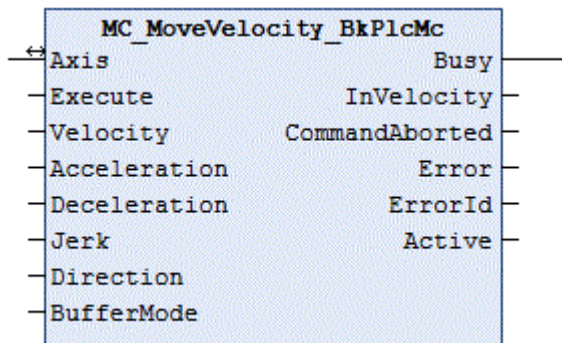
- Als nächstes wird überprüft, ob eine Bewegung um **Distance** zu einem Konflikt mit einem aktiven Software-Endschalter liegt. In diesem Fall wird mit **Error** und **ErrorID:=dwTcHydErrCdSoftEnd** reagiert.
- In Abhängigkeit vom in **Axis.pStAxParams^.nProfile** festgelegten Bewegungsalgorithmus kann die Achse entweder nur aus dem Stillstand oder auch aus einer anderen noch nicht abgeschlossenen Bewegung heraus die hier gestartete Bewegung beginnen. Sollte sie derzeit nicht in der Lage sein diesen neuen Auftrag zu akzeptieren, wird mit **Error** und **ErrorID:=dwTcHydErrCdNotStartable** reagiert.
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch, wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Velocity** zu klein (weniger als 1% der Referenzgeschwindigkeit) ist, wird mit **Error** und **ErrorID:=dwTcHydErrCdSetVelo** reagiert.
- Wenn **Acceleration** zu klein ist (**Velocity** kann nicht innerhalb von 100 Sekunden erreicht werden), wird mit **Error** und **ErrorID:=dwTcHydErrCdAcc** reagiert.
- Wenn **Deceleration** zu klein ist (**Velocity** kann nicht innerhalb von 100 Sekunden abgebaut werden), wird mit **Error** und **ErrorID:=dwTcHydErrCdDec** reagiert.
- Wenn der Bewegungsalgorithmus bereits einen Fehlercode signalisiert, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Bewegung initiiert. Dazu werden die Parameter **Distance**, **Velocity**, **Acceleration** und **Deceleration** auf die maximal zulässigen Werte begrenzt und an den Bewegungsalgorithmus übergeben. Die Achse befindet sich ab jetzt im Zustand **McState\_DiscreteMotion** [► 96] und der Baustein beginnt mit der Überwachung der Bewegung.

Sollte während der Abarbeitung der Bewegung vom Bewegungsalgorithmus ein Fehlercode signalisiert werden, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert. Wird die vollständige Bewegung durch die Aktivität eines anderen Bausteins verhindert, wird mit **CommandAborted** reagiert. Erreicht der Bewegungsalgorithmus die Zielbedingungen der Achse, wird mit **Done** reagiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Bewegung auf FALSE gesetzt, wird die eingeleitete Bewegung unbeeinflusst weiter bearbeitet. Die Signale am Ende der Bewegung (**Error**, **ErrorID**, **CommandAborted**, **Done**) werden für einen Zyklus gegeben.

## 4.2.16 MC\_MoveVelocity\_BkPlcMc (ab V3.0)



Der Funktionsbaustein startet und überwacht die Bewegung einer Achse.

### Eingänge

```
VAR_INPUT
  Execute:          BOOL;
  Velocity:         LREAL;
  Acceleration:    LREAL;
  Deceleration:    LREAL;
  Direction:       MC_Direction_BkPlcMc;
  BufferMode:      MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;    (ab/from V3.0.8)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Bewegung.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde.
Direction	MC_Direction_BkPlcMc	Eine entsprechend <a href="#">MC_Direction_BkPlcMc</a> [ <a href="#">▶ 115</a> ] codierte Richtungsangabe.
BufferMode	MC_Direction_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten <a href="#">Aborting_BkPlcMc</a> belegt werden. (ab V3.0.8)

### Ein-/Ausgänge

```
VAR_INOUT
  Axis:            Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis Ref BkPlcMc</a> [ <a href="#">▶ 84</a> ] zu übergeben.

### Ausgänge

```
VAR_OUTPUT
  Busy:           BOOL;
  InVelocity:     BOOL;
  CommandAborted: BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
InVelocity	BOOL	Dieser Ausgang wird TRUE, wenn die Achse das erste Mal die geforderte Geschwindigkeit erreicht.
CommandAborted	BOOL	Hier wird ein Abbruch der Bewegung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

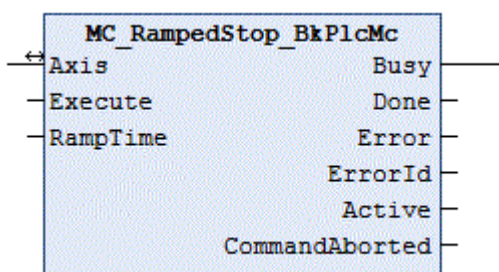
- In Abhängigkeit vom in **Axis.pStAxParams^.nProfile** festgelegten Bewegungsalgorithmus kann die Achse entweder nur aus dem Stillstand oder auch aus einer anderen noch nicht abgeschlossenen Bewegung heraus die hier gestartete Bewegung beginnen. Sollte sie derzeit nicht in der Lage sein diesen neuen Auftrag zu akzeptieren, wird mit **Error** und **ErrorID:=dwTcHydErrCdNotStartable** reagiert.
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch, wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Velocity** zu klein (weniger als 1% der Referenzgeschwindigkeit) ist, wird mit **Error** und **ErrorID:=dwTcHydErrCdSetVelo** reagiert.
- Wenn der Bewegungsalgorithmus bereits einen Fehlercode signalisiert, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Bewegung initiiert. Dazu wird in Abhängigkeit von **Direction** und den Parametern der Softwareendschalter ein Wert für die Zielposition gewählt. Dann werden die Parameter **Velocity**, **Acceleration** und **Deceleration** auf die maximal zulässigen Werte begrenzt und an den Bewegungsalgorithmus übergeben. Die Achse befindet sich ab jetzt im Zustand McState\_Continuousmotion [► 96] und der Baustein beginnt mit der Überwachung der Bewegung.

Sollte während der Abarbeitung der Bewegung vom Bewegungsalgorithmus ein Fehlercode signalisiert werden, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert. Wird die vollständige Bewegung durch die Aktivität eines anderen Bausteins verhindert, wird mit **CommandAborted** reagiert. Erreicht der Bewegungsalgorithmus die geforderte Geschwindigkeit wird **InVelocity** gesetzt.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Bewegung auf FALSE gesetzt wird die eingeleitete Bewegung unbeeinflusst weiter bearbeitet. Die Signale am Ende der Bewegung (**Error**, **ErrorID**, **CommandAborted**, **InVelocity**) werden für einen Zyklus gegeben.

**4.2.17 MC\_RampedStop\_BkPlcMc**



Der Funktionsbaustein bricht eine aktuell ausgeführte Bewegung ab.

## Eingänge

```
VAR_INPUT
  Execute:          BOOL;
  RampTime:        LREAL; (ab/from V3.0.5)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang beendet eine Bewegung der Achse.
RampTime	LREAL	[s] Die geforderte Anhaltezeit.

## Ein-/Ausgänge

```
VAR_INOUT
  Axis:            Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

## Ausgänge

```
VAR_OUTPUT
  Busy:            BOOL;
  Done:            BOOL;
  Error:           BOOL;
  ErrorID:         UDINT;
  Active:          BOOL;
  CommandAborted: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Operation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Active	BOOL	Hier wird angezeigt, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird angezeigt, dass die Abarbeitung dieses Kommandos durch ein anderes Kommando abgebrochen wurde.

## Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

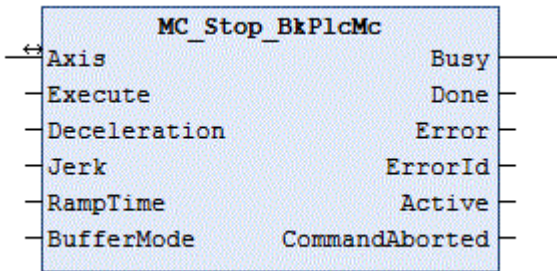
- Ein Stopp kann nur ausgeführt werden, wenn die Achse eine aktive Bewegung ausführt. Befindet sie sich im Stillstand meldet der Baustein sofort **Done**.
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Befindet sich die Achse in einem Zustand, in dem sie durch eine Kopplung mit einer anderen Achse oder einen vergleichbaren Mechanismus kontrolliert wird, reagiert sie mit **Error** und **ErrorID:=dwTcHydErrCdNotReady**.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Stop-Operation initiiert. Dabei wird **RampTime** verwendet, um unter Berücksichtigung der Bezugsgeschwindigkeit eine Verzögerung zu berechnen. Mit dieser Verzögerung wird die Sollgeschwindigkeit mit einer reinen Zeitrampe bis auf 0 reduziert.

**VORSICHT** Es wird keine definierte Endposition angefahren und die Achse kann einen Softwareendschalter überfahren.



### 4.2.18 MC\_Stop\_BkPlcMc (ab V3.0)



Der Funktionsbaustein bricht eine aktuell ausgeführte Bewegung einer Achse ab und überwacht die Stopp-Operation.

**i** Die von diesem Baustein eingeleitete Stopp-Operation ist nicht durch andere Bausteine unterbrechbar. Soll eine Achse während der Stopp-Operation durchstartbar sein ist ein MC\_Halt\_BkPlcMc Baustein zu verwenden.

#### Eingänge

```
VAR_INPUT
  Execute:          BOOL;
  Deceleration:    LREAL; (ab/from V3.0.5)
  Jerk:            LREAL; (ab/from V3.0.5)
  RampTime:       LREAL; (ab/from V3.0.5)
  BufferMode:      MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc; (ab/from V3.0.8)
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang beendet eine Bewegung der Achse.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die anzuwendende Verzögerung.
Jerk	LREAL	[mm/s <sup>3</sup> ] Der anzuwendende Ruck.
RampTime	LREAL	[s] Die geforderte Anhaltezeit.
BufferMode	MC_BufferMode_BkPlcMc	reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting_BkPlcMc belegt werden. (ab V3.0.8)

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  Active:        BOOL;
  CommandAborted: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Operation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Active	BOOL	Hier wird angezeigt, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird angezeigt, dass die Abarbeitung dieses Kommandos durch ein anderes Kommando abgebrochen wurde.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Ein Stopp kann nur ausgeführt werden, wenn die Achse eine aktive Bewegung ausführt. Befindet sie sich im Stillstand meldet der Baustein sofort **Done**.
- Befindet sich die Achse in einem gestörten Zustand oder führt sie gerade eine Stopp-Operation durch wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Befindet sich die Achse in einem Zustand, in dem sie durch eine Kopplung mit einer anderen Achse oder einen vergleichbaren Mechanismus kontrolliert wird, reagiert sie mit **Error** und **ErrorID:=dwTcHydErrCdNotReady**.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Stop-Operation initiiert. Dabei wird **Deceleration** angewendet, wenn dieser Parameter erkennbar größer als 0 ist. Andernfalls wird **RampTime** verwendet, um unter Berücksichtigung der Bezugsgeschwindigkeit eine Verzögerung zu berechnen. Ist ein ruckbegrenzender Stellwert-Generator ausgewählt wird **Jerk** verwendet, wenn dieser Parameter erkennbar größer als 0 ist. Ist keiner der genannten Parameter erkennbar größer als 0, werden die Achsparameter MaxDec und MaxJerk verwendet.

Unter Berücksichtigung der aktuellen Sollgeschwindigkeit und der jetzt geltenden Parameter wird die nächsterreichbare Position ermittelt und als neue Zielposition übernommen. Nachdem diese Position erreicht wurde, geht die Achse in ihr reguläres Verhalten im Ruhezustand über.

**i** Die RampTime legt die Zeit fest, in der die Achse von ihrer Bezugsgeschwindigkeit zum Stillstand abgebremst werden soll. Fährt die Achse mit einer anderen Geschwindigkeit, wird sich die Bremszeit auf einen entsprechenden Anteil verringern. Bei Stellwert-Generatoren mit Schleichfahrt addiert sich der dabei entstehende Zeitbedarf zur Bremszeit.

Sollte während der Abarbeitung der Bewegung vom Bewegungsalgorithmus ein Fehlercode signalisiert werden, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert. Wird die vollständige Abarbeitung durch die Aktivität eines anderen Bausteins verhindert, wird mit **CommandAborted** reagiert. Ein erfolgreicher Abschluss der Operation wird mit **Done** gemeldet.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Operation auf FALSE gesetzt, wird der eingeleitete Stopp unbeeinflusst weiter bearbeitet. Die Signale am Ende der Bewegung (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

**i** Der Ausgang Active ist derzeit mit dem Ausgang Busy identisch.

**4.2.19 MC\_MoveJog\_BkPIcMc**

MC_MoveJog_BkPIcMc		
Done	BOOL	Done
DoneForward	BOOL	Busy
DoneReverse	BOOL	CommandAborted
Error	BOOL	Error
ErrorID	UDINT	ErrorID
Active	BOOL	
Deceleration	REAL	
Jerk	REAL	
RampTime	REAL	

Der Funktionsbaustein startet und überwacht die Bewegung einer Achse.

 **Eingänge**

```
VAR_INPUT
  JogForward:      BOOL;
  JogBackwards:   BOOL;
  Mode:            E_TcMcJogMode;
  Position:        LREAL;
  Velocity:        LREAL;
  Acceleration:    LREAL;
  Deceleration:    LREAL;
  Jerk:            LREAL;
END_VAR
```

Name	Typ	Beschreibung
JogForward	BOOL	Mit der steigenden Flanke wird das Kommando ausgeführt und die Achse wird in positiver Fahrtrichtung bewegt. Je nach Betriebsart (siehe Mode), fährt die Achse solange das Signal TRUE bleibt oder stoppt automatisch nach einer festgelegten Distanz. Während der Bewegung werden keine weiteren Signalfanken angenommen, auch nicht am Eingang JogBackwards. Bei gleichzeitiger Signalfanke an den Eingängen JogForward und JogBackwards hat JogForward Vorrang.
JogBackwards	BOOL	Mit der steigenden Flanke wird das Kommando ausgeführt und die Achse wird in negativer Fahrtrichtung bewegt. JogForward und JogBackwards sollten alternativ getriggert werden, sind aber auch intern gegeneinander verriegelt.
Mode	E_TcMcJogMode	Der Eingang legt die Betriebsart E_TcMcJogMode fest, in der die Handfunktion ausgeführt werden soll.
Position	LREAL	[mm] relative Distanz, um die in der Betriebsart MC_JOGMODE_INCHING verfahren wird.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] reserviert.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:            Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:            BOOL;
  Done:            BOOL;
  CommandAborted: BOOL;
  Error:           BOOL;
  ErrorID:         UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Bewegung signalisiert.

Name	Typ	Beschreibung
CommandAborted	BOOL	Hier wird ein Abbruch der Bewegung signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Direction:** reserviert. Dieser Eingang wurde nur aus Kompatibilitätsgründen ergänzt und sollte nicht oder mit der Konstanten MC\_Shortest\_Way\_BkPlcMc belegt werden. (ab V3.0.8)

**BufferMode:** reserviert. Dieser Eingang wurde vorbereitend ergänzt und sollte derzeit nicht oder mit der Konstanten Aborting\_BkPlcMc belegt werden. (ab V3.0.8)

### Verhalten des Bausteins

Auf eine steigende Flanke an **JogForward** oder **JogBackwards** wird, abhängig vom verwendeten Mode [► 150] eine Bewegung gestartet.

- MC\_JOGMODE\_STANDARD\_SLOW: triggert ein MC\_MoveVelocity\_BkPlcMc [► 78] bei steigender Flanke und ein MC\_Stop\_BkPlcMc [► 81] bei fallender Flanke.
- MC\_JOGMODE\_STANDARD\_FAST: triggert ein MC\_MoveVelocity\_BkPlcMc bei steigender Flanke und ein MC\_Stop\_BkPlcMc bei fallender Flanke.
- MC\_JOGMODE\_CONTINUOUS: triggert ein MC\_MoveVelocity\_BkPlcMc bei steigender Flanke und ein MC\_Stop\_BkPlcMc bei fallender Flanke.
- MC\_JOGMODE\_INCHING: triggert ein MC\_MoveRelative\_BkPlcMc [► 76] bei steigender Flanke.

Die unterlagerten Bausteine überprüfen das übergeordnete Achsinterface und melden die Probleme am Ausgang **Error** und **ErrorID**

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten, wird die Bewegung initiiert.

Sollte während der Abarbeitung der Bewegung vom Bewegungsalgorithmus ein Fehlercode signalisiert werden, wird mit **Error** und **ErrorID**=Bewegungsalgorithmus-Fehlercode reagiert. Wird die vollständige Bewegung durch die Aktivität eines anderen Bausteins verhindert, wird mit **CommandAborted** reagiert. Erreicht der Bewegungsalgorithmus die Zielbedingungen der Achse, wird mit **Done** reagiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Bewegung auf FALSE gesetzt, wird die eingeleitete Bewegung unbeeinflusst weiter bearbeitet. Die Signale am Ende der Bewegung (**Error**, **ErrorID**, **CommandAborted**, **Done**) werden für einen Zyklus gegeben.

## 4.3 Datentypen

### 4.3.1 Axis\_Ref\_BkPlcMc (ab V3.0)

Die Variablen in dieser Struktur fassen die Unterbestandteile der Achse zusammen. Eine Variable dieses Typs wird an die meisten Bausteine der Bibliothek übergeben und somit entspricht dieser Typ dem AXIS\_REF Datentyp der PlcOpen.

#### Syntax

```

TYPE Axis_Ref_BkPlcMc:
STRUCT
  sAxisName:          STRING(83) := 'NoName';
  pStAxLogBuffer:     POINTER TO ST_TcMcLogBuffer:=0;
  pStDeviceInput:     POINTER TO ST_TcPlcDeviceInput:=0;
  pStDeviceOutput:    POINTER TO ST_TcPlcDeviceOutput:=0;
  pStAxAuxLabels:     POINTER TO ST_TcMcAuxDataLabels:=0;
  pStAxAutoParams:    POINTER TO ST_TcMcAutoIdent:=0;
  pStAxCommandBuf:    POINTER TO ST_TcPlcCmdBuffer_BkPlcMc:=0;
  nActiveRequest:     UDINT := 0;
  nNextRequest:       UDINT := 1;
  bParamsEnable:      BOOL:=FALSE;
  nState:             E_TcMCFbState:=McState_Standstill;
  nInitState:         INT:=0;
  nInitError:         DINT:=0;

```

```

nInterfaceType:   UINT := 16#FFFF;
nDeviceInType:   UINT := 16#FFFF;
nDeviceOutType:  UINT := 16#FFFF;
nRtDataType:    UINT := 16#FFFF;
nParamType:     UINT := 16#FFFF;
nLogBufferType:  UINT := 16#FFFF;
nAxAutoIdentType:  UINT := 16#FFFF;
nCmdBufferType:  UINT := 16#FFFF;
nLogLevel:      DINT := 0;
nDebugTag:      UDINT := 16#00000000;
stAxParams:     ST_TcHydAxParam;
stAxRtData:     ST_TcHydAxRtData;
END_STRUCT
END_TYPE
    
```

**Parameter**

Name	Typ	Beschreibung
sAxisName	STRING	Der Text-Name der Achse.
pStAxLogBuffer	POINTER TO ST_TcMcLogBuffer	Die Adresse einer Variablen vom Typ <u>ST_TcMcLogBuffer</u> [▶ 142]. Diese Variable enthält den LogBuffer der Library.
pStDeviceInput	POINTER TO ST_TcPlcDeviceInput	Die Adresse einer Variablen vom Typ <u>ST_TcPlcDeviceInput</u> [▶ 138]. Diese Variable enthält alle Eingangsinterfaces der Achse.
pStDeviceOutput	POINTER TO ST_TcPlcDeviceOutput	Die Adresse einer Variablen vom Typ <u>ST_TcPlcDeviceOutput</u> [▶ 140]. Diese Variable enthält alle Ausgangsinterfaces der Achse.
pStAxAuxLabels	POINTER TO ST_TcMcAuxDataLabels	Die Adresse einer Variable vom Typ <u>ST_TcMcAuxDataLabels</u> [▶ 137]. Diese Variable enthält optional die Bezeichnungen der Applikations-Parameter in <u>ST_TcHydAxParam:fCustomerData[..]</u> .
pStAxAutoParams	POINTER TO ST_TcMcAutoident	Die Adresse einer Variable vom Typ <u>ST_TcMcAutoident</u> [▶ 118]. Diese Variable enthält optional die Parameter für einen <u>MC_AxUtiAutoident_BkPlcMc</u> [▶ 254] Baustein.
pStAxCommandBuf	POINTER TO ST_TcPlcCmdBuffer_BkPlcMc	Ab V3.0.8 ist bei diversen Bausteinen der durch die PLCopen definierte Eingang <b>BufferMode</b> vorhanden. Die damit steuerbare Funktionalität wird derzeit vorbereitet. In diesem Zusammenhang ist dieser Befehls-Puffer ergänzt worden.
nActiveRequest	UDINT	Hier stellt jeder Baustein eine Kennzahl ein, der eine Funktion auf dieser Achse startet. Anschließend überwacht der Baustein diese Variable auf Veränderung durch einen anderen Baustein, der durch eine andere Funktion die Kontrolle übernimmt. So kann jeder Baustein feststellen, ob die von ihm gestartete Funktion durch einen anderen Baustein abgebrochen wurde und entsprechende Signale erzeugen.
nNextRequest	UDINT	Reserviert. Wird für die Erzeugung von neuen Werten für <b>nActiveRequest</b> benutzt.
bParamsEnable	BOOL	Diese Variable ist nur auf TRUE, wenn die Parameter durch Laden aus einer Datei in einen gültigen Zustand versetzt wurden. Ein Abspeichern der Parameter setzt ebenfalls dieses Signal, da die Übereinstimmung der Daten in der Parameterstruktur und in der Datei ebenfalls sichergestellt ist. Solange diese Variable nicht auf TRUE steht ist die Achse nicht betriebsbereit.  <b>Zur Laufzeit wird bei schreibenden Zugriffen auf die Parameterstruktur diese Variable temporär auf FALSE und anschließend wieder auf ihren Vorzustand gesetzt.</b>

Name	Typ	Beschreibung
nState	E_TcMCFbState	Hier ist in einer entsprechend E_TcMCFbState <a href="#">[▶ 96]</a> codierten Form der aktuelle Zustand der Achse hinterlegt.
nInitState	INT	Der aktuelle Zustand der Initialisierung.
nInitError	DINT	Ein eventuell beim Initialisieren festgestellte Fehlercode.
nInterfaceType	UINT	Der Typecode des derzeit gültigen Axis_Ref_BkPlcMc Variablentyps.
nDeviceInType	UINT	Der Typecode des derzeit gültigen ST_TcPlcDeviceInput <a href="#">[▶ 138]</a> Variablentyps.
nDeviceOutType	UINT	Der Typecode des derzeit gültigen ST_TcPlcDeviceOutput <a href="#">[▶ 140]</a> Variablentyps.
nRtDataType	UINT	Der Typecode des derzeit gültigen ST_TcHydAxRtData <a href="#">[▶ 131]</a> Variablentyps.
nParamType	UINT	Der Typecode des derzeit gültigen ST_TcHydAxParam <a href="#">[▶ 120]</a> Variablentyps.
nLogBufferType	UINT	Der Typecode des derzeit gültigen ST_TcMcLogBuffer <a href="#">[▶ 142]</a> Variablentyps.
nAxAutoIdentType	UINT	Der Typecode des derzeit gültigen ST_TcMcAutoIdent <a href="#">[▶ 118]</a> Variablentyps.
nCmdBufferType	UINT	reserviert. Der Typecode des derzeit gültigen Befehls-Puffer-Variablentyps.
nLogLevel	DINT	Der <a href="#">Message Level [▶ 328]</a> , ab dem ein Eintrag in den Logging Puffer erfolgen soll.
nDebugTag	UDINT	Viele Bausteine der Bibliothek tragen hier für die Dauer ihrer Durchführung eine Debug-Kennung ein.
stAxParams	ST_TcHydAxParam	Diese Variable vom Typ ST_TcHydAxParam <a href="#">[▶ 120]</a> enthält die Parameter der Achse.
stAxRtData	ST_TcHydAxRtData	Diese Variable vom Typ ST_TcHydAxRtData <a href="#">[▶ 131]</a> enthält die Laufzeitdaten der Achse.



Um die Datenstrukturen der Bibliothek unabhängiger von der Architektur der CPU (I86, Strong ARM) zu machen, ist es erforderlich, an einigen Stellen Daten in der Reihenfolge umzuordnen oder Platzhalter einzufügen. Diese Platzhalter erhalten einen Namen der Form "bAlign\_1" wobei die Zahlenangabe keinerlei Bedeutung besitzt. Weder Vorhandensein, Benennung, Typ oder Dimensionierung werden garantiert.

### 4.3.2 E\_TcPlcBufferedCmdType\_BkPlcMc

Die Konstanten in dieser Auflistung werden zur Kennzeichnung von bepufferten Achskommandos. Siehe hierzu [MC BufferMode BkPlcMc \[▶ 112\]](#).

#### Syntax

```

TYPE E_TcPlcBufferedCmdType_BkPlcMc : (
(* last modification: xx.xx.2009 *)
iBufferedCmd_NoOperation,
iBufferedCmd_MoveAbsolute,
iBufferedCmd_MoveRelative,
iBufferedCmd_MoveVelocity,
(**)
iBufferedCmd_Stop,
iBufferedCmd_ResetAndStop,
iBufferedCmd_Halt,
iBufferedCmd_CamIn,
iBufferedCmd_GearIn,
iBufferedCmd_Power,

```

```
iBufferedCmd_Home,
iBufferedCmd_StepAbsSwitch,
iBufferedCmd_StepLimitSwitch,
iBufferedCmd_StepBlock,
iBufferedCmd_StepDirect,
iBufferedCmd_FinishHoming,
(**)
iBufferedCmdEx_Jerk:=100,
iBufferedCmdEx_Acc,
iBufferedCmdEx_Velo,
iBufferedCmdEx_Creep,
(**)
iBufferedCmd_
);
END_TYPE
```

**Werte**

Name	Beschreibung
iBufferedCmd_NoOperation	Diese Konstante wird als Initial-Wert für Aufruf-Parameter von Funktionsbausteinen und in Variablen benutzt.
iBufferedCmd_MoveAbsolute	Das gepufferte Kommando ist durch einen MC_MoveAbsolute_BkPlcMc-Baustein eingetragen worden. Siehe Hinweis #1.
iBufferedCmd_MoveRelative	Das gepufferte Kommando ist durch einen MC_MoveRelative_BkPlcMc-Baustein eingetragen worden. Siehe Hinweis #1.
iBufferedCmd_MoveVelocity	Das gepufferte Kommando ist durch einen MC_MoveVelocity_BkPlcMc-Baustein eingetragen worden. Siehe Hinweis #1.
iBufferedCmd_Stop	reserviert, nicht implementiert.
iBufferedCmd_ResetAndStop	reserviert, nicht implementiert.
iBufferedCmd_Halt	reserviert, nicht implementiert.
iBufferedCmd_CamIn	reserviert, nicht implementiert.
iBufferedCmd_GearIn	reserviert, nicht implementiert.
iBufferedCmd_Power	reserviert, nicht implementiert.
iBufferedCmd_Home	reserviert, nicht implementiert.
iBufferedCmd_StepAbsSwitch	reserviert, nicht implementiert.
iBufferedCmd_StepLimitSwitch	reserviert, nicht implementiert.
iBufferedCmd_StepBlock	reserviert, nicht implementiert.
iBufferedCmd_StepDirect	reserviert, nicht implementiert.
iBufferedCmd_FinishHoming	reserviert, nicht implementiert.
iBufferedCmdEx_Jerk	Der mit konstantem Ruck gefahrene Anteil eines Kommandos ist durch einen Baustein eingetragen worden. Siehe Hinweis #2.
iBufferedCmdEx_Acc	Der mit konstanter Beschleunigung oder Verzögerung gefahrene Anteil eines Kommandos ist durch einen Baustein eingetragen worden. Siehe Hinweis #2.
iBufferedCmdEx_Velo	Der mit konstanter Geschwindigkeit gefahrene Anteil eines Kommandos ist durch einen Baustein eingetragen worden. Siehe Hinweis #2.
iBufferedCmdEx_Creep	reserviert, nicht implementiert.

**i** #1: Wenn die Achse einen Sollwertgenerator-Typ nutzt, der keinen Look Ahead realisiert werden vollständige Kommandos als ein Puffer-Element eingetragen.

**i** #2: Wenn die Achse einen Sollwertgenerator-Typ nutzt, der einen Look Ahead realisiert werden Kommandos in Teil-Abschnitte zerlegt und als Paket von typischerweise sieben Puffer-Elemente (Ruck, Beschleunigung, Ruck, Geschwindigkeit, Ruck, Verzögerung, Ruck) eingetragen.

### 4.3.3 E\_TcMcCurrentStep (ab V3.0)

Die Konstanten in dieser Auflistung werden für die Kennzeichnung der internen Zustände der Stellwertgeneratoren benutzt.



Nicht jeder dieser Zustände wird von allen Stellwertgeneratoren benutzt.

#### Syntax

```

TYPE E_TcMcCurrentStep : (
iTcHydStateIdle,
iTcHydStateTcAccP,
iTcHydStateTcAccM,
iTcHydStatePcAccP,
iTcHydStatePcAccM,
iTcHydStateConstVeloP,
iTcHydStateConstVeloM,
iTcHydStatePcDecP,
iTcHydStatePcDecM,
iTcHydStateCreepVeloP,
iTcHydStateCreepVeloM,
iTcHydStateTcDecP,
iTcHydStateTcDecM,
iTcHydStateFeedStopPos,
iTcHydStateFeedStopNeg,
iTcHydStateDoBrake,
iTcHydStateCoupling := 1000,
iTcHydStateCoupled,
iTcHydStateExtCoupled,
iTcHydStateExtGenerated := 2000,
iTcHydStateEmergencyBreak := 9000,
iTcHydStateFault := 9999
);
END_TYPE
    
```

#### Werte

Name	Beschreibung
iTcHydStateIdle	Die Achse führt keine aktive Fahrbewegung aus. Ihr Verhalten wird von ST_TcHydAxParam.fLagAmp, ST_TcHydAxParam.fTargetClamping und ST_TcHydAxParam.fReposDistance bestimmt.
iTcHydStateTcAccP	Die Achse baut entsprechend ST_TcHydAxRtData.fDestAcc einen positiven Stellwert auf. Dieser Wert wird durch einen der Startbausteine entsprechend den Daten des Fahrauftrags gesetzt. Erreicht der Stellwert den vorgesehenen Fahrstellwert wird in den Zustand gewechselt. Wird vorher festgestellt, dass der Bremsvorgang für die Zielfahrt eingeleitet werden muss, wird in den Zustand <b>iTcHydStatePcDecP</b> gewechselt. Bei Wegfall der Vorschubfreigabe wird in den Zustand <b>iTcHydStateFeedStopPos</b> gewechselt.
iTcHydStateTcAccM	Die Achse baut entsprechend ST_TcHydAxRtData.fDestAcc einen negativen Stellwert auf. Dieser Wert wird durch einen der Startbausteine entsprechend den Daten des Fahrauftrags gesetzt. Erreicht der Stellwert den vorgesehenen Fahrstellwert wird in den Zustand gewechselt. Wird vorher festgestellt, dass der Bremsvorgang für die Zielfahrt eingeleitet werden muss, wird in den Zustand <b>iTcHydStatePcDecM</b> gewechselt. Bei Wegfall der Vorschubfreigabe wird in den Zustand <b>iTcHydStateFeedStopNeg</b> gewechselt.
iTcHydStatePcAccP	Die Achse befindet sich in der weggesteuerten Beschleunigungsphase einer Fahrbewegung in positiver Richtung. Dabei wird der Stellwert entsprechend ST_TcHydAxRtData.fDestAcc auf einen vom Fahrauftrag festgelegten Wert aufgebaut. Anschließend wird in den Zustand <b>iTcHydStateConstVeloP</b> gewechselt.



Name	Beschreibung
iTcHydStatePcAccM	Die Achse befindet sich in der weggesteuerten Beschleunigungsphase einer Fahrbewegung in negativer Richtung. Dabei wird der Stellwert entsprechend ST_TcHydAxRtData.fDestAcc auf einen vom Fahrauftrag festgelegten Wert aufgebaut. Anschließend wird in den Zustand <b>iTcHydStateConstVeloM</b> gewechselt.
iTcHydStateConstVeloP	Die Achse fährt mit konstantem Stellwert in positiver Richtung. Der Stellwert wird durch den Fahrauftrag festgelegt.
iTcHydStateConstVeloM	Die Achse fährt mit konstantem Stellwert in negativer Richtung. Der Stellwert wird durch den Fahrauftrag festgelegt.
iTcHydStatePcDecP	Die Achse befindet sich in der weggesteuerten Bremsphase einer Fahrbewegung in positiver Richtung. Dabei wird der Stellwert bis auf ST_TcHydAxParam.fCreepSpeed abgebaut. Anschließend wird in den Zustand <b>iTcHydStateCreepVeloP</b> gewechselt.
iTcHydStatePcDecM	Die Achse befindet sich in der weggesteuerten Bremsphase einer Fahrbewegung in negativer Richtung. Dabei wird der Stellwert bis auf ST_TcHydAxParam.fCreepSpeed abgebaut. Anschließend wird in den Zustand <b>iTcHydStateCreepVeloM</b> gewechselt.
iTcHydStateCreepVeloP	Die Achse fährt mit konstantem Stellwert in positiver Richtung. Der Stellwert wird durch ST_TcHydAxParam.fCreepSpeed festgelegt.
iTcHydStateCreepVeloM	Die Achse fährt mit konstantem Stellwert in negativer Richtung. Der Stellwert wird durch ST_TcHydAxParam.fCreepSpeed festgelegt.
iTcHydStateTcDecP	Die Achse führt ausgehend von einer Fahrbewegung in positiver Richtung einen regulären Stopp aus. Dabei wird der Stellwert mit ST_TcHydAxParam.fStopRamp abgebaut. Anschließend wird in den Zustand <b>iTcHydStateIdle</b> gewechselt.
iTcHydStateTcDecM	Die Achse führt ausgehend von einer Fahrbewegung in negativer Richtung einen regulären Stopp aus. Dabei wird der Stellwert mit ST_TcHydAxParam.fStopRamp abgebaut. Anschließend wird in den Zustand <b>iTcHydStateIdle</b> gewechselt.
iTcHydStateFeedStopPos	Die Achse führt wegen nicht anstehender Vorschubfreigabe in positiver Richtung (in ST_TcHydAxRtData.nDeCtrlDWord ist dwTcHydDcDwFdPosEna nicht gesetzt) einen Zwischenstopp aus. Dabei wird der Stellwert mit ST_TcHydAxParam.fStopRamp abgebaut. Anschließend wird auf eine Vorschubfreigabe gewartet.
iTcHydStateFeedStopNeg	Die Achse führt wegen nicht anstehender Vorschubfreigabe in negativer Richtung (in ST_TcHydAxRtData.nDeCtrlDWord ist dwTcHydDcDwFdNegEna nicht gesetzt) einen Zwischenstopp aus. Dabei wird der Stellwert mit ST_TcHydAxParam.fStopRamp abgebaut. Anschließend wird auf eine Vorschubfreigabe gewartet.
iTcHydStateDoBrake	Die Achse führt eine Wartezeit aus. Dies ist erforderlich, wenn wegen einer Bremse oder eines Schaltventils geschaltet werden muss.
iTcHydStateCoupling	Die Achse befindet sich im Übergang in den Zustand <b>iTcHydStateCoupled</b> .
iTcHydStateCoupled	Der Stellwert der Achse wird nach dem Prinzip des elektronischen Getriebes vom Stellwert einer anderen Achse abgeleitet.
iTcHydStateExtCoupled	Der Stellwert der Achse wird nach dem Prinzip des stufenlosen Getriebes berechnet.
iTcHydStateExtGenerated	Der Stellwert der Achse wird durch einen externen Baustein erzeugt. Dabei kann es sich um einen Baustein der Library oder um einen applikationsspezifischen Baustein handeln.
iTcHydStateEmergencyBreak	Die Achse führt einen Notstopp aus. Dabei wird der Stellwert mit ST_TcHydAxParam.fEmergencyRamp abgebaut. Anschließend wird untersucht, ob sich die Achse in einem Störzustand (ST_TcHydAxRtData.nErrorCode ungleich 0) befindet. Wenn ja wird in den Zustand <b>iTcHydStateFault</b> gewechselt, andernfalls in den Zustand <b>iTcHydStateIdle</b> .

Name	Beschreibung
iTcHydStateFault	Die Achse ist in einem Störzustand. Sie führt keine aktiv kontrollierten Bewegungen aus und nimmt auch keine Bewegungsaufträge an. Um die Achse wieder in einen ungestörten Zustand zu versetzen ist ein Baustein des Typs MC_Reset_BkPlcMc oder MC_ResetAndStop_BkPlcMc aufzurufen.

### 4.3.4 E\_TcMcDriveType (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung von für die Stellwertausgabe einer Achse verwendeter Hardware benutzt.

#### Syntax

```

TYPE E_TcMcDriveType :(
  (*
  The sequence below must not be changed!
  New types have to be added at the end.
  In case a type becomes obsolete it has to be replaced by a dummy
  to ensure the numerical meaning of the other codes.
  *)
  (*
  Die bestehende Reihenfolge darf nicht veraendert werden.
  Neue Typen muessen am Ende eingefuegt werden.
  Wenn ein Typ wegfallen sollte, muss er durch einen Dummy
  ersetzt werden, um die numerische Zuordnung zu garantieren.
  *)
  (* last modification: 26.02.2016 *)
  iTcMc_Drive_Customized,
  iTcMc_DriveLowCostStepper,
  iTcMc_DriveKL2521,
  iTcMc_DriveKL4032,
  iTcMc_DriveAx2000_B900R,
  iTcMc_DriveM2400_D1,
  iTcMc_DriveM2400_D2,
  iTcMc_DriveM2400_D3,
  iTcMc_DriveM2400_D4,
  iTcMc_DriveLowCostStepperHS,
  iTcMc_DriveLowCostStepperFS,
  iTcMc_DriveIx2512_1Coil,
  iTcMc_DriveIx2512_2Coil,
  iTcMc_DriveKL2531,
  iTcMc_DriveKL2541,
  iTcMc_DriveEL4132,
  iTcMc_DriveAx2000_B200R,
  iTcMc_DriveAx2000_B110R,
  iTcMc_DriveKL2532,
  iTcMc_DriveKL2552,
  iTcMc_DriveKL2535_1Coil,
  iTcMc_DriveKL2535_2Coil,
  iTcMc_DriveKL2545_1Coil,
  iTcMc_DriveKL2545_2Coil,
  iTcMc_DriveLowCostInverter,
  iTcMc_Drive_CoE_DS408,
  iTcMc_DriveAx2000_B110A,
  iTcMc_DriveAx5000_B110A,
  iTcMc_DriveAx2000_B750A,
  iTcMc_Drive_CoE_DS402,
  iTcMc_DriveAx5000_B110SR,
  iTcMc_DriveEL4x22,
  iTcMc_DriveEL2521,
  iTcMc_DrivePumpEtcIO,
  iTcMc_DriveEL2535_1Coil,
  iTcMc_DriveEL2535_2Coil,
  iTcMc_DriveEL7201,
  iTcMc_DriveEL7037,
  iTcMc_DriveEL7047,
  iTcMc_DriveEM8908,
  iTcMc_DriveAx5000_B110INC,
  iTcMc_Drive_TestOnly:=1000
);
END_TYPE

```

**Werte**

Name	Beschreibung
iTcMc_Drive_Customized	iTcMc_Drive_Customized: Der Stellwert für den Antrieb wird nicht für die Ausgabe auf einer bestimmten Hardware aufbereitet. Diese Aufbereitung ist durch die PLC-Applikation selbst vorzunehmen.
iTcMc_DriveLowCostStepper	iTcMc_DriveLowCostStepper: Die inkrementelle Sollpositionsänderungen werden als digitale Ausgangssignale für einen direkt angesteuerten Schrittmotor erzeugt. Dies ist ein aus Gründen der Kompatibilität weiterhin unterstützter Code und ist gleichbedeutend mit iTcMc_DriveLowCostStepperHS.
iTcMc_DriveKL2521	iTcMc_DriveKL2521: Der Stellwert für den Antrieb wird für die Ausgabe auf eine Pulse Train Klemme KL2521 aufbereitet.
iTcMc_DriveKL4032	iTcMc_DriveKL4032: Der Stellwert für den Antrieb wird für die Ausgabe auf einer Analog-Ausgangsklemme ±10 V KL4032 aufbereitet.
iTcMc_DriveAx2000_B900R	iTcMc_DriveAx2000_B900R: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX2000 Stellgerät an einem Resolver-Motor am Beckhoff RealTime Ethernet Feldbus aufbereitet.
iTcMc_DriveM2400_D1	iTcMc_DriveM2400_D1: Der Stellwert für den Antrieb wird für die Ausgabe auf dem ersten Kanal einer M2400 Box am Beckhoff II/O aufbereitet.
iTcMc_DriveM2400_D2	iTcMc_DriveM2400_D2: Der Stellwert für den Antrieb wird für die Ausgabe auf dem zweiten Kanal einer M2400 Box am Beckhoff II/O aufbereitet.
iTcMc_DriveM2400_D3	iTcMc_DriveM2400_D3: Der Stellwert für den Antrieb wird für die Ausgabe auf dem dritten Kanal einer M2400 Box am Beckhoff II/O aufbereitet.
iTcMc_DriveM2400_D4	iTcMc_DriveM2400_D4: Der Stellwert für den Antrieb wird für die Ausgabe auf dem vierten Kanal einer M2400 Box am Beckhoff II/O aufbereitet.
iTcMc_DriveLowCostStepperHS	iTcMc_DriveLowCostStepperHS: Die inkrementelle Sollpositionsänderungen werden als digitale Ausgangssignale für einen direkt angesteuerten Schrittmotor erzeugt. Es wird das Halbschritt-Verfahren angewendet.
iTcMc_DriveLowCostStepperFS	iTcMc_DriveLowCostStepperFS: Die inkrementelle Sollpositionsänderungen werden als digitale Ausgangssignale für einen direkt angesteuerten Schrittmotor erzeugt. Es wird das Vollschritt-Verfahren angewendet.
iTcMc_DriveIx2512_1Coil	iTcMc_DriveIx2512_1Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf einer Feldbus Box IP/IE2512 aufbereitet. Es gelten die Regeln für Ventile mit einer Spule.
iTcMc_DriveIx2512_2Coil	iTcMc_DriveIx2512_2Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf einer Feldbus Box IP/IE2512 aufbereitet. Es gelten die Regeln für Ventile mit zwei Spulen.
iTcMc_DriveKL2531	iTcMc_DriveKL2531: Der Stellwert für den Antrieb wird für die Ausgabe auf eine Schrittmotorendstufenklemme KL2531 aufbereitet.
iTcMc_DriveKL2541	iTcMc_DriveKL2541: Der Stellwert für den Antrieb wird für die Ausgabe auf eine Schrittmotorendstufenklemme KL2541 aufbereitet.
iTcMc_DriveEL4132	iTcMc_DriveEL4132: Der Stellwert für den Antrieb wird für die Ausgabe auf einer Analog-Ausgangsklemme ±10 V EL4132 aufbereitet.
iTcMc_DriveAx2000_B200R	iTcMc_DriveAx2000_B200R: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX2000 Stellgerät an einem Resolver-Motor am Beckhoff II/O Feldbus aufbereitet.

Name	Beschreibung
iTcMc_DriveAx2000_B110R	iTcMc_DriveAx2000_B110R: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX2000 Stellgerät an einem Resolver-Motor am EtherCAT Feldbus aufbereitet.
iTcMc_DriveKL2532	iTcMc_DriveKL2532: Der Stellwert für den Antrieb wird für die Ausgabe auf eine DC-Motorendstufenklemme KL2532 aufbereitet.
iTcMc_DriveKL2552	
iTcMc_DriveKL2535_1Coil	iTcMc_DriveKL2535_1Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf eine PWM-Endstufenklemme KL2535 aufbereitet.
iTcMc_DriveKL2535_2Coil	iTcMc_DriveKL2535_2Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf eine PWM-Endstufenklemme KL2535 aufbereitet.
iTcMc_DriveKL2545_1Coil	iTcMc_DriveKL2545_1Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf eine PWM-Endstufenklemme KL2545 aufbereitet.
iTcMc_DriveKL2545_2Coil	iTcMc_DriveKL2545_2Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf eine PWM-Endstufenklemme KL2545 aufbereitet.
iTcMc_DriveLowCostInverter	iTcMc_DriveLowCostInverter: Der Stellwert für den Antrieb wird für die Ausgabe als digitale Ausgangssignale für einen Frequenzumrichter mit programmierten Festfrequenzen aufbereitet.
iTcMc_Drive_CoE_DS408	iTcMc_Drive_CoE_DS408: Der Stellwert für den Antrieb wird für die Ausgabe auf einem Proportionalventil am EtherCAT Feldbus aufbereitet.
iTcMc_DriveAx2000_B110A	iTcMc_DriveAx2000_B110A: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX2000 Stellgerät an einem Absolut-Multiturnencoder-Motor am EtherCAT Feldbus aufbereitet.
iTcMc_DriveAx5000_B110A	iTcMc_DriveAx5000_B110A: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX5000 Stellgerät an einem Absolut-Multiturnencoder-Motor am EtherCAT Feldbus aufbereitet.
iTcMc_DriveAx2000_B750A	iTcMc_DriveAx2000_B750A: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX2000 Stellgerät an einem Absolut-Multiturnencoder-Motor am Sercos Feldbus aufbereitet.
iTcMc_Drive_CoE_DS402	iTcMc_Drive_CoE_DS402: In Vorbereitung.
iTcMc_DriveAx5000_B110SR	iTcMc_DriveAx5000_B110SR: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX5000 Stellgerät an einem Absolut-Singleturnencoder- oder Resolver-Motor am EtherCAT Feldbus aufbereitet.
iTcMc_DriveEL4x22	iTcMc_DriveEL4x22: In Vorbereitung.
iTcMc_DriveEL2521	iTcMc_DriveEL2521: Der Stellwert für den Antrieb wird für die Ausgabe auf eine Pulse Train Klemme KL2521 aufbereitet.
iTcMc_DrivePumpEtcIO	iTcMc_DrivePumpEtcIO: reserviert
iTcMc_DriveEL2535_1Coil	iTcMc_DriveEL2535_1Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf eine PWM-Endstufenklemme EL2535 aufbereitet.
iTcMc_DriveEL2535_2Coil	iTcMc_DriveEL2535_2Coil: Der Stellwert für den Antrieb wird für die Ausgabe auf eine PWM-Endstufenklemme EL2535 aufbereitet.
iTcMc_DriveEL7201	iTcMc_DriveEL7201: Der Stellwert für den Antrieb wird für die Ausgabe auf einer Servo-Klemme EL7201 aufbereitet.
iTcMc_DriveEL7037	iTcMc_DriveEL7037: Der Stellwert für den Antrieb wird für die Ausgabe auf eine Schrittmotorendstufenklemme EL7037 aufbereitet.
iTcMc_DriveEL7047	iTcMc_DriveEL7047: Der Stellwert für den Antrieb wird für die Ausgabe auf eine Schrittmotorendstufenklemme EL7047 aufbereitet.
iTcMc_DriveEM8908	iTcMc_DriveEM8908: Reserviert für Branchenpaket.
iTcMc_DriveAx5000_B110INC	iTcMc_DriveAx5000_B110INC: Der Stellwert für den Antrieb wird für die Ausgabe auf einem AX5000 Stellgerät an einem Inkremental-Encoder am EtherCAT Feldbus aufbereitet.
iTcMc_Drive_TestOnly	iTcMc_Drive_TestOnly: reserviert für interne Tests; nicht benutzen.

**iTcMc\_DriveEL7031:** Der Stellwert für den Antrieb wird für die Ausgabe auf eine Schrittmotorendstufenklemme EL7031 aufbereitet.

**iTcMc\_DriveEL7041:** Der Stellwert für den Antrieb wird für die Ausgabe auf eine Schrittmotorendstufenklemme EL7041 aufbereitet.

**iTcMc\_DriveKL2542:** Der Stellwert für den Antrieb wird für die Ausgabe auf eine DC-Motorendstufenklemme KL2542 aufbereitet.

### 4.3.5 E\_TcMcEncoderType (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung von für die Istwerterfassung einer Achse verwendeter Hardware benutzt.

#### Syntax

```

TYPE E_TcMcEncoderType : (
  (*
  The sequence below must not be changed!
  New types have to be added at the end.
  In case a type becomes obsolete it has to be replaced by a dummy
  to ensure the numerical meaning of the other codes.
  *)
  (*
  Die bestehende Reihenfolge darf nicht veraendert werden.
  Neue Typen muessen am Ende eingefuegt werden.
  Wenn ein Typ wegfallen sollte, muss er durch einen Dummy
  ersetzt werden, um die numerische Zuordnung zu garantieren.
  *)
  (* last modification: 17.01.2013 *)
  iTcMc_EncoderSim,
  iTcMc_EncoderDigIncrement,
  iTcMc_EncoderLowCostStepper,
  iTcMc_EncoderKL2521,
  iTcMc_EncoderKL3042,
  iTcMc_EncoderKL5001,
  iTcMc_EncoderKL5101,
  iTcMc_EncoderAx2000_B900R,
  iTcMc_EncoderDigCam,
  iTcMc_EncoderIx5009,
  iTcMc_EncoderM2510,
  iTcMc_EncoderKL3002,
  iTcMc_EncoderKL2531,
  iTcMc_EncoderKL5111,
  iTcMc_EncoderAbs32,
  iTcMc_EncoderM3120,
  iTcMc_EncoderKL2541,
  iTcMc_EncoderEL3102,
  iTcMc_EncoderEL3142,
  iTcMc_EncoderEL5001,
  iTcMc_EncoderEL5101,
  iTcMc_EncoderEL5111,
  iTcMc_EncoderKL3062,
  iTcMc_EncoderKL3162,
  iTcMc_EncoderAx2000_B200R,
  iTcMc_EncoderAx2000_B110R,
  iTcMc_EncoderEL3162,
  iTcMc_EncoderKL2542,
  iTcMc_EncoderKL2545,
  iTcMc_EncoderAx2000_B110A,
  iTcMc_EncoderAx5000_B110A,
  iTcMc_EncoderAx2000_B750A,
  iTcMc_EncoderCoE_DS406,
  iTcMc_EncoderCoE_DS402SR,
  iTcMc_EncoderAx5000_B110SR,
  iTcMc_EncoderCoE_DS402A,
  iTcMc_EncoderEL2521,
  iTcMc_EncoderAbs32Etc,
  iTcMc_EncoderEL7201SR,
  iTcMc_EncoderDigPulseCount,
  iTcMc_EncoderEL3255,
  iTcMc_EncoderEL7047,
  iTcMc_DriveEM8908A,
  iTcMc_DriveEM8908C,
  iTcMc_EncoderCoE5001,
  iTcMc_EncoderEL7201A,

```

```

iTcMc_DriveAx5000_B110INC,
iTcMc_EncoderEL5032,
iTcMc_EncoderEL5021,
iTcMc_Encoder_TestOnly:=1000
);
END_TYPE

```

## Werte

Name	Beschreibung
iTcMc_EncoderSim	Die virtuelle Istposition der Achse ist eine Kopie der Sollposition. <b>Dieser Typ darf an einer realen Maschine nur bei virtuellen Achsen eingestellt sein. Andernfalls wird die Achse unkontrollierte und nicht vorhersagbare Bewegungen ausführen.</b>
iTcMc_EncoderDigIncrement	Der inkrementelle Istwert der Achse wird durch Auswertung von zwei digitalen Eingangsbits gebildet. Diese stellen die A- und B-Spur eines Inkremental-Encoders dar und werden entsprechend dem Prinzip des Quadratur-Decoders mit Vierfachauswertung ausgewertet. <b>Es darf pro PLC-Zyklus nur eines der Eingangsbits seinen Zustand ändern. Die maximale Geschwindigkeit beträgt also 1 Inkrement pro TCycle.</b>
iTcMc_EncoderLowCostStepper	Die inkrementelle Istposition wird aus den Ausgangssignalen für einen digital angesteuerten Schrittmotor erzeugt.
iTcMc_EncoderKL2521	Die inkrementelle Istposition wird aus dem Impulszähler einer Pulse Train Klemme KL2521 erzeugt.
iTcMc_EncoderKL3042	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsklemme 0..20 mA KL3042 erzeugt.
iTcMc_EncoderKL5001	Die absolute Istposition wird aus dem Zähler-Wert einer SSI-Eingangsklemme KL5001 erzeugt.
iTcMc_EncoderKL5101	Die inkrementelle Istposition wird aus dem Zähler-Wert einer Eingangsklemme KL5101 erzeugt.
iTcMc_EncoderAx2000_B900R	Die inkrementelle Istposition wird aus dem Zähler-Wert eines AX2000 Stellgeräts an einem Resolver-Motor am Beckhoff RealTime Ethernet Feldbus ermittelt.
iTcMc_EncoderDigCam	Das Positionsnocken-Byte wird aus vier digitalen Eingangsbits gebildet.
iTcMc_EncoderIx5009	Die absolute Istposition wird aus dem Zähler-Wert einer SSI-Feldbusbox IP/IE5009 erzeugt.
iTcMc_EncoderM2510	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsbox ±10 V M2510 erzeugt.
iTcMc_EncoderKL3002	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsklemme ±10V KL3002 erzeugt.
iTcMc_EncoderKL2531	Die inkrementelle Istposition wird aus dem Impulszähler einer Schrittmotor Klemme KL2531 erzeugt.
iTcMc_EncoderKL5111	Die inkrementelle Istposition wird aus dem Zähler-Wert einer Eingangsklemme KL5111 erzeugt.
iTcMc_EncoderAbs32	Die absolute Istposition wird aus dem 32-Bit-Wert einer allgemeinen Eingangselektronik erzeugt.
iTcMc_EncoderM3120	Die inkrementelle Istposition wird aus dem Zähler-Wert eines M3120 Lightbus Moduls erzeugt.
iTcMc_EncoderKL2541	Die inkrementelle Istposition wird aus dem Impulszähler (Motorpulse oder Encoder) einer Schrittmotor Klemme KL2541 erzeugt.
iTcMc_EncoderEL3102	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsklemme ±10V EL3102 erzeugt.

Name	Beschreibung
iTcMc_EncoderEL3142	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsklemme 0..20 mA EL3142 erzeugt.
iTcMc_EncoderEL5001	Die absolute Istposition wird aus dem Zähler-Wert einer SSI-Eingangsklemme EL5001 erzeugt.
iTcMc_EncoderEL5101	Die inkrementelle Istposition wird aus dem Zähler-Wert einer Eingangsklemme EL5101 erzeugt.
iTcMc_EncoderEL5111	Die inkrementelle Istposition wird aus dem Zähler-Wert einer Eingangsklemme EL5111 erzeugt.
iTcMc_EncoderKL3062	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsklemme 0..10 V KL3062 erzeugt.
iTcMc_EncoderKL3162	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsklemme 0..10 V KL3162 erzeugt.
iTcMc_EncoderAx2000_B200R	Die inkrementelle Istposition wird aus dem Zähler-Wert eines AX2000 Stellgeräts an einem Resolver-Motor am Beckhoff II/O Feldbus ermittelt.
iTcMc_EncoderAx2000_B110R	Die inkrementelle Istposition wird aus dem Zähler-Wert eines AX2000 Stellgeräts an einem Resolver-Motor am EtherCAT Feldbus ermittelt.
iTcMc_EncoderEL3162	Die absolute Istposition wird aus dem ADW-Wert einer Analog-Eingangsklemme 0..10 V EL3162 erzeugt.
iTcMc_EncoderKL2542	Die inkrementelle Istposition wird aus dem Zähler-Wert einer Eingangsklemme KL2542 erzeugt.
iTcMc_EncoderKL2545	Die inkrementelle Istposition wird aus dem Zähler-Wert einer Eingangsklemme KL2545 erzeugt.
iTcMc_EncoderAx2000_B110A	Die absolute Istposition wird aus dem Zähler-Wert eines AX2000 Stellgeräts an einem Absolut-Multiturnencoder-Motor am EtherCAT Feldbus ermittelt.
iTcMc_EncoderAx5000_B110A	Die absolute Istposition wird aus dem Zähler-Wert eines AX5000 Stellgeräts an einem Absolut-Multiturnencoder-Motor am EtherCAT Feldbus ermittelt.
iTcMc_EncoderAx2000_B750A	Die absolute Istposition wird aus dem Zähler-Wert eines AX2000 Stellgeräts an einem Absolut-Multiturnencoder-Motor am Sercos Feldbus ermittelt.
iTcMc_EncoderCoE_DS406	Ein Encoder mit CoE_406 Unterstützung am EtherCAT Feldbus.
iTcMc_EncoderCoE_DS402SR	In Vorbereitung.
iTcMc_EncoderAx5000_B110SR	Die inkrementelle Istposition wird aus dem Zähler-Wert eines AX5000 Stellgeräts an einem Singleturnencoder-Motor am EtherCAT Feldbus ermittelt.
iTcMc_EncoderCoE_DS402A	In Vorbereitung.
iTcMc_EncoderEL2521	Die inkrementelle Istposition wird aus dem Impulszähler einer Pulse Train Klemme EL2521 erzeugt.
iTcMc_EncoderAbs32Etc	Die absolute Istposition wird aus dem 32-Bit-Wert einer allgemeinen EtherCAT Eingangselektronik erzeugt. Es wird keine Profil-Unterstützung vorausgesetzt.
iTcMc_EncoderEL7201SR	Die inkrementelle Istposition wird aus dem Zähler-Wert einer Servo-Klemme EL7201 erzeugt.
iTcMc_EncoderDigPulseCount	Zählt die Flanken (positive und negative) von Pulsen. Die Drehrichtung wird über die Ausgabe des Drives bestimmt. <b>Es kann in einem PLC-Zyklus nur ein Puls erkannt werden.</b>
iTcMc_EncoderEL3255	iTcMc_EncoderEL3255: In Vorbereitung.
iTcMc_EncoderEL7047	
iTcMc_DriveEM8908A	
iTcMc_DriveEM8908C	
iTcMc_EncoderCoE5001	

Name	Beschreibung
iTcMc_EncoderEL7201A	
iTcMc_DriveAx5000_B110INC	Die inkrementelle Istposition wird aus dem Zähler-Wert eines AX5000 Stellgeräts an einem Inkremental-Encoder am EtherCAT Feldbus ermittelt.
iTcMc_EncoderEL5032	Die absolute Istposition wird aus dem Zähler-Wert einer EnDat-2.2 Eingangsklemme EL5032 erzeugt.
iTcMc_EncoderEL5021	Die absolute Istposition wird aus dem Zähler-Wert einer Sin/Cos-Eingangsklemme EL5021 erzeugt.
iTcMc_Encoder_TestOnly	reserviert für interne Tests; nicht benutzen.

**iTcMc\_EncoderEL7041:** Die inkrementelle Istposition wird aus dem Impulszähler (Motorpulse oder Encoder) einer Schrittmotor Klemme EL7041 erzeugt.

### 4.3.6 E\_TcMCFbState (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung der Laufzeitzustände von Achsen benutzt.

#### Syntax

```
TYPE E_TcMCFbState :
(
McState_Standstill := 0,
McState_DiscreteMotion,
McState_Continuousmotion,
McState_Synchronizedmotion,
McState_Stopping,
McState_Errorstop,
McState_Homing,
McState_Disabled
);
END_TYPE
```

#### Werte

Name	Beschreibung
McState_Standstill	Die Achse hat keinen Fahrauftrag. Je nach Parametrierung erfolgt eine aktive Lageregelung, eine Repositionier-Überwachung, die Ausgabe eines Press-Stellwerts oder nichts davon.
McState_DiscreteMotion	Die Achse führt eine Bewegung mit einer definierten Zielposition und Geschwindigkeit aus.
McState_Continuousmotion	Die Achse führt eine Bewegung ohne definierte Zielposition aus. Lediglich die Geschwindigkeit ist vorgegeben.
McState_Synchronizedmotion	Die Achse führt eine Bewegung aus, die von der Bewegung einer anderen Achse abgeleitet ist.
McState_Stopping	Die Achse führt eine Stop-Operation aus.
McState_Errorstop	Die Achse wurde wegen eines Problems gestoppt. Sie ist derzeit nicht startbar und muss durch einen Reset wieder in einen startbereiten Zustand versetzt werden.
McState_Homing	Die Achse führt eine Referenzfahrt aus.
McState_Disabled	Die Reglerfreigabe der Achse ist FALSE.

### 4.3.7 E\_TcMcHomingType (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung der Referenziermethode von Achsen benutzt.

#### Syntax

```
TYPE E_TcMcHomingType : (
iTcMc_HomingOnBlock,
iTcMc_HomingOnIndex,
```



```
iTcMc_HomingOnSync,
iTcMc_HomingOnMultiSync,
iTcMc_HomingOnExec
);
END_TYPE
```

**Werte**

Name	Beschreibung
iTcMc_HomingOnBlock	Die Achse wird mit ST_TcHydAxParam.fEnc_RefIndexVelo in der durch ST_TcHydAxParam.bEnc_RefIndexPositive festgelegten Richtung bewegt. Wird für eine Zeitdauer von 2 Sekunden keine Bewegung festgestellt gilt der Festanschlag (Block) als erreicht. Der Istwert der Achse wird auf den Wert der Referenzposition gesetzt.
iTcMc_HomingOnIndex	Die Achse wird mit ST_TcHydAxParam.fEnc_RefIndexVelo in der durch ST_TcHydAxParam.bEnc_RefIndexPositive festgelegten Richtung bewegt. Wird in ST_TcHydAxRtData.nDeCtrlDWord die Referenznocke (Bit 5, dwTcHydDcDwRefIndex) erkannt stoppt die Achse. Anschließend wird sie mit ST_TcHydAxParam.fEnc_RefSyncVelo in der durch ST_TcHydAxParam.bEnc_RefSyncPositive festgelegten Richtung bewegt, bis die Referenznocke wieder verlassen wird. Der Istwert der Achse wird auf den Wert der Referenzposition gesetzt.
iTcMc_HomingOnSync	Die Achse wird mit ST_TcHydAxParam.fEnc_RefIndexVelo in der durch ST_TcHydAxParam.bEnc_RefIndexPositive festgelegten Richtung bewegt. Wird in ST_TcHydAxRtData.nDeCtrlDWord die Referenznocke (Bit 5, dwTcHydDcDwRefIndex) erkannt stoppt die Achse. Anschließend wird sie mit ST_TcHydAxParam.fEnc_RefSyncVelo in der durch ST_TcHydAxParam.bEnc_RefSyncPositive festgelegten Richtung bewegt, bis die Referenznocke wieder verlassen wird. Dann wird das Hardware-Latch des Encoders aktiviert und die Achse so lange weiter bewegt, bis das Latch gültig wird. Nach dem Stoppen der Achse wird der Istwert der Achse auf einen Wert gesetzt, der aus der Referenzposition und der seit dem Sync-Puls des Encoders zurückgelegten Strecke errechnet wird.
iTcMc_HomingOnMultiSync	Der Istwert der Achse wird unmittelbar auf den Wert der Referenzposition gesetzt.
iTcMc_HomingOnExec	Das Hardware-Latch des Encoders wird aktiviert. Die Achse wird mit ST_TcHydAxParam.fEnc_RefSyncVelo in der durch ST_TcHydAxParam.bEnc_RefIndexPositive festgelegten Richtung so lange bewegt, bis das Latch zweimal gültig geworden ist. Wird das Ende des Fahrwegs erkannt bevor zwei Sync-Pulse erkannt wurden, wird der Vorgang in entgegengesetzter Richtung wiederholt. Führt auch dies nicht zum Erfolg wird die Referenzfahrt abgebrochen. Andernfalls wird aus dem Abstand der Sync-Pulse und dem fEnc_BaseDistance die aktuelle Istposition bestimmt.

**4.3.8 E\_TcMCPParameter (ab V3.0)**

Die Konstanten in dieser Auflistung werden zur Parameternummerierung benutzt.

**Syntax**

```
TYPE E_TcMCPParameter :
(
(*
=====
A T T E N T I O N
=====
= These Codes are also used to identify parameters in files.
= Any change of the meaning of any code here will make any file
= incompatible without notice and may even cause a crash of
= the control system!
=====
= CONSEQUENCE: Only adding new codes is allowed!
=====
```

```

=====
= These codes are also used for ADS communication
=====
*)
( *
=====
A C H T U N G
=====
= Diese Codes werden auch zur Kennzeichnung von Parametern
= in den Dateien verwendet. Eine Veraenderung der Codes wuerde
= die Dateien (nicht erkennbar) inkompatibel machen und koennte
= zum Systemabsturz fuehren!
=====
= ALSO: Es duerfen nur neue Codes dazugefuegt werden!
=====
= Diese Codes werden ebenfalls fuer die ADS-Kommunikation benutzt
=====
*)
McPara_CommandedPosition:=1,
McPara_SWLimitPos,
McPara_SWLimitNeg,
McPara_EnableLimitPos,
McPara_EnableLimitNeg,
McPara_EnablePosLagMonitoring,
McPara_MaxPositionLag,
McPara_MaxVelocitySystem,
McPara_MaxVelocityAppl,
McPara_ActualVelocity,
McPara_CommandedVelocity,
McPara_MaxAccelerationSystem,
McPara_MaxAccelerationAppl,
McPara_MaxDecelerationSystem,
McPara_MaxDecelerationAppl,
McPara_MaxJerk,
( * ===== *)
McPara_BkPlcMc_ProfilType:=1000,
McPara_BkPlcMc_EnvCycletime,
McPara_BkPlcMc_AxName,
McPara_BkPlcMc_TimeBased,
McPara_BkPlcMc_JerkEnabled,
McPara_BkPlcMc_LogLevel,
McPara_BkPlcMc_CycleDivider,
McPara_BkPlcMc_ParamFileName,

McPara_BkPlcMc_EncoderType:=1100,
McPara_BkPlcMc_EncoderHomingType,
McPara_BkPlcMc_EncoderZeroShift,
McPara_BkPlcMc_EncoderIncWeighting,
McPara_BkPlcMc_EncoderIncInterpolation,
McPara_BkPlcMc_EncoderRefIndexVelo,
McPara_BkPlcMc_EncoderRefIndexPositive,
McPara_BkPlcMc_EncoderRefSyncVelo,
McPara_BkPlcMc_EncoderRefSyncPositive,
McPara_BkPlcMc_EncoderDefaultHomePosition,
McPara_BkPlcMc_EncoderReversed,
McPara_BkPlcMc_EncoderBaseDistance,
McPara_BkPlcMc_EncoderModuloBase,
McPara_BkPlcMc_EncoderEnableLatch,
McPara_BkPlcMc_EncoderLatchedPos,
McPara_BkPlcMc_EncoderRefShift,
McPara_BkPlcMc_EncoderRefFlag,
McPara_BkPlcMc_EncoderPotiRgToRl,
McPara_BkPlcMc_EncoderOverrunMask,
McPara_BkPlcMc_EncoderPositionMask,
McPara_BkPlcMc_EncoderZeroSwap,
McPara_BkPlcMc_EncoderNoUpload,
McPara_BkPlcMc_EncoderModuloMode,

McPara_BkPlcMc_ValveOverlapCompP:=1200,
McPara_BkPlcMc_ValveBendPointVelo,
McPara_BkPlcMc_ValveBendPointOutput,
McPara_BkPlcMc_ValveResponseTime,
McPara_BkPlcMc_ValveOverlapCompM,
McPara_BkPlcMc_CylinderArreaA:=1280,
McPara_BkPlcMc_CylinderArreaB,

McPara_BkPlcMc_DriveType:=1300,
McPara_BkPlcMc_AreaRatio,
McPara_BkPlcMc_DriveReversed,
McPara_BkPlcMc_DriveDefaultPowerOk
McPara_BkPlcMc_DriveAbsoluteOutput,

```

```

McPara_BkPlcMc_DriveIncWeighting,
McPara_BkPlcMc_DriveIncInterpolation,
McPara_BkPlcMc_DriveNoUpload,

McPara_BkPlcMc_DriveIsHybrid,
McPara_BkPlcMc_HybridConcept,
McPara_BkPlcMc_Pump_Cavities,
McPara_BkPlcMc_Pump_EncType,
McPara_BkPlcMc_Pump_N_max,
McPara_BkPlcMc_Pump_N_min,
McPara_BkPlcMc_Pump_P_max,
McPara_BkPlcMc_Pump_P_min,
McPara_BkPlcMc_Pump_Q_fast_P,
McPara_BkPlcMc_Pump_Q_slow_P,
McPara_BkPlcMc_Pump_Q_fast_M,
McPara_BkPlcMc_Pump_Q_slow_M,
McPara_BkPlcMc_Pump_Q_leak,
McPara_BkPlcMc_Pump_Enc_Offset,
McPara_BkPlcMc_Cylinder_A_addP,
McPara_BkPlcMc_Cylinder_A_addM,
McPara_BkPlcMc_PrscScaling_A,
McPara_BkPlcMc_PrscScaling_B,
McPara_BkPlcMc_PrscScaling_Sys,
McPara_BkPlcMc_Motor_RampTime,
McPara_BkPlcMc_Pump_Regenerative,
McPara_BkPlcMc_Virtual_A_addP,
McPara_BkPlcMc_Virtual_A_addM,
McPara_BkPlcMc_Aside_PrscHiResADC,
McPara_BkPlcMc_Bside_PrscHiResADC,
McPara_BkPlcMc_System_PrscHiResADC,

McPara_BkPlcMc_StartRamp:=1400,
McPara_BkPlcMc_obsolete_1,
McPara_BkPlcMc_obsolete_2,

McPara_BkPlcMc_StopRamp:=1500,
McPara_BkPlcMc_EmergencyRamp,
McPara_BkPlcMc_BrakeOn,
McPara_BkPlcMc_BrakeOff,
McPara_BkPlcMc_BrakeSafety,

McPara_BkPlcMc_CreepSpeedP:=1600,
McPara_BkPlcMc_CreepDistanceP,
McPara_BkPlcMc_BrakeDistanceP,
McPara_BkPlcMc_BrakeDeadTimeP,
McPara_BkPlcMc_CreepSpeedM,
McPara_BkPlcMc_CreepDistanceM,
McPara_BkPlcMc_BrakeDistanceM,
McPara_BkPlcMc_BrakeDeadTimeM,
McPara_BkPlcMc_AsymmetricalTargeting,

McPara_BkPlcMc_LagAmp:=1700,
McPara_BkPlcMc_LagAmpAdaptLimit,
McPara_BkPlcMc_LagAmpAdaptFactor,
McPara_BkPlcMc_ZeroCompensation,
McPara_BkPlcMc_TargetClamping,
McPara_BkPlcMc_ReposDistance,
McPara_BkPlcMc_AutoBrakeDistance,
McPara_BkPlcMc_EnableControlLoopOnFault,
McPara_BkPlcMc_LagAmpDx,
McPara_BkPlcMc_LagAmpTi,
McPara_BkPlcMc_LagAmpWuLimit,
McPara_BkPlcMc_LagAmpOutLimit,

McPara_BkPlcMc_VeloAmp,
McPara_BkPlcMc_VeloAmpDx,
McPara_BkPlcMc_VeloAmpTi,
McPara_BkPlcMc_VeloAmpWuLimit,
McPara_BkPlcMc_VeloAmpOutLimit,
McPara_BkPlcMc_FeedForward,

McPara_BkPlcMc_LagAmpTd,
McPara_BkPlcMc_LagAmpTdd,
McPara_BkPlcMc_LagAmpCfb_tA,
McPara_BkPlcMc_LagAmpCfb_kA,
McPara_BkPlcMc_LagAmpCfb_tV,
McPara_BkPlcMc_LagAmpCfb_kV,
McPara_BkPlcMc_LagCtrlType,
McPara_BkPlcMc_LagAmpCfb_tF,
McPara_BkPlcMc_LagAmpCfb_kF,

```

```

McPara_BkPlcMc_AccFeedForward,

McPara_BkPlcMc_Pctrl_kP:=1780,
McPara_BkPlcMc_Pctrl_Tn,
McPara_BkPlcMc_Pctrl_Tv,
McPara_BkPlcMc_Pctrl_Nf,
McPara_BkPlcMc_Pctrl_Preset,
McPara_BkPlcMc_Pctrl_WuLimit,
McPara_BkPlcMc_Pctrl_AlignAreas,

McPara_BkPlcMc_MonPositionRange:=1800,
McPara_BkPlcMc_MonTargetRange,
McPara_BkPlcMc_MonTargetFilter,
McPara_BkPlcMc_MonPositionLagFilter,
McPara_BkPlcMc_MonDynamicLagLimit,
McPara_BkPlcMc_MonPehEnable,
McPara_BkPlcMc_MonPehTimeout,
McPara_BkPlcMc_DigInputReversed,

McPara_PFW_EnableLimitPos:=1898,
McPara_PFW_EnableLimitNeg:=1899,

McPara_BkPlcMc_JogVeloFast:=1900,
McPara_BkPlcMc_JogVeloSlow,

McPara_BkPlcMc_CustomerData:=2000,

McPara_BkPlcMc_AutoId_EnaEoT:=3000,
McPara_BkPlcMc_AutoId_EnaOv1,
McPara_BkPlcMc_AutoId_EnaZadj,
McPara_BkPlcMc_AutoId_EnaRatio,
McPara_BkPlcMc_AutoId_EnaLinTab,
McPara_BkPlcMc_AutoId_EoT_N:=3100,
McPara_BkPlcMc_AutoId_EoT_P,
McPara_BkPlcMc_AutoId_EoI_N,
McPara_BkPlcMc_AutoId_EoI_P,
McPara_BkPlcMc_AutoId_EoTlim_N,
McPara_BkPlcMc_AutoId_EoTlim_P,
McPara_BkPlcMc_AutoId_DecFactor,
McPara_BkPlcMc_AutoId_EoVlim_N,
McPara_BkPlcMc_AutoId_EoVlim_P,
McPara_BkPlcMc_AutoId_LastIdent_N,
McPara_BkPlcMc_AutoId_LastIdent_P,
McPara_BkPlcMc_AutoId_TblCount:=3150,
McPara_BkPlcMc_AutoId_TblLowEnd,
McPara_BkPlcMc_AutoId_TblHighEnd,
McPara_BkPlcMc_AutoId_TblRamp,
McPara_BkPlcMc_AutoId_TblSettling,
McPara_BkPlcMc_AutoId_TblRecovery,
McPara_BkPlcMc_AutoId_TblMinCycle,
McPara_BkPlcMc_AutoId_LinTblAvailable,
McPara_BkPlcMc_AutoId_TblValveType,
McPara_BkPlcMc_AutoId_LinTab_1:=3200,
McPara_BkPlcMc_AutoId_LinTab_2:=3400,
(* ----- *)
McRtData_BkPlcMc_ActualPosition:=10000,
McRtData_BkPlcMc_ActualAcceleration,
McRtData_BkPlcMc_PosError,
McRtData_BkPlcMc_DistanceToTarget,
McRtData_BkPlcMc_ActPressure,
McRtData_BkPlcMc_ActPressureA,
McRtData_BkPlcMc_ActPressureB,
McRtData_BkPlcMc_ActForce,
McRtData_BkPlcMc_ValvePressure,
McRtData_BkPlcMc_SupplyPressure,
McRtData_BkPlcMc_SetPosition,
McRtData_BkPlcMc_SetVelocity,
McRtData_BkPlcMc_SetAcceleration,
McRtData_BkPlcMc_SetPressure,
McRtData_BkPlcMc_SetOverride,
McRtData_BkPlcMc_LatchPosition,
McRtData_BkPlcMc_CtrlOutLag,
McRtData_BkPlcMc_CtrlOutClamping,
McRtData_BkPlcMc_CtrlOutOverlapComp,
McRtData_BkPlcMc_TargetPosition,
McRtData_BkPlcMc_NSDW:=11000,
McRtData_BkPlcMc_DCDW,
McRtData_BkPlcMc_ErrCode,
McRtData_BkPlcMc_FbState,
McRtData_BkPlcMc_CurStep,

```

```

McRtData_BkPlcMc_ParamsUnsave,
McRtData_BkPlcMc_RawPosition,
McRtData_BkPlcMc_ActPosCams,
McRtData_BkPlcMc_ReloadParams,
McRtData_BkPlcMc_EncoderMinPos,
McRtData_BkPlcMc_EncoderMaxPos,
McRtData_BkPlcMc_BufferedEntries,
McRtData_BkPlcMc_Pump_Switched:=12000,
McRtData_BkPlcMc_Pump_AreaSwitched,
McRtData_BkPlcMc_Pump_Angle:=12100,
McRtData_BkPlcMc_Pump_ModuloAngle,
McRtData_BkPlcMc_Pump_Speed,
McRtData_BkPlcMc_Pump_Torque,
McRtData_BkPlcMc_Motor_N_max,
McRtData_BkPlcMc_Active_Area_P,
McRtData_BkPlcMc_Active_Area_M,
McRtData_BkPlcMc_Active_Qmax_P,
McRtData_BkPlcMc_Active_Qmax_M,
McRtData_BkPlcMc_Active_Feed_P,
McRtData_BkPlcMc_Active_Feed_M,
McRtData_BkPlcMc_Active_N_max,
McRtData_BkPlcMc_Active_Vmax_P,
McRtData_BkPlcMc_Active_Vmax_M,
(* ----- *)
(**)
McPara_BkPlcMc_
(**)
McPara_BkPlcMc_FileMarkComplete:=32767 (* Ax.Params.bLinTabAvailable AutoIdent: .. / AutoIdent: ..
*)229
);
END_TYPE

```

**Werte**

Name	Beschreibung
McPara_CommandedPosition:=1	Die letzte kommandierte Zielposition der Achse.
McPara_SWLimitPos	Der Software-Endschalter in positiver Richtung.
McPara_SWLimitNeg	Der Software-Endschalter in negativer Richtung.
McPara_EnableLimitPos	Die Freigabe für den Software-Endschalter in positiver Richtung.
McPara_EnableLimitNeg	Die Freigabe für den Software-Endschalter in negativer Richtung.
McPara_EnablePosLagMonitoring	
McPara_MaxPositionLag	Der Schwellwert für die Schleppabstandsüberwachung.
McPara_MaxVelocitySystem	Die vom System gesetzte Obergrenze für die von der Applikation maximal kommandierbare Geschwindigkeit.
McPara_MaxVelocityAppl	Die von der Applikation maximal kommandierbare Geschwindigkeit.
McPara_ActualVelocity	Die Ist-Geschwindigkeit der Achse.
McPara_CommandedVelocity	Die letzte kommandierte Geschwindigkeit der Achse.
McPara_MaxAccelerationSystem	Die vom System gesetzte Obergrenze für die von der Applikation maximal kommandierbare Beschleunigung.
McPara_MaxAccelerationAppl	Die von der Applikation maximal kommandierbare Beschleunigung.
McPara_MaxDecelerationSystem	Die vom System gesetzte Obergrenze für die von der Applikation maximal kommandierbare Verzögerung.
McPara_MaxDecelerationAppl	Die von der Applikation maximal kommandierbare Verzögerung.
McPara_MaxJerk	Die vom System gesetzte Obergrenze für den von der Applikation maximal kommandierbaren Ruck.

Name	Beschreibung
McPara_BkPlcMc_ProfilType:=1000	: Der Typ der Sollwert-Generierung.
McPara_BkPlcMc_EnvCycletime	Die Zykluszeit der Task, in der die Kern-Bausteine (Encoder, Sollwertgenerator etc.) der Achse aufgerufen werden.
McPara_BkPlcMc_AxName	
McPara_BkPlcMc_TimeBased	Die Umschaltung der Sollwert-Generierung: Timebased oder Displacementbased.
McPara_BkPlcMc_JerkEnabled	Das Steuerwort für die Ruckbegrenzung.
McPara_BkPlcMc_LogLevel	Der Schwellwert für die Meldungsaufzeichnung.
McPara_BkPlcMc_CycleDivider	
McPara_BkPlcMc_ParamFileName	Name der Parameter-Datei.
McPara_BkPlcMc_EncoderType:=1100	Der Typ der Encoder-Auswertung.
McPara_BkPlcMc_EncoderHomingType	Achsen mit inkrementellem Wegmess-System: Die Default-Methode der Referenz-Fahrt.
McPara_BkPlcMc_EncoderZeroShift	Achsen mit absolutem Wegmess-System: Die Nullpunktverschiebung der Encoder-Auswertung.
McPara_BkPlcMc_EncoderIncWeighting	Die Inkrement-Gewichtung der Encoder-Auswertung.
McPara_BkPlcMc_EncoderIncInterpolation	Die Inkrement-Zwischenteilung der Encoder-Auswertung.
McPara_BkPlcMc_EncoderRefIndexVelo	Achsen mit inkrementellem Wegmess-System: Die Referenz-Fahrt sucht den Index (Nocken) mit dieser Geschwindigkeit.
McPara_BkPlcMc_EncoderRefIndexPositive	Achsen mit inkrementellem Wegmess-System: Die Referenz-Fahrt sucht den Index (Nocken) in positiver Richtung.
McPara_BkPlcMc_EncoderRefSyncVelo	Achsen mit inkrementellem Wegmess-System: Die Referenz-Fahrt sucht das Referenzier-Signal mit dieser Geschwindigkeit.
McPara_BkPlcMc_EncoderRefSyncPositive	Achsen mit inkrementellem Wegmess-System: Die Referenz-Fahrt sucht das Referenzier-Signal in positiver Richtung.
McPara_BkPlcMc_EncoderDefaultHomePosition,	Achsen mit inkrementellem Wegmess-System: Ein Default-Wert für die Referenz-Fahrt.
McPara_BkPlcMc_EncoderReversed	Freigabe für die invertierte Encoder-Auswertung.
McPara_BkPlcMc_EncoderBaseDistance	reserviert für abstandscodierte Encoder.
McPara_BkPlcMc_EncoderModuloBase	reserviert, nicht implementiert.
McPara_BkPlcMc_EncoderEnableLatch	Die Freigabe für die Latchfunktion einer Encoder-Hardware.
McPara_BkPlcMc_EncoderLatchedPos	
McPara_BkPlcMc_EncoderRefShift	Achsen mit inkrementellem Wegmess-System: Die Nullpunktverschiebung der Encoder-Auswertung.
McPara_BkPlcMc_EncoderRefFlag	Der Referenzier-Status der Achse.
McPara_BkPlcMc_EncoderPotiRgToRI	Bei Potentiometer-Encodern: Das Verhältnis von Potentiometer-Gesamtwiderstand zum Lastwiderstand (Eingangswiderstand der Klemme).
McPara_BkPlcMc_EncoderOverrunMask	Eine Maske zum Feststellen eines Encoder-Überlaufs.
McPara_BkPlcMc_EncoderPositionMask	Eine Maske zum Isolieren der gültigen Bits innerhalb der gemappten Variablen.
McPara_BkPlcMc_EncoderZeroSwap	Die blockweise Verschiebung des Zählbereichs der Encoder-Auswertung.
McPara_BkPlcMc_EncoderNoUpload	

Name	Beschreibung
McPara_BkPlcMc_EncoderModuloMode	
McPara_BkPlcMc_ValveOverlapCompP:=1200	Die Kompensation der Ventilüberdeckung für die positive Richtung.
McPara_BkPlcMc_ValveBendPointVelo	Die Geschwindigkeit für die Kompensation des Kennlinienknicks.
McPara_BkPlcMc_ValveBendPointOutput	Die Ventilausgabe für die Kompensation des Kennlinienknicks.
McPara_BkPlcMc_ValveResponseTime	Die Kompensation der Ventilreaktionszeit.
McPara_BkPlcMc_ValveOverlapCompM	Die Kompensation der Ventilüberdeckung für die negative Richtung.
McPara_BkPlcMc_CylinderArreaA:=1280	Die Zylinderfläche der in positiver Richtung drückenden Seite des Zylinders.
McPara_BkPlcMc_CylinderArreaB	Die Zylinderfläche der in negativer Richtung drückenden Seite des Zylinders.
McPara_BkPlcMc_DriveType:=1300	Der Typ der Antriebs-Anpassung.
McPara_BkPlcMc_AreaRatio	Das richtungsabhängige Geschwindigkeitsverhältnis.
McPara_BkPlcMc_DriveReversed	Freigabe für die invertierte Ausgabe-Anpassung.
McPara_BkPlcMc_DriveDefaultPowerOk	Drive-Power wird als verfügbar angenommen, kein Hardware-Signal erforderlich.
McPara_BkPlcMc_DriveAbsoluteOutput	Freigabe für die Absolutwertbildung bei der Ausgabe- Anpassung.
McPara_BkPlcMc_DriveIncWeighting	Die Gewichtung der Ausgabe-Anpassung.
McPara_BkPlcMc_DriveInclInterpolation	Die Zwischenteilung der Ausgabe-Anpassung.
McPara_BkPlcMc_DriveNoUpload	Ein TRUE hier verhindert die automatische Ermittlung von Achs-Parametern durch Auslesen von Daten aus einem Antrieb.
McPara_BkPlcMc_DrivelsHybrid	
McPara_BkPlcMc_HybridConcept	
McPara_BkPlcMc_Pump_Cavities	
McPara_BkPlcMc_Pump_EncType	
McPara_BkPlcMc_Pump_N_max	
McPara_BkPlcMc_Pump_N_min	
McPara_BkPlcMc_Pump_P_max	
McPara_BkPlcMc_Pump_P_min	
McPara_BkPlcMc_Pump_Q_fast_P	
McPara_BkPlcMc_Pump_Q_slow_P	
McPara_BkPlcMc_Pump_Q_fast_M	
McPara_BkPlcMc_Pump_Q_slow_M	
McPara_BkPlcMc_Pump_Q_leak	
McPara_BkPlcMc_Pump_Enc_Offset	
McPara_BkPlcMc_Cylinder_A_addP	
McPara_BkPlcMc_Cylinder_A_addM	
McPara_BkPlcMc_PrsScaling_A	
McPara_BkPlcMc_PrsScaling_B	
McPara_BkPlcMc_PrsScaling_Sys	
McPara_BkPlcMc_Motor_RampTime	
McPara_BkPlcMc_Pump_Regenerative	

Name	Beschreibung
McPara_BkPlcMc_Virtual_A_addP	Die in positiver Richtung zuschaltbare Wirkungsfläche ist beim Ölbedarf zu berücksichtigen, trägt aber zum Kraftaufbau nicht bei.
McPara_BkPlcMc_Virtual_A_addM	Die in negativer Richtung zuschaltbare Wirkungsfläche ist beim Ölbedarf zu berücksichtigen, trägt aber zum Kraftaufbau nicht bei.
McPara_BkPlcMc_Aside_PrHiResADC	Der Drucksensor an der in positiver Richtung wirkenden Fläche wird mit einer 24-Bit-Klemme eingelesen.
McPara_BkPlcMc_Bside_PrHiResADC	Der Drucksensor an der in negativer Richtung wirkenden Fläche wird mit einer 24-Bit-Klemme eingelesen.
McPara_BkPlcMc_System_PrHiResADC	Der Drucksensor am vorgespannten Speicher mit einer 24-Bit-Klemme eingelesen.
McPara_BkPlcMc_StartRamp:=1400	Nur für bestimmte Sollwertgeneratoren: Die Beschleunigungsrampe.
McPara_BkPlcMc_obsolete_1	
McPara_BkPlcMc_obsolete_2	
McPara_BkPlcMc_StopRamp:=1500	Nur für bestimmte Sollwertgeneratoren: Die Bremsrampe.
McPara_BkPlcMc_EmergencyRamp	Bei einem Nothalt: Die Zeit zum Abbremsen von Maximalgeschwindigkeit zum Stillstand.
McPara_BkPlcMc_BrakeOn	Eine Verzögerung zwischen dem Signal zum Lösen einer Bremse und dem aktiven Fahren der Achse.
McPara_BkPlcMc_BrakeOff	Eine Verzögerung zwischen dem aktiven Fahren der Achse und dem Signal zum Aktivieren einer Bremse.
McPara_BkPlcMc_BrakeSafety	Eine Verzögerung zwischen dem aktiven Fahren der Achse in eine Richtung und einem aktiven Fahren in entgegengesetzter Richtung.
McPara_BkPlcMc_CreepSpeedP:=1600	Die Schleichgeschwindigkeit in positiver Richtung. Bei symmetrischer Zielfahrt: Die Schleichgeschwindigkeit in negativer Richtung.
McPara_BkPlcMc_CreepDistanceP	Die Schleichstrecke in positiver Richtung. Bei symmetrischer Zielfahrt: Die Schleichstrecke in negativer Richtung.
McPara_BkPlcMc_BrakeDistanceP	Die Bremsstrecke in positiver Richtung. Bei symmetrischer Zielfahrt: Die Bremsstrecke in negativer Richtung.
McPara_BkPlcMc_BrakeDeadTimeP	Die Bremsstotzeit in positiver Richtung.
McPara_BkPlcMc_CreepSpeedM	Bei asymmetrischer Zielfahrt: Die Schleichgeschwindigkeit in negativer Richtung.
McPara_BkPlcMc_CreepDistanceM	Bei asymmetrischer Zielfahrt: Die Schleichstrecke in negativer Richtung.
McPara_BkPlcMc_BrakeDistanceM	Bei asymmetrischer Zielfahrt: Die Bremsstrecke in negativer Richtung.
McPara_BkPlcMc_BrakeDeadTimeM	Die Bremsstotzeit in negativer Richtung.
McPara_BkPlcMc_AsymmetricalTargeting	
McPara_BkPlcMc_LagAmp:=1700	Der Verstärkungsfaktor für den proportionalen Anteil im Lageregler.
McPara_BkPlcMc_LagAmpAdaptLimit	reserviert.
McPara_BkPlcMc_LagAmpAdaptFactor	reserviert.
McPara_BkPlcMc_ZeroCompensation	Die Offset-Kompensation der Ausgabe.



Name	Beschreibung
McPara_BkPlcMc_TargetClamping	Der Default-Ausgabewert für die Klemm-Funktion.
McPara_BkPlcMc_ReposDistance	Der Schwellwert für die automatische Repositionierung.
McPara_BkPlcMc_AutoBrakeDistance	
McPara_BkPlcMc_EnableControlLoopOnFault	
McPara_BkPlcMc_LagAmpDx	Der Schwellwert für den integrierenden Anteil im Lageregler.
McPara_BkPlcMc_LagAmpTi	Die Zeitkonstante für den integrierenden Anteil im Lageregler.
McPara_BkPlcMc_LagAmpWuLimit	Die Begrenzung (wind up limit) für den integrierenden Anteil im Lageregler.
McPara_BkPlcMc_LagAmpOutLimit	Die Ausgabe-Begrenzung für den Lageregler.
McPara_BkPlcMc_VeloAmp	Der Verstärkungsfaktor für den proportionalen Anteil im Geschwindigkeitsregler.
McPara_BkPlcMc_VeloAmpDx	Der Schwellwert für den integrierenden Anteil im Geschwindigkeitsregler.
McPara_BkPlcMc_VeloAmpTi	Die Zeitkonstante für den integrierenden Anteil im Geschwindigkeitsregler.
McPara_BkPlcMc_VeloAmpWuLimit	
McPara_BkPlcMc_VeloAmpOutLimit	Die Ausgabe-Begrenzung für den Geschwindigkeitsregler.
McPara_BkPlcMc_FeedForward	Die Vorsteuergewichtung der Achse.
McPara_BkPlcMc_LagAmpTd	Ein Parameter des erweiterten Lagereglers im MC_AxRtPosPiControllerEx_BkPlcMc [▶ 169]-Baustein: Die Verstärkung des D-Anteils.
McPara_BkPlcMc_LagAmpTdd	Ein Parameter des erweiterten Lagereglers im MC_AxRtPosPiControllerEx_BkPlcMc [▶ 169]-Baustein: Die Dämpfung des D-Anteils.
McPara_BkPlcMc_LagAmpCfb_tA	Ein Parameter des erweiterten Lagereglers im MC_AxRtPosPiControllerEx_BkPlcMc-Baustein: Die Filterzeit der Istbeschleunigungs-Rückführung.
McPara_BkPlcMc_LagAmpCfb_kA	Ein Parameter des erweiterten Lagereglers im MC_AxRtPosPiControllerEx_BkPlcMc-Baustein: Die Verstärkung der Istbeschleunigungs-Rückführung.
McPara_BkPlcMc_LagAmpCfb_tV	Ein Parameter des erweiterten Lagereglers im MC_AxRtPosPiControllerEx_BkPlcMc-Baustein: Die Filterzeit der Istgeschwindigkeits-Rückführung.
McPara_BkPlcMc_LagAmpCfb_kV	Ein Parameter des erweiterten Lagereglers im MC_AxRtPosPiControllerEx_BkPlcMc-Baustein: Die Verstärkung der Istgeschwindigkeits-Rückführung.
McPara_BkPlcMc_LagCtrlType	
McPara_BkPlcMc_LagAmpCfb_tF	
McPara_BkPlcMc_LagAmpCfb_kF	
McPara_BkPlcMc_AccFeedForward	
McPara_BkPlcMc_Pctrl_kP:=1780	Die Proportional-Verstärkung eines Kraft- oder Druckreglers.
McPara_BkPlcMc_Pctrl_Tn	Die Zeitkonstante im I-Anteil eines Kraft- oder Druckreglers.
McPara_BkPlcMc_Pctrl_Tv	Zeitkonstante im D-Anteil eines Kraft- oder Druckreglers.
McPara_BkPlcMc_Pctrl_Nf	Die Tiefe eines Mittelwertfilters im D-Anteil eines Kraft- oder Druckreglers.

Name	Beschreibung
McPara_BkPlcMc_Pctrl_Preset	Der Initialwert des I-Anteils eines Kraft- oder Druckreglers.
McPara_BkPlcMc_Pctrl_WuLimit	Die Begrenzung des I-Anteils eines Kraft- oder Druckreglers.
McPara_BkPlcMc_Pctrl_AlignAreas	Die Anpassung der Ausgabe eines Kraft- oder Druck-Reglers an die richtungsabhängigen Wirkungsflächen.
McPara_BkPlcMc_MonPositionRange:=1800	Die Toleranz für das Positionsfenster.
McPara_BkPlcMc_MonTargetRange	Die Toleranz für das Zielfenster.
McPara_BkPlcMc_MonTargetFilter	Die Filterzeit für das Zielfenster.
McPara_BkPlcMc_MonPositionLagFilter	Die Filterzeit der Schleppabstandsüberwachung.
McPara_BkPlcMc_MonDynamicLagLimit	Die Toleranz für die dynamische Schleppabstandsüberwachung.
McPara_BkPlcMc_MonPehEnable	Die Freigabe für die Überwachung der Fertigmeldung im Ziel.
McPara_BkPlcMc_MonPehTimeout	Die Filterzeit für die Überwachung der Fertigmeldung im Ziel.
McPara_BkPlcMc_DigInputReversed	
McPara_PFW_EnableLimitPos:=1898	
McPara_PFW_EnableLimitNeg:=1899	
McPara_BkPlcMc_JogVeloFast:=1900	Ein Default-Wert für eine schnelle Jog-Geschwindigkeit.
McPara_BkPlcMc_JogVeloSlow	Ein Default-Wert für eine langsame Jog-Geschwindigkeit.
McPara_BkPlcMc_CustomerData:=2000	Ein Feld mit Parametern, die eine Applikation frei verwenden kann. Diese Parameter werden mit den Achsparametern gespeichert und geladen.
McPara_BkPlcMc_Autold_EnaEoT:=3000	Automatische Identifikation: Ermittlung der Hart-Anschläge des Zylinders.
McPara_BkPlcMc_Autold_EnaOvl	Automatische Identifikation: Ermittlung der Ventil-Überdeckung.
McPara_BkPlcMc_Autold_EnaZadj	Automatische Identifikation: Ermittlung des Offsets.
McPara_BkPlcMc_Autold_EnaAratio	Automatische Identifikation: Ermittlung des richtungsbezogenen Geschwindigkeits-Verhältnisses.
McPara_BkPlcMc_Autold_EnaLinTab	Automatische Identifikation: Ermittlung der Kennlinie.
McPara_BkPlcMc_Autold_EoT_N:=3100	Automatische Identifikation: Hart-Anschlag des Zylinders in negativer Richtung.
McPara_BkPlcMc_Autold_EoT_P	Automatische Identifikation: Hart-Anschlag des Zylinders in positiver Richtung.
McPara_BkPlcMc_Autold_Eol_N	
McPara_BkPlcMc_Autold_Eol_P	
McPara_BkPlcMc_Autold_EoTlim_N	Automatische Identifikation: Ermittelte negativer Hart-Anschlag des Zylinders.
McPara_BkPlcMc_Autold_EoTlim_P	Automatische Identifikation: Ermittelte positiver Hart-Anschlag des Zylinders.
McPara_BkPlcMc_Autold_DecFactor	Automatische Identifikation: Faktor für die Gewichtung der Verzögerung.
McPara_BkPlcMc_Autold_EoVlim_N	Automatische Identifikation: Geschwindigkeits-Begrenzung der Kennlinien-Ermittlung in negativer Richtung.

Name	Beschreibung
McPara_BkPlcMc_Autold_EoVlim_P	Automatische Identifikation: Geschwindigkeits-Begrenzung der Kennlinien-Ermittlung in positiver Richtung.
McPara_BkPlcMc_Autold_LastIdent_N	Automatische Identifikation: Der Ausgabewert der letzten erfolgreichen Messung in negativer Richtung.
McPara_BkPlcMc_Autold_LastIdent_P	Automatische Identifikation: Der Ausgabewert der letzten erfolgreichen Messung in positiver Richtung.
McPara_BkPlcMc_Autold_TblCount:=3150	Automatische Identifikation: Die Anzahl der geforderten Tabellenpunkte. Da der Nullpunkt mitgezählt wird aber nur einmal vorhanden ist, muss dieser Parameter immer ungerade sein. Werte von 3 bis 99 werden akzeptiert. Allerdings ist ein Wert kleiner als 11 nicht empfehlenswert.
McPara_BkPlcMc_Autold_TblLowEnd	Automatische Identifikation: Unteres Ende der vorgesehenen Mess-Strecke.
McPara_BkPlcMc_Autold_TblHighEnd	Automatische Identifikation: Oberes Ende der vorgesehenen Mess-Strecke.
McPara_BkPlcMc_Autold_TblRamp	Automatische Identifikation: Rampe für das Aufbauen der Mess-Ausgabe. Die angegebene Zeit bezieht sich auf einen Wechsel von Null auf Vollaussteuerung. Kleinere Ausgabe-Änderungen werden in einem proportionalen Zeitanteil durchgeführt.
McPara_BkPlcMc_Autold_TblSettling	Automatische Identifikation: Wartezeit zwischen Aufbau der Mess-Ausgabe und Beginn der Messung.
McPara_BkPlcMc_Autold_TblRecovery	Automatische Kennlinien-Identifikation: Wartezeit bei einem Richtungswechsel.
McPara_BkPlcMc_Autold_TblMinCycle	
McPara_BkPlcMc_Autold_LinTblAvailable	Dieses Signal wird am Ende einer erfolgreichen Kennlinien-Vermessung auf TRUE gesetzt.
McPara_BkPlcMc_Autold_TblValveType	Automatische Identifikation: Der erwartete Typ der Kennlinie.
McPara_BkPlcMc_Autold_LinTab_1:=3200	Automatische Identifikation: Punkte der Kennlinie, bezogene Geschwindigkeit.
McPara_BkPlcMc_Autold_LinTab_2:=3400	Automatische Identifikation: Punkte der Kennlinie, bezogene Ausgabe.
McRtData_BkPlcMc_ActualPosition:=10000	Die Ist-Position.
McRtData_BkPlcMc_ActualAcceleration	Die Ist-Beschleunigung.
McRtData_BkPlcMc_PosError	Der Schleppabstand.
McRtData_BkPlcMc_DistanceToTarget	Die Reststrecke zum Ziel.
McRtData_BkPlcMc_ActPressure	Die Ist-Druckdifferenz am Ventil.
McRtData_BkPlcMc_ActPressureA	Der Ist-Druck in der A-Kammer des Zylinders.
McRtData_BkPlcMc_ActPressureB	Der Ist-Druck in der B-Kammer des Zylinders.
McRtData_BkPlcMc_ActForce	Die Ist-Kraft.
McRtData_BkPlcMc_ValvePressure	Der Druckabfall am Ventil.
McRtData_BkPlcMc_SupplyPressure	Der Istwert des Versorgungsdrucks.
McRtData_BkPlcMc_SetPosition	Der aktuelle Positions-Sollwert.
McRtData_BkPlcMc_SetVelocity	Der aktuelle Geschwindigkeits-Sollwert.
McRtData_BkPlcMc_SetAcceleration	Der aktuelle Beschleunigungs-Sollwert.
McRtData_BkPlcMc_SetPressure	Der Sollwert für Druck- oder Kraftregler.
McRtData_BkPlcMc_SetOverride	Der aktuell geltende Override-Wert.

Name	Beschreibung
McRtData_BkPlcMc_LatchPosition	Die (verrechnete) Referenzposition. Dies ist die Position, an der bei einer Homing-Operation die Istposition abschließend gesetzt wurde.
McRtData_BkPlcMc_CtrlOutLag	Die aktuelle Ausgabe des Lagereglers.
McRtData_BkPlcMc_CtrlOutClamping	Der aktuelle Wert der Klemm-Ausgabe.
McRtData_BkPlcMc_CtrlOutOverlapComp	Der aktuell ausgegebene Anteil der Überdeckungskompensation.
McRtData_BkPlcMc_TargetPositio	Die letzte kommandierte Zielposition der Achse. Diese Position wird nicht durch ein Stopp-Kommando verändert.
McRtData_BkPlcMc_NSDW:=11000	Das Zustandswort der Achse mit (u.a.) den Betriebszuständen. Es besteht kein Zusammenhang mit dem Statuswort eines externen Geräts.
McRtData_BkPlcMc_DCDW	Das Steuerwort der Achse mit (u.a.) den Freigaben. Es besteht kein Zusammenhang mit dem Steuerwort eines externen Geräts.
McRtData_BkPlcMc_ErrCode	Der aktuelle Fehler-Code der Achse.
McRtData_BkPlcMc_FbState	Der aktuelle (durch PLCopen definierte) Arbeitsschritt der Achse. Siehe auch E_TcMCFbState.
McRtData_BkPlcMc_CurStep	Der aktuelle (interne) Arbeitsschritt der Achse. Siehe auch E_TcMcCurrentStep.
McRtData_BkPlcMc_ParamsUnsave	Ein TRUE hier signalisiert, dass ein Parameter signifikant geändert wurde, die Parameter-Datei aber noch nicht wieder geschrieben wurde. Dieses Signal kann nicht durch die Bibliothek gegeben werden wenn der Parameter direkt (ohne Verwendung der Schreib-Bausteine) verändert wurde.
McRtData_BkPlcMc_RawPosition	Die nicht durch eine Nullpunkt-Verschiebung manipulierte Ist-Position.
McRtData_BkPlcMc_ActPosCams	Bei Achsen mit digitalen Positions-Sensoren: Die Sensor-Signale.
McRtData_BkPlcMc_ReloadParams	Bei Parameter-Änderung durch die Laufzeit: Eine Aufforderung an den PlcMcManager, die Parameter neu auszulesen.
McRtData_BkPlcMc_EncoderMinPos	reserviert.
McRtData_BkPlcMc_EncoderMaxPos	reserviert.
McRtData_BkPlcMc_BufferedEntries	Für Achsen mit einem Kommando-Puffer: Die Anzahl der gepufferten Kommandos.
McRtData_BkPlcMc_Pump_Switched:=12000	Bei Hybrid-Achsen: Der Status der Pumpenumschaltung.
McRtData_BkPlcMc_Pump_AreaSwitched	Bei Hybrid-Achsen: Der Status der Flächenumschaltung.
McRtData_BkPlcMc_Pump_Angle:=12100	Bei Hybrid-Achsen: Der Pumpen-Istwinkel.
McRtData_BkPlcMc_Pump_ModuloAngle	Bei Hybrid-Achsen: Der Pumpen-Istwinkel innerhalb der aktuellen Umdrehung.
McRtData_BkPlcMc_Pump_Speed	Bei Hybrid-Achsen: Die Pumpen-Istdrehzahl.
McRtData_BkPlcMc_Pump_Torque	Bei Hybrid-Achsen: Das Ist-Moment des Pumpenantriebs.
McRtData_BkPlcMc_Motor_N_max	Bei Hybrid-Achsen: Die Drehzahlbegrenzung für den Pumpenantrieb.

Name	Beschreibung
McRtData_BkPlcMc_Active_Area_P	Bei Hybrid-Achsen: Die aktuelle in positiver Richtung wirkende Wirkungsfläche.
McRtData_BkPlcMc_Active_Area_M	Bei Hybrid-Achsen: Die aktuelle in negativer Richtung wirkende Wirkungsfläche.
McRtData_BkPlcMc_Active_Qmax_P	Bei Hybrid-Achsen: Die aktuelle in positiver Richtung wirkende Förderleistung der Pumpe.
McRtData_BkPlcMc_Active_Qmax_M	Bei Hybrid-Achsen: Die aktuelle in negativer Richtung wirkende Förderleistung der Pumpe.
McRtData_BkPlcMc_Active_Feed_P	Bei Hybrid-Achsen: Die aktuelle in positiver Richtung wirkende Vorschubkonstante.
McRtData_BkPlcMc_Active_Feed_M	Bei Hybrid-Achsen: Die aktuelle in negativer Richtung wirkende Vorschubkonstante.
McRtData_BkPlcMc_Active_N_max	Bei Hybrid-Achsen: Die aktuelle Drehzahlbegrenzung für die Pumpe.
McRtData_BkPlcMc_Active_Vmax_P	Bei Hybrid-Achsen: Die aktuelle Maximalgeschwindigkeit in positiver Richtung.
McRtData_BkPlcMc_Active_Vmax_M	Bei Hybrid-Achsen: Die aktuelle Maximalgeschwindigkeit in negativer Richtung.
McPara_BkPlcMc_	
McPara_BkPlcMc_FileMarkComplete	

**McPara\_BkPlcMc\_AsymmetricalTargeting:** Die Freigabe für die asymmetrische Zielfahrt.

**McPara\_BkPlcMc\_AutoID\_EnaEol\_N:** Automatische Identifikation: Ermittelte negativer Hart-Anschlag des Zylinders in Inkrementen.

**McPara\_BkPlcMc\_AutoID\_EnaEol\_P:** Automatische Identifikation: Ermittelte positiver Hart-Anschlag des Zylinders in Inkrementen.

**McPara\_BkPlcMc\_AutoID\_MinCycle:** Automatische Identifikation: Minimale Mess-Strecke.

**McPara\_BkPlcMc\_Auto\_BrakeDistance:** Die Freigabe für die automatische Berechnung der Bremsstrecke.

**McPara\_BkPlcMc\_CycleDevider:** reserviert, nicht implementiert.

**McPara\_BkPlcMc\_DigInputsReversed:** Freigabe für die Invertierung der Eingangssignale einer Achse mit digitalen Positionssensoren.

**McPara\_BkPlcMc\_EnableControlLoopOnFaults:** Die Freigabe für die Lageregelung bei Achsfehlern.

**McPara\_BkPlcMc\_EncNoUpload:** Ein TRUE hier verhindert die automatische Ermittlung von Achs-Parametern durch Auslesen von Daten aus einem Encoder.

**McPara\_BkPlcMc\_EncoderLatchedPosition:** Die bei einem Homing gelatchte Position.

**McPara\_BkPlcMc\_obsolete\_XYZ:** Platzhalter für nicht mehr unterstützte Parameter. Diese Parameter-Codes dürfen nicht für neue Parameter wiederverwendet werden. Um dies sicher zu stellen werden solche Zahlenwerte mit Namen dieser Form belegt.

**McPara\_BkPlcMc\_VelopWuLimit:** Die Begrenzung (wind up limit) für den integrierenden Anteil im Geschwindigkeitsregler.

**McPara\_PFW\_Xyz:** Diese Parameter sind für eine Branchenlösung reserviert.

**McRtData\_BkPlcMc\_AxName:** Der textuelle Name der Achse.

**McRtData\_BkPlcMc\_FileMarkComplete:** In einer Parameter-Datei: Die logische Ende-Kennung.

### 4.3.9 E\_TcMcProfileType (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung der Regeln für die Stellwertgenerierung einer Achse verwendet.

#### Syntax

```

TYPE E_TcMcProfileType :
(
(*
The sequence below must not be changed!
New types have to be added at the end.
In case a type becomes obsolete it has to be replaced by a dummy
to ensure the numerical meaning of the other codes.
*)
(*
Die bestehende Reihenfolge darf nicht veraendert werden.
Neue Typen muessen am Ende eingefuegt werden.
Wenn ein Typ wegfallen sollte, muss er durch einen Dummy
ersetzt werden, um die numerische Zuordnung zu garantieren.
*)
iTcMc_ProfileConstAcc,
iTcMc_ProfileTimePosCtrl,
iTcMc_ProfileCosine,
iTcMc_ProfileCtrlBased,
iTcMc_ProfileTimeRamp,
iTcMc_ProfileJerkBased,
iTcMc_ProfileBufferedJerk,
iTcMc_ProfileSwitchedVelo,
iTcMc_Profile_TestOnly:=100
);
END_TYPE

```

#### Werte

Name	Beschreibung
iTcMc_ProfileConstAcc	Nur aus Kompatibilitätsgründen vorhanden, ist durch iTcMc_ProfileCtrlBased abgelöst.
iTcMc_ProfileTimePosCtrl	Nur aus Kompatibilitätsgründen vorhanden, wird nicht mehr unterstützt.
iTcMc_ProfileCosine	Nur aus Kompatibilitätsgründen vorhanden, wird nicht mehr unterstützt.
iTcMc_ProfileCtrlBased	Der Stellwert für den Antrieb wird mit abschnittsweise konstanten Beschleunigungen und Verzögerungen erzeugt. Als steuernde Größe wirken die Zeit (Beschleunigung, Geschwindigkeitswechsel, Stopp) und der Weg (Positionieren).  Dieser Generatortyp kann optional auch rein zeitgesteuert mit ständig geschlossenem Lageregler arbeiten.
iTcMc_ProfileTimeRamp	Der Stellwert für den Antrieb wird mit zeitgesteuerten Rampen für Beschleunigungen und Verzögerungen erzeugt. Als steuernde Größe wirken die Zeit (Beschleunigung, Geschwindigkeitswechsel, Stopp) und der Weg (Abbremsen, anhalten).  Dieser Generatortyp ist für Achsen vorgesehen, die an Stelle eines Encoders lediglich über digitale Nocken verfügen.
iTcMc_ProfileJerkBased	Der Stellwert für den Antrieb wird mit abschnittsweise konstanten Beschleunigungen und Verzögerungen erzeugt. Dabei wird die Verzögerung zum Ziel hin mit begrenztem Ruck abgebaut. Optional kann auch der Beschleunigungsaufbau mit begrenztem Ruck erfolgen. Als steuernde Größe wirken die Zeit (Beschleunigung, Geschwindigkeitswechsel, Stopp) und der Weg (Positionieren).  Einige Funktionen werden von diesem Generatortyp nicht oder unvollständig unterstützt.  Dieser Generatortyp kann optional auch rein zeitgesteuert mit ständig geschlossenem Lageregler arbeiten.
iTcMc_ProfileBufferedJerk	reserviert
iTcMc_ProfileSwitchedVelo	reserviert für Branchenpaket.

Name	Beschreibung
iTcMc_Profile_TestOnly	Dieser Typ ist nur für den internen Test von noch nicht freigegebenen Baustein-Prototypen vorgesehen. Er kann nicht über den PlcMcManager eingestellt werden.

### 4.3.10 E\_TcMcPressureReadingMode (ab V3.0)

Die Konstanten in dieser Auflistung werden an Bausteine für die Erfassung von Kraft- oder Druckistwerten [► 20] übergeben. Sie legen fest, welcher Istwert in der ST\_TcHydAxRtData [► 131] Struktur der Achse das Ergebnis der Auswertung zu aktualisieren ist.

#### Syntax

```

TYPE E_TcMcPressureReadingMode :
(
    iTcHydPressureReadingDefault,
    iTcHydPressureReadingActPressure,
    iTcHydPressureReadingActPressureA,
    iTcHydPressureReadingActPressureB,
    iTcHydPressureReadingActForce,
    iTcHydPressureReadingSupplyPressure,
    iTcHydPressureReadingValvePressure
);
END_TYPE
    
```

#### Werte

Name	Beschreibung
iTcHydPressureReadingDefault	Die Zielvariable ist vom verwendeten Baustein abhängig.
iTcHydPressureReadingActPressure	Zielvariable ist fActPressure. Einige Bausteine aktualisieren automatisch fActPressureA und fActPressureB.
iTcHydPressureReadingActPressureA	Zielvariable ist fActPressureA.
iTcHydPressureReadingActPressureB	Zielvariable ist fActPressureA.
iTcHydPressureReadingActForce	Zielvariable ist fActForce. Einige Bausteine aktualisieren automatisch fActPressure, fActPressureA und fActPressureB.
iTcHydPressureReadingSupplyPressure	Zielvariable ist fSupplyPressure.
iTcHydPressureReadingValvePressure	Zielvariable ist fValvePressure.

### 4.3.11 E\_TcMcValveType

Die Konstanten in dieser Auflistung werden zur Kennzeichnung von Regeln für die automatische Identifikation von Kennlinien einer Achse verwendet.

#### Syntax

```

TYPE E_TcMcValveType :
(
    (*
    The sequence below must not be changed!
    New types have to be added at the end.
    In case a type becomes obsolete it has to be replaced by a dummy
    to ensure the numerical meaning of the other codes.
    *)
    (*
    Die bestehende Reihenfolge darf nicht veraendert werden.
    Neue Typen muessen am Ende eingefuegt werden.
    Wenn ein Typ wegfallen sollte, muss er durch einen Dummy
    ersetzt werden, um die numerische Zuordnung zu garantieren.
    *)
    iTcMc_ValveTypeDefault,
    iTcMc_ValveTypeAbrupt,
    iTcMc_ValveTypeDecomp,
    iTcMc_ValveTypeLinearP,
    iTcMc_ValveTypeLinearM,
    
```

```
iTcMc_ValveTypeCopyToP,
iTcMc_ValveTypeCopyToM
);
END_TYPE
```

## Werte

Name	Beschreibung
iTcMc_ValveTypeDefault	Standard-Methode: Vermessung in beide Bewegungsrichtungen.
iTcMc_ValveTypeAbrupt	Diese Einstellung ist an Ventile mit abruptem Übergang aus dem Überdeckungsbereich vorgesehen. Dies ist nur bei sehr wenigen Ventil-Varianten der Fall und äußert sich ohne diese Einstellung durch ein sehr hartes Verhalten vor allem zu Beginn der automatischen Identifikation. <b>Hinweis</b> Diese Einstellung sollte nur in Abstimmung mit dem Hydraulik-Support vorgenommen werden.
iTcMc_ValveTypeDecomp	Diese Einstellung ist an Ventile mit Druckentlastung im Überdeckungsbereich (h-Symbol) angepasst.
iTcMc_ValveTypeLinearP	Bei dieser Einstellung wird die Identifikation nur in der negativen Richtung durchgeführt. Für die positive Richtung wird eine lineare Kennlinie angenommen, deren Endpunkt unter Verwendung des eingestellten Geschwindigkeitsverhältnis aus der maximalen Geschwindigkeit in negativer Richtung errechnet wird. <b>Hinweis</b> Das Geschwindigkeitsverhältnis wird nicht automatisch ermittelt.
iTcMc_ValveTypeLinearM	Bei dieser Einstellung wird die Identifikation nur in der positiven Richtung durchgeführt. Für die negative Richtung wird eine lineare Kennlinie angenommen, deren Endpunkt unter Verwendung des eingestellten Geschwindigkeitsverhältnis aus der maximalen Geschwindigkeit in positiver Richtung errechnet wird. <b>Hinweis</b> Das Geschwindigkeitsverhältnis wird nicht automatisch ermittelt.
iTcMc_ValveTypeCopyToP	Bei dieser Einstellung wird die Identifikation nur in der negativen Richtung durchgeführt. Für die positive Richtung werden die Messpunkte unter Verwendung des eingestellten Geschwindigkeitsverhältnis aus den Messpunkten der negativen Richtung errechnet. <b>Hinweis</b> Das Geschwindigkeitsverhältnis wird nicht automatisch ermittelt.
iTcMc_ValveTypeCopyToM	Bei dieser Einstellung wird die Identifikation nur in der positiven Richtung durchgeführt. Für die negative Richtung werden die Messpunkte unter Verwendung des eingestellten Geschwindigkeitsverhältnis aus den Messpunkten der positiven Richtung errechnet. <b>Hinweis</b> Das Geschwindigkeitsverhältnis wird nicht automatisch ermittelt.

### 4.3.12 MC\_BufferMode\_BkPlcMc (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Steuerung des Blendings nach PLC Open verwendet.

#### Syntax

```
TYPE MC_BufferMode_BkPlcMc :
(
  Aborting_BkPlcMc := 0,
  Buffered_BkPlcMc,
  BlendingLow_BkPlcMc,
  BlendingPrevious_BkPlcMc,
  BlendingNext_BkPlcMc,
  BlendingHigh_BkPlcMc
);
END_TYPE
```



**Werte**

Name	Beschreibung
Aborting_BkPlcMc	Der Default Fall: Das neue Kommando wird sofort wirksam und bricht ein bereits wirksam gewordenes Kommando ab. Der das abgebrochene Kommando überwachende Baustein wird mit CommandAborted reagieren.
Buffered_BkPlcMc	Für Achsen mit Kommando-Puffer: Dieses Kommando wird automatisch gestartet, wenn alle vorherigen Kommandos vollständig abgearbeitet wurden.
BlendingLow_BkPlcMc	Für Achsen mit Kommando-Puffer: Dieses Kommando wird ohne Zwischenhalt an das vorherige Kommando angeschlossen. Dabei wird der Übergangspunkt, wenn möglich mit der niedrigeren Geschwindigkeit der beteiligten Kommandos passiert.
BlendingPrevious_BkPlcMc	Für Achsen mit Kommando-Puffer: Dieses Kommando wird ohne Zwischenhalt an das vorherige Kommando angeschlossen. Dabei wird der Übergangspunkt, wenn möglich mit der kommandierten Geschwindigkeit des vorherigen Kommandos passiert.
BlendingNext_BkPlcMc	Für Achsen mit Kommando-Puffer: Dieses Kommando wird ohne Zwischenhalt an das vorherige Kommando angeschlossen. Dabei wird der Übergangspunkt, wenn möglich mit der kommandierten Geschwindigkeit des neuen Kommandos passiert.
BlendingHigh_BkPlcMc	Für Achsen mit Kommando-Puffer: Dieses Kommando wird ohne Zwischenhalt an das vorherige Kommando angeschlossen. Dabei wird der Übergangspunkt, wenn möglich mit der höheren Geschwindigkeit der beteiligten Kommandos passiert.

**4.3.13 MC\_CAM\_ID\_BkPlcMc (ab V3.0)**

(internal use only).

**Syntax**

```

TYPE MC_CAM_ID_BkPlcMc:
STRUCT
  stCamRef:      MC_CAM_REF_BkPlcMc;
  bValidated:    BOOL:=FALSE;
  bPeriodic:     BOOL:=FALSE;
  bMasterAbs:    BOOL:=FALSE;
  bSlaveAbs:     BOOL:=FALSE;
  blsChanged:    BOOL:=TRUE;
END_STRUCT
END_TYPE
    
```

**Werte**

Name	Beschreibung
stCamRef	Eine Kopie der <a href="#">MC_CAM_REF_BkPlcMc [► 113]</a> Struktur.
bValidated	Hier wird diese Struktur als gültig gekennzeichnet, wenn sie durch einen Baustein vom Typ <a href="#">MC_CamTableSelect_BkPlcMc [► 52]</a> initialisiert wurde.
bPeriodic	Reserviert
bMasterAbs	Hier wird festgelegt, ob die Daten der Master-Spalte absolut sind oder sich auch die Masterposition zum Zeitpunkt der Kopplung beziehen.
bSlaveAbs	Hier wird festgelegt, ob die Daten der Slave-Spalte absolut sind oder sich auch die Slaveposition zum Zeitpunkt der Kopplung beziehen.
blsChanged	Reserviert

**4.3.14 MC\_CAM\_REF\_BkPlcMc (ab V3.0)**

(internal use only).

**Syntax**

```
TYPE MC_CAM_REF_BkPlcMc:
STRUCT
  pTable:          POINTER TO LREAL:=0;
  nFirstIdx:       UDINT:=1;
  nLastIdx:        UDINT:=1;
  bEquiDistant:    BOOL:=FALSE;
END_STRUCT
END_TYPE
```

**Werte**

Name	Beschreibung
pTable	Die Adresse der Kurventabelle.
nFirstIdx	Der Index der ersten Tabellenzeile.
nLastIdx	Der Index der letzten Tabellenzeile.
bEquiDistant	Reserviert

**4.3.15 MC\_CAMSWITCH\_REF\_BkPlcMc (ab V3.0)**

Eine Variable dieses Typs wird an einen MC DigitalCamSwitch BkPlcMc [► 53] Baustein übergeben.

**Syntax**

```
TYPE CAMSWITCH_REF_BkPlcMc:
STRUCT
  Switch:          ARRAY [ciBkPlcMc_CamSwitchRef_MinIdx..ciBkPlcMc_CamSwitchRef_MaxIdx] OF MC_CAMSWITCH
                  _REFTYPE_BkPlcMc;
END_STRUCT
END_TYPE

TYPE MC_CAMSWITCH_REFTYPE_BkPlcMc:
STRUCT
  TrackNumber:     INT;
  FirstOnPosition: LREAL;
  LastOnPosition:  LREAL;
  AxisDirection:   INT;
  CamSwitchMode:   INT;
  Duration:        LREAL;
  (* private members, do not touch *)
  nCurrentState:   SINT:=0;
  bTriggered:      BOOL:=FALSE;
  fTimer:          LREAL;
  (**)
END_STRUCT
END_TYPE
```

**Parameter**

Name	Typ	Beschreibung
TrackNumber	INT	Dies ist ein Index in einem ARRAY [ciBkPlcMc_TrackRef_MinIdx..ciBkPlcMc_TrackRef_MaxIdx] OF <u>MC TRACK_REF_BkPlcMc</u> [► 116], das an einen Baustein vom Typ <u>MC DigitalCamSwitch BkPlcMc</u> [► 53] übergeben wird.
FirstOnPosition	LREAL	[mm] Der Beginn der Nockenspur. Bei zeitgesteuerten Nocken ist dies die Triggerposition.
LastOnPosition	LREAL	[mm] Das Ende der Nockenspur. Nicht wirksam bei zeitgesteuerten Nocken.
AxisDirection	INT	Hier wird festgelegt, bei welcher Bewegungsrichtung der Nocken aktiv wird: 0 = beide Richtungen, 1 = positive Richtung, 2 = negative Richtung.
CamSwitchMode	INT	Der Arbeitsmodus des Nockens: Für weggesteuerte Nocken ist hier der Wert 0, für zeitgesteuerte Nocken der Wert 1 einzutragen.
Duration	LREAL	[s] Bei zeitgesteuerten Nocken ist hier die Einschaltdauer in Sekunden einzutragen.

Name	Typ	Beschreibung
nCurrentState	SINT	Diese Elemente sind Laufzeitvariablen und dürfen nicht von der Applikation beeinflusst oder benutzt werden.
bTriggered	BOOL	
fTimer	LREAL	

### 4.3.16 MC\_Direction\_BkPlcMc (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung der Bewegungsrichtung von Achsen benutzt.

#### Syntax

```

TYPE MC_Direction_BkPlcMc:
(
MC_Positive_Direction_BkPlcMc := 1,
MC_Shortest_Way_BkPlcMc,
MC_Negative_Direction_BkPlcMc,
MC_Current_Direction_BkPlcMc,
MC_SwitchPositive_Direction_BkPlcMc,
MC_SwitchNegative_Direction_BkPlcMc
);
END_TYPE
    
```

#### Werte

Name	Beschreibung
MC_Positive_Direction_BkPlcMc	Die Bewegung erfolgt in Richtung steigender Positionswerte.
MC_Shortest_Way_BkPlcMc	Die Bewegungsrichtung wird so gewählt, dass sich eine möglichst kurze Weglänge ergibt.
MC_Negative_Direction_BkPlcMc	Die Bewegung erfolgt in Richtung sinkender Positionswerte.
MC_Current_Direction_BkPlcMc	Die Bewegung erfolgt in gleicher Richtung wie die zuletzt ausgeführte Bewegung.
MC_SwitchPositive_Direction_BkPlcMc	nicht unterstützt
MC_SwitchNegative_Direction_BkPlcMc	nicht unterstützt

### 4.3.17 MC\_HomingMode\_BkPlcMc (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung der Modi beim Referenzieren von Achsen benutzt.

#### Syntax

```

TYPE MC_HomingMode_BkPlcMc:
(
MC_DefaultHomingMode_BkPlcMc,
MC_AbsSwitch_BkPlcMc,
MC_LimitSwitch_BkPlcMc,
MC_RefPulse_BkPlcMc,
MC_Direct_BkPlcMc,
MC_Absolute_BkPlcMc,
MC_Block_BkPlcMc,
MC_FlyingSwitch_BkPlcMc,
MC_FlyingRefPulse_BkPlcMc
);
END_TYPE
    
```

#### Werte

Name	Beschreibung
MC_DefaultHomingMode_BkPlcMc	Es wird die in den Achsparametern festgelegte Referenziermethode verwendet.
MC_AbsSwitch_BkPlcMc	Es wird die Methode iTcMc_HomingOnIndex verwendet.

Name	Beschreibung
MC_LimitSwitch_BkPlcMc	nicht unterstützt
MC_RefPulse_BkPlcMc	Es wird die Methode iTcMc_HomingOnSync verwendet.
MC_Direct_BkPlcMc	Es wird die Methode iTcMc_HomingOnExec verwendet.
MC_Absolute_BkPlcMc	nicht unterstützt
MC_Block_BkPlcMc	Es wird die Methode iTcMc_HomingOnBlock verwendet.
MC_FlyingSwitch_BkPlcMc	nicht unterstützt
MC_FlyingRefPulse_BkPlcMc	nicht unterstützt

### 4.3.18 MC\_StartMode\_BkPlcMc (ab V3.0)

Die Konstanten in dieser Auflistung werden zur Kennzeichnung der Modi beim Starten von Achsen benutzt.

#### Syntax

```
TYPE MC_StartMode_BkPlcMc:
(
    MC_StartMode_Absolute:=1,
    MC_StartMode_Relative,
    MC_StartMode_RampIn
);
END_TYPE
```

#### Werte

Name	Beschreibung
MC_StartMode_Absolute	Die von MC_CamIn_BkPlcMc Bausteinen ermittelte Slave-Sollposition wird als Absolutwert betrachtet.
MC_StartMode_Relative	Die von MC_CamIn_BkPlcMc Bausteinen ermittelte Slave-Sollposition wird als Entfernung vom Ort der Ankopplung betrachtet.
MC_StartMode_RampIn	Nicht unterstützt

### 4.3.19 MC\_TRACK\_REF\_BkPlcMc (ab V3.0)

#### Syntax

```
TYPE TRACK_REF_BkPlcMc:
STRUCT
    Track:          ARRAY [ciBkPlcMc_TrackRef_MinIdx..ciBkPlcMc_TrackRef_MaxIdx] OF MC_TRACK_REFTYPE
    _BkPlcMc;
END_STRUCT
END_TYPE

TYPE MC_TRACK_REFTYPE_BkPlcMc:
STRUCT
    OnCompensation: LREAL;
    OffCompensation:LREAL;
    Hysteresis:     LREAL;
END_STRUCT
END_TYPE
```

#### Parameter

Name	Beschreibung
OnCompensation	Die zu kompensierende Einschaltzeit in Sekunden.
OffCompensation	Die zu kompensierende Ausschaltzeit in Sekunden.
Hysteresis	Um diese Entfernung muss sich die Achse vom Schaltpunkt entfernt haben, bevor ein erneutes Erreichen des Schaltpunkts ausgewertet wird.  Wenn als Totzeitkompensation ein positiver Wert angegeben wird, erfolgt die Signalgabe verzögert. Ein negativer Wert verursacht eine voreilende Signalgabe.

Name	Beschreibung
	<b>Die Zeit kann nicht genau eingehalten werden, wenn sich die steuernde Größe mit schwankender Rate ändert. Ist diese steuernde Größe eine Achsen-Istposition, muss dazu die Achsen-Istgeschwindigkeit konstant sein.</b>

### 4.3.20 OUTPUT\_REF\_BkPlcMc (ab V3.0)

Eine Struktur dieses Typs wird an Bausteine der Typen [MC\\_ReadDigitalOutput\\_BkPlcMc\(\)](#) [► 34], [MC\\_WriteDigitalOutput\\_BkPlcMc\(\)](#) [► 46] und [MC\\_DigitalCamSwitch\\_BkPlcMc\(\)](#) [► 53] übergeben.

#### Syntax

```
TYPE OUTPUT_REF_BkPlcMc:
STRUCT
    OutputBits: UDINT:=0;
END_STRUCT
END_TYPE
```

#### Parameter

Name	Typ	Beschreibung
OutputBits	UDINT	Die über diese Struktur adressierten Ausgänge.

### 4.3.21 ST\_FunctionGeneratorFD\_BkPlcMc (ab V3.0.31)

In dieser Struktur werden Parameter für die Definition der Ausgangssignale eines Funktionsgenerators zusammengefasst. Eine Struktur dieses Typs wird an [MC\\_FunctionGeneratorFD\\_BkPlcMc](#) [► 211]() Bausteine übergeben.

#### Syntax

```
TYPE ST_FunctionGeneratorFD_BkPlcMc :
STRUCT
    Sin_Amplitude:    LREAL:=0.0;
    Sin_Phase:        LREAL:=0.0;
    Sin_Offset:       LREAL:=0.0;

    Cos_Amplitude:    LREAL:=0.0;
    Cos_Phase:        LREAL:=0.0;
    Cos_Offset:       LREAL:=0.0;

    Rect_Amplitude:   LREAL:=0.0;
    Rect_Phase:       LREAL:=0.0;
    Rect_Ratio:       LREAL:=0.5;
    Rect_Offset:      LREAL:=0.0;

    Saw_Amplitude:    LREAL:=0.0;
    Saw_Phase:        LREAL:=0.0;
    Saw_Ratio:        LREAL:=0.5;
    Saw_Offset:       LREAL:=0.0;
END_STRUCT
END_TYPE
```

#### Parameter

Name	Typ	Beschreibung
Sin_Amplitude	LREAL	Der Scheitelwert der Signale.
Cos_Amplitude		
Rect_Amplitude		
Saw_Amplitude		
Sin_Phase	LREAL	Die Phasenverschiebung der Signale.
Cos_Phase		
Rect_Phase		

Name	Typ	Beschreibung
Saw_Phase		
Sin_Offset Cos_Offset Rect_Offset Saw_Offset	LREAL	Der Nullpunktversatz der Signale.
Rect_Ratio Saw_Ratio	LREAL	Das Tastverhältnis des Rechteck- bzw. des Sägezahnsignals.

### 4.3.22 ST\_FunctionGeneratorTB\_BkPlcMc (ab V3.0.31)

In dieser Struktur werden Parameter für die Zeitbasis eines oder mehrerer Funktionsgeneratoren zusammengefasst. Eine Struktur dieses Typs wird an [MC\\_FunctionGeneratorTB\\_BkPlcMc \[▶ 212\]\(\)](#), [MC\\_FunctionGeneratorFD\\_BkPlcMc \[▶ 117\]\(\)](#) und [MC\\_FunctionGeneratorSetFrg\\_BkPlcMc \[▶ 212\]\(\)](#) Bausteine übergeben.

#### Syntax

```

TYPE ST_FunctionGeneratorTB_BkPlcMc :
STRUCT
    Frequency:          LREAL:=0.000001;
    Freeze:             BOOL:=FALSE;

    CycleCount:        DINT:=0;
    CurrentTime:       LREAL:=0.0;
    CurrentRatio:      LREAL:=0.0;
END_STRUCT
END_TYPE

```

#### Parameter

Name	Typ	Beschreibung
Frequency	LREAL	Die Arbeitsfrequenz der von einem <a href="#">MC_FunctionGeneratorTB_BkPlcMc [▶ 212]()</a> Baustein erzeugten Zeitbasis in Herz.
Freeze	BOOL	Wenn diese Variable auf TRUE steht, wertet ein <a href="#">MC_FunctionGeneratorTB_BkPlcMc [▶ 212]()</a> Baustein die Struktur nicht aus.
CycleCount	DINT	Die Anzahl der vollständig erzeugten Signalabläufe.
CurrentTime	LREAL	Die seit Beginn des aktuell erzeugten Signalablaufs vergangene Zeit.
CurrentRatio	LREAL	Der normierte Fortschritt seit Beginn des aktuell erzeugten Signalablaufs.

### 4.3.23 ST\_TcMcAutoIdent (ab V3.0.4)

In dieser Struktur werden die Parameter für einen [MC\\_AxUtiAutoIdent\\_BkPlcMc \[▶ 254\]](#) Baustein abgelegt. Näheres über die Bedeutung der einzelnen Elemente ist dort beschrieben.

#### Syntax

```

TYPE ST_TcMcAutoIdent :
(* last modification: 08.11.2019 *)
STRUCT
    EndOfTravel_Negativ:          LREAL:=0.0;
    EndOfTravel_Positiv:         LREAL:=0.0;
    EndOfTravel_NegativLimit:    LREAL:=0.0;
    EndOfTravel_PositivLimit:    LREAL:=0.0;
    DecelerationFactor:          LREAL:=1.0;
    EndOfVelocity_NegativLimit:  LREAL:=0.0;
    EndOfVelocity_PositivLimit:  LREAL:=0.0;
    EndOfTravel_LastIdent_P:     LREAL:=0.0;
    EndOfTravel_LastIdent_M:     LREAL:=0.0;
    ValveCharacteristicLowEnd:    LREAL:=0.0;
    ValveCharacteristicHighEnd:  LREAL:=0.0;
    ValveCharacteristicRamp:      LREAL:=0.0;
    ValveCharacteristicSettling: LREAL:=0.0; (* starting with V3.0.32 *)

```

```

ValveCharacteristicRecovery:LREAL:=0.0;
ValveCharacteristicMinCycle:LREAL:=0.0;

Valve_LinLimitP: LREAL:=0.0;    (* starting with V3.0.46 *)
Valve_LinLimitM: LREAL:=0.0;

ValveCharacteristicTable:  ARRAY[1..100,1..2] OF LREAL;

EndOfIncrements_Negativ:   DINT:=0;
EndOfIncrements_Positiv:   DINT:=0;

ValveCharacteristicType:   INT:=0; (* starting with V3.0.33 *)
ValveCharacteristicTblCount:INT:=0;

EnableEndOfTravel:        BOOL:=FALSE;
EnableOverlap:            BOOL:=FALSE;
EnableZeroAdjust:        BOOL:=FALSE;
EnableArreaRatio:        BOOL:=FALSE;
EndOfTravel_PositivDone:  BOOL:=FALSE;
EndOfTravel_NegativDone:  BOOL:=FALSE;
EnableValveCharacteristic: BOOL:=FALSE;
EnableNoUturn:           BOOL:=FALSE;
END_STRUCT
END_TYPE

```

**Parameter**

Name	Typ	Beschreibung
EndOfTravel_Negativ	LREAL	
EndOfTravel_Positiv	LREAL	
EndOfTravel_NegativLimit	LREAL	
EndOfTravel_PositivLimit	LREAL	
DecelerationFactor	LREAL	
EndOfVelocity_NegativLimit	LREAL	
EndOfVelocity_PositivLimit	LREAL	
EndOfTravel_LastIdent_P	LREAL	
EndOfTravel_LastIdent_M	LREAL	
ValveCharacteristicLowEnd	LREAL	
ValveCharacteristicHighEnd	LREAL	
ValveCharacteristicRamp	LREAL	
ValveCharacteristicSettling	LREAL	
ValveCharacteristicRecovery	LREAL	
ValveCharacteristicMinCycle	LREAL	
Valve_LinLimitP	LREAL	
Valve_LinLimitM	LREAL	
ValveCharacteristicTable	ARRAY	
EndOfIncrements_Negativ	DINT	
EndOfIncrements_Positiv	DINT	
ValveCharacteristicType	INT	
ValveCharacteristicTblCount	INT	
EnableEndOfTravel	BOOL	
EnableOverlap	BOOL	
EnableZeroAdjust	BOOL	
EnableArreaRatio	BOOL	
EndOfTravel_PositivDone	BOOL	
EndOfTravel_NegativDone	BOOL	
EnableValveCharacteristic	BOOL	
EnableNoUturn	BOOL	

### 4.3.24 ST\_TcHydAxParam (ab V3.0)

Diese Struktur enthält sämtliche Parameter der Achse. Unter Setup (teilweise in Vorbereitung) werden geeignete Vorgehensweisen zur Achsinbetriebnahme vorgestellt.



Die Reihenfolge der Parameter ist nicht garantiert.

#### Syntax

```

TYPE ST_TcHydAxParam :
(* last modification: 08.07.2020 *)
STRUCT
  (* =====
  this section isn't saved / dieser Bereich wird nicht gesichert
  ===== *)
  sParamFileName: STRING(255) := 'DefAxParmFile.dat';
  (* =====
  from this point all parameters are saved /
  von hier an werden alle Parameter gesichert
  ===== *)
  fAcc: LREAL := 2000.0;
  fAreaRatio: LREAL := 1.0;
  fBrakeDeadTimeM: LREAL := 0.0;
  fBrakeDeadTimeP: LREAL := 0.0;
  fBrakeDistanceM: LREAL := 0.1;
  fBrakeDistanceP: LREAL := 0.1;
  fBrakeOffDelay: LREAL := 0.0;
  fBrakeOnDelay: LREAL := 0.0;
  fBrakeSafetyDelay: LREAL := 0.0;
  fCreepDistanceM: LREAL := 1.0;
  fCreepDistanceP: LREAL := 1.0;
  fCreepSpeedM: LREAL := 80.0;
  fCreepSpeedP: LREAL := 80.0;
  fCustomerData: ARRAY [1..iTcHydfCustDataMaxIdx] OF LREAL;
  fCycletime: LREAL := 0.010;
  fCylinder_ArreaA: LREAL := 1.0;
  fCylinder_ArreaB: LREAL := 1.0;
  fCylinder_Mass: LREAL := 1.0;
  fCylinder_Stroke: LREAL := 1.0;
  fDec: LREAL := 2000.0;
  fDrive_IncInterpolation: LREAL := 1.0;
  fDrive_IncWeighting: LREAL := 0.001;
  fEmergencyRamp: LREAL := 0.1;
  fEnc_BaseDistance: LREAL := 0.001;
  fEnc_DefaultHomePosition: LREAL := 0.0;
  fEnc_IncInterpolation: LREAL := 1.0;
  fEnc_IncWeighting: LREAL := 0.001;
  fEnc_ModuloBase: LREAL := 0.001;
  fEnc_PotiRgToRl: LREAL := 0.0;
  fEnc_RefIndexVelo: LREAL := 0.1;
  fEnc_RefSyncVelo: LREAL := 0.1;
  fEnc_ZeroShift: LREAL := 0.0;
  fJogVeloFast: LREAL := 100.0;
  fJogVeloSlow: LREAL := 25.0;
  fFeedForward: LREAL := 1.0;
  fAccFeedForward: LREAL := 0.0;
  fLagAmp: LREAL := 0.05;
  fLagAmpDp: LREAL := 0.0;
  fLagAmpDx: LREAL := 0.0;
  fLagAmpTi: LREAL := 0.0;
  fLagAmpOutL: LREAL := 0.0;
  fLagAmpWuL: LREAL := 0.0;
  fLagAmpTd: LREAL := 0.0;
  fLagAmpTdd: LREAL := 0.0;
  fLagAmpCfb_kV: LREAL := 0.0;
  fLagAmpCfb_tV: LREAL := 0.0;
  fLagAmpCfb_kA: LREAL := 0.0;
  fLagAmpCfb_tA: LREAL := 0.0;
  fLagAmpCfb_kF: LREAL := 0.0;
  fLagAmpCfb_tF: LREAL := 0.0;
  fMaxAcc: LREAL := 500.0;
  fMaxDec: LREAL := 500.0;
  fMaxDynamicLag: LREAL := 0.0;
  fMaxJerk: LREAL := 1000.0;
  fMaxLag: LREAL := 0.0;

```



```

fMaxLagFilter:          LREAL := 0.0;
fMaxVelo:              LREAL := 500.0;
fMonPositionRange:    LREAL := 1.0;
fMonTargetFilter:     LREAL := 1.0;
fMonTargetRange:     LREAL := 1.0;
fPEH_Timeout:         LREAL := 0.0;
fRefVelo:             LREAL := 500.0;
fReposDistance:      LREAL := 0.0;
fSoftEndMax:         LREAL := 10000.0;
fSoftEndMin:         LREAL := 0.0;
fStartAccDistance:   LREAL := 1.0;
fStartRamp:          LREAL := 1.0;
fStopRamp:           LREAL := 1.0;
fTargetClamping:     LREAL := 0.0;
fVeloAmp:            LREAL := 0.0;
fVeloAmpDx:          LREAL := 0.0;
fVeloAmpTi:          LREAL := 0.0;
fVeloAmpOutL:        LREAL := 0.0;
fVeloAmpWuL:         LREAL := 0.0;
fValve_BendPointOutput: LREAL := 0.0;
fValve_BendPointVelo: LREAL := 0.0;
fValve_OverlapCompM: LREAL := 0.0;
fValve_OverlapCompP: LREAL := 0.0;
fValve_ResponseTime: LREAL := 0.0;
fZeroCompensation:   LREAL := 0.0;

nEnc_OverrunMask:     DWORD := 0;
nEnc_PositionMask:   DWORD := 0;
nEnc_ZeroSwap:        DINT := 0;
nDigInReversed:      DINT := 0;

nCycleDivider:       INT := 1;
nDrive_Type:         E_TcMcDriveType:=iTcMc_Drive_Customized;
nEnc_HomingType:     E_TcMcHomingType:=iTcMc_HomingOnBlock;
nEnc_Type:           E_TcMcEncoderType:=iTcMc_EncoderSim;

nJerkEnabled:        WORD := 16#0101;
nProfileType:        E_TcMcProfileType:=iTcMc_ProfileCtrlBased;
nControllerType:     WORD := 16#0101;
nOverlapDefMode:     WORD := 0;

bAsymmetricalTargeting:  BOOL := FALSE;
bDrive_AbsoluteOutput:   BOOL := FALSE;
bDrive_DefaultPowerOk:  BOOL := FALSE;
bDrive_Reversed:        BOOL := FALSE;
bEnableAutoBrakeDistance:  BOOL := FALSE;
bEnableControlLoopOnFault:  BOOL := FALSE;
bEnc_RefIndexPositive:   BOOL := FALSE;
bEnc_RefSyncPositive:    BOOL := FALSE;

bEnc_Reversed:         BOOL := FALSE;
bMaxLagEna:           BOOL := FALSE;
bPEH_Enable:          BOOL := FALSE;
bPosCtrlAccEna:       BOOL := FALSE;
bSoftEndMaxEna:       BOOL := FALSE;
bSoftEndMinEna:       BOOL := FALSE;
bTimeBased:           BOOL := FALSE;
bLinTabAvailable:     BOOL := FALSE;

bEnc_NoUpload:        BOOL := FALSE;
bDrive_NoUpload:      BOOL := FALSE;
bDriveIsHybrid:       BOOL := FALSE;
bAlignedStart:        BOOL := FALSE;
bEncModuloMode:       BOOL := FALSE;

(*-----*)

stHybrid:             ST_TcHybridAxParam;
stPctrl:              ST_TcPctrlParam;
END_STRUCT
END_TYPE

```

## Parameter

Name	Typ	Beschreibung
sParamFileName	STRING	Unter diesem Dateinamen werden die Achsparameter als DAT-Datei gespeichert.
fAcc	LREAL	[mm/s <sup>2</sup> ] Die absolute Beschleunigungs-Begrenzung der Achse.
fAreaRatio	LREAL	[1] Hier kann die Richtungsabhängigkeit der Geschwindigkeit kompensiert werden.
fBrakeDeadTimeM	LREAL	[s] Ab V3.0.8: Dieser Parameter ermöglicht es, die für die negative Richtung eingestellte Bremsstrecke einen zur Istgeschwindigkeit proportionalen Anteil zu erweitern.
fBrakeDeadTimeP	LREAL	[s] Ab V3.0.8: Dieser Parameter ermöglicht es, die für die positive Richtung eingestellte Bremsstrecke einen zur Istgeschwindigkeit proportionalen Anteil zu erweitern.
fBrakeDistanceM	LREAL	[mm] Ab V3.0.8: Bremsstrecke: Bei bAsymmetricalTargeting = TRUE wird in dieser negativen Entfernung zum Ziel die aktive profilgesteuerte Stellwertgenerierung eingestellt und optional ein Stillstandslageregler oder ein anderer im Ziel geltender Mechanismus aktiviert.
fBrakeDistanceP	LREAL	[mm] Ab V3.0.8: Bremsstrecke: In dieser nicht richtungsabhängigen oder (bei bAsymmetricalTargeting = TRUE) positiven Entfernung zum Ziel wird die aktive profilgesteuerte Stellwertgenerierung eingestellt und optional ein Stillstandslageregler oder ein anderer im Ziel geltender Mechanismus aktiviert.
fBrakeOffDelay	LREAL	[s] Wenn dieser Parameter auf einen Wert größer 0 gesetzt wird, hält der Stellwertgenerator zwischen der steigenden Flanke an ST_TcPlcDeviceOutput.bBrakeOff und dem Beginn der Beschleunigungsphase eine Wartezeit ein.
fBrakeOnDelay	LREAL	[s] Wenn dieser Parameter auf einen Wert größer 0 gesetzt wird, hält der Stellwertgenerator zwischen dem Ende der aktiven Profilgenerierung und der fallenden Flanke an <a href="#">ST_TcPlcDeviceOutput [► 140].bBrakeOff</a> eine Wartezeit ein.

Name	Typ	Beschreibung
fBrakeSafetyDelay	LREAL	[s] Wenn dieser Parameter auf einen Wert größer 0 gesetzt wird, hält der Stellwertgenerator der fallenden Flanke an <u>ST_TcPlcDeviceOutput</u> [▶ 140].bBrakeOff am Ende einer aktiven Profilgenerierung und der steigenden Flanke des nächsten Fahrauftrags eine Wartezeit ein.
fCreepDistanceM	LREAL	[mm] Ab V3.0.8: Bei bAsymmetricalTargeting = TRUE wird ab dieser negativen Entfernung zum Ziel für die letzte Phase der profilgesteuerten Stellwertgenerierung fCreepSpeedM als Stellwert verwendet.
fCreepDistanceP	LREAL	[mm] Ab V3.0.8: Ab dieser nicht richtungsabhängigen oder (bei bAsymmetricalTargeting = TRUE) positiven Entfernung zum Ziel wird für die letzte Phase der profilgesteuerten Stellwertgenerierung fCreepSpeedP als Stellwert verwendet.
fCreepSpeedM	LREAL	[mm/s] Ab V3.0.8: Diese Geschwindigkeit wird bei bAsymmetricalTargeting = TRUE und negativer Bewegungsrichtung für die letzte Phase der profilgesteuerten Stellwertgenerierung verwendet.
fCreepSpeedP	LREAL	[mm/s] Ab V3.0.8: Diese Geschwindigkeit wird nicht richtungsabhängig oder (bei bAsymmetricalTargeting = TRUE) bei positiver Bewegungsrichtung für die letzte Phase der profilgesteuerten Stellwertgenerierung verwendet.
fCustomerData	ARRAY	Hier stehen 20 LREAL Parameter zur freien Verwendung durch die Applikation zur Verfügung. Sie werden zusammen mit den anderen Achsparametern geladen und gespeichert. Bausteine der Bibliothek machen keinen selbständigen Gebrauch von diesen Parametern, können aber durch die Art des Aufrufs von der Applikation hierzu veranlasst werden.
fCycletime	LREAL	[s] Die Zykluszeit der PLC-Task, von der die Bibliotheksbausteine aufgerufen werden. Dieser Wert wird durch einen <u>MC_AxUtiStandardInit_BkPlcMc</u> [▶ 238]() Baustein automatisch

Name	Typ	Beschreibung
		ermittelt und darf durch die Applikation verwendet, aber <b>nicht verändert</b> werden.
fCylinder_ArreaA	LREAL	[mm <sup>2</sup> ] Die aktive Fläche des Zylinders, die bei einer Bewegung in positiver Richtung mit Druck beaufschlagt ist.
fCylinder_ArreaB	LREAL	[mm <sup>2</sup> ] Die aktive Fläche des Zylinders, die bei einer Bewegung in negativer Richtung mit Druck beaufschlagt ist.
fCylinder_Mass	LREAL	reserviert.
fCylinder_Stroke	LREAL	reserviert.
fDec	LREAL	fDec: [mm/s <sup>2</sup> ] Die absolute Verzögerungs-Begrenzung der Achse.
fDrive_IncInterpolation	LREAL	Dieser Parameter wird bei einigen Ausgabegeräten für die interne Umrechnung des Geschwindigkeits-Stellwerts verwendet.
fDrive_IncWeighting	LREAL	Dieser Parameter wird bei einigen Ausgabegeräten für die interne Umrechnung des Geschwindigkeits-Stellwerts verwendet.
fEmergencyRamp	LREAL	[s] Dieser Parameter legt die Zeit fest, die benötigt wird, um von fRefVelo zum Stillstand zu verzögern. Er wird von verschiedenen Stellwertgeneratoren zur Reaktion auf nicht geplante Notstop-Anforderungen (wegfallende Reglerfreigabe, Fehlerfall, Bausteinanruf) benutzt.
fEnc_BaseDistance	LREAL	[mm] Dieser Parameter wird bei der Auswertung von Encodern mit abstandscodierten Nullmarken verwendet.
fEnc_DefaultHomePosition	LREAL	[mm] Hier kann eine Position hinterlegt werden, die als Referenzposition an einen <code>MC Home BkPlcMc [▶ 68]()</code> Baustein übergeben werden kann. Wird eine Referenzfahrt durch den PlcMcManager ausgelöst, wird der hier abgelegte Wert so verwendet. Soll dies auch bei einer von der PLC-Applikation ausgelösten Referenzfahrt der Fall sein, ist dieser Parameter beim Aufruf des verwendeten Bausteins zu übergeben.
fEnc_IncInterpolation	LREAL	[mm/n] Dieser Parameter legt die Auflösung der Istpositionsermittlung der Achse fest.

Name	Typ	Beschreibung
fEnc_IncWeighting	LREAL	[1] Dieser Parameter legt die Auflösung der Istpositionsermittlung der Achse fest.
fEnc_ModuloBase	LREAL	
fEnc_PotiRgToRI	LREAL	[1] Wird von einigen Bausteinen zur Linearisierung von einfachen durch den Eingangswiderstand der interface-Elektronik belasteten Potentiometer-Weggebern verwendet.
fEnc_RefIndexVelo	LREAL	[1] Dieser Parameter legt den Stellwert als Anteil von fRefVelo fest, der bei einer Referenzfahrt während der Suche des Referenzindex (Nocken) ausgegeben wird.
fEnc_RefSyncVelo	LREAL	[81] Dieser Parameter legt den Stellwert als Anteil von fRefVelo fest, der bei einer Referenzfahrt während der Suche des Referenzpulses (Sync-Puls, Null-Impuls) ausgegeben wird.
fEnc_ZeroShift	LREAL	[mm] Dieser Parameter verschiebt den Nullpunkt der Istpositionsermittlung der Achse.
fJogVeloFast	LREAL	[mm/s] Sollgeschwindigkeit für die schnelle Handfahrt.
fJogVeloSlow	LREAL	[mm/s] Sollgeschwindigkeit für die langsame Handfahrt.
fFeedForward	LREAL	
fAccFeedForward	LREAL	[s] Die optionale Beschleunigungsvorsteuerung der Achse.
fLagAmp	LREAL	[mm/s pro mm → 1/s] Die Kp-Verstärkung des Stillstands-Lagereglers.
fLagAmpDp	LREAL	[mm] In Vorbereitung: Das Ansprechfenster des erweiterten Stillstands-Lagereglers.
fLagAmpDx	LREAL	[mm] In Vorbereitung: Das Ansprechfenster des Stillstands-Lagereglers.
fLagAmpTi	LREAL	In Vorbereitung: Die Integrationszeit des Stillstands-Lagereglers.
fLagAmpOutL	LREAL	In Vorbereitung: Die Ausgabebegrenzung Stillstands-Lagereglers.
fLagAmpWuL	LREAL	In Vorbereitung: Die Begrenzung des I-Anteils Stillstands-Lagereglers.
fLagAmpTd	LREAL	[1] Optional: Vorhaltezeit des Differential-Anteils des Lagereglers. <b>Dieser Parameter wird nur von MC_AxRtPosPiControllerEx_BkPlcMc() verwendet.</b>

Name	Typ	Beschreibung
fLagAmpTdd	LREAL	[s] Optional: Dämpfungszeit des Differential-Anteils des Lagereglers. <b>Dieser Parameter wird nur von MC_AxRtPosPiControllerEx_BkP IcMc() verwendet.</b>
fLagAmpCfb_kV	LREAL	V: : [1] Optional: Gewichtungsfaktor der Istgeschwindigkeits-Aufschaltung in der Zustandsrückführung des Lagereglers. <b>Dieser Parameter wird nur von MC_AxRtPosPiControllerEx_BkP IcMc() verwendet.</b>
fLagAmpCfb_tV	LREAL	[1] Optional: Filterzeit der Istgeschwindigkeits-Aufschaltung in der Zustandsrückführung des Lagereglers. <b>Dieser Parameter wird nur von MC_AxRtPosPiControllerEx_BkP IcMc() verwendet.</b>
fLagAmpCfb_kA	LREAL	[1] Optional: Gewichtungsfaktor der Istbeschleunigungs-Aufschaltung in der Zustandsrückführung des Lagereglers. <b>Dieser Parameter wird nur von MC_AxRtPosPiControllerEx_BkP IcMc() verwendet.</b>
fLagAmpCfb_tA	LREAL	[1] Optional: Filterzeit der Istbeschleunigungs-Aufschaltung in der Zustandsrückführung des Lagereglers. <b>Dieser Parameter wird nur von MC_AxRtPosPiControllerEx_BkP IcMc() verwendet.</b>
fLagAmpCfb_kF	LREAL	[1] Optional: Gewichtungsfaktor der Istkraft-Aufschaltung in der Zustandsrückführung des Lagereglers. <b>Dieser Parameter wird nur von MC_AxRtPosPiControllerEx_BkP IcMc() verwendet.</b>
fLagAmpCfb_tF	LREAL	[1] Optional: Filterzeit der Istkraft-Aufschaltung in der Zustandsrückführung des Lagereglers.
fMaxAcc	LREAL	[mm/s <sup>2</sup> ] Die für die Bausteine geltende Beschleunigungs-Begrenzung der Achse. Dieser Wert ist auf fAcc begrenzt.
fMaxDec	LREAL	[mm/s <sup>2</sup> ] Die für die Bausteine geltende Verzögerungs-Begrenzung der Achse. Dieser Wert ist auf fDec begrenzt.
fMaxDynamicLag	LREAL	[s] Dieser Parameter legt einen der Grenzwerte für die Schlepp-Überwachung fest.

Name	Typ	Beschreibung
fMaxJerk	LREAL	[mm/s <sup>3</sup> ] Die für die Bausteine geltende Ruckbegrenzung der Achse. Dieser Wert wird verwendet, wenn als Profiltyp iTcMc_ProfileJerkBased eingestellt ist.
fMaxLag	LREAL	[mm] Dieser Parameter legt einen der Grenzwerte für die Schlepp-Überwachung fest.
fMaxLagFilter	LREAL	[s] Dieser Parameter legt einen der Grenzwerte für die Schlepp-Überwachung fest.
fMaxVelo	LREAL	[mm/s] Die maximale durch Bausteine verwendbare Geschwindigkeit. Wird von einem Baustein ein höherer Wert verwendet wird dieser in der Regel ohne eine Fehlermeldung entsprechend begrenzt. Dieser Parameter wird auf fRefVelo begrenzt.
fMonPositionRange	LREAL	[mm] Dieser Parameter wird für die Zielfenster-Überwachung verwendet.
fMonTargetFilter	LREAL	[s] Dieser Parameter wird für die Zielfenster-Überwachung verwendet.
fMonTargetRange	LREAL	[mm] Dieser Parameter wird für die Zielfenster-Überwachung verwendet.
fPEH_Timeout	LREAL	[s] Dieser Parameter legt den Grenzwert für die PEH-Überwachung fest.
fRefVelo	LREAL	[mm/s] Dieser Parameter legt die maximale absolute Geschwindigkeit der Achse fest.
fReposDistance	LREAL	[mm] Wenn dieser Parameter größer als 0 ist und die Achse ein einmal angefahrenes Ziel um mehr als diese Strecke verlassen hat wird automatisch eine erneute Positionierung auf das Ziel veranlasst.
fSoftEndMax	LREAL	[mm] Der obere (positive) Software-Endschalter.
fSoftEndMin	LREAL	[mm] Der untere (negative) Software-Endschalter.
fStartAccDistance	LREAL	obsolet, wird in Kürze entfernt.
fStartRamp	LREAL	[s] Dieser Parameter legt die Zeit fest die bei Profiltyp iTcMc_ProfileTimeRamp benötigt wird, um auf fRefVelo zu beschleunigen.
fStopRamp	LREAL	[s] Dieser Parameter legt die Zeit fest, die benötigt wird, um von fRefVelo zum Stillstand zu verzögern. Er wird bei Profiltyp iTcMc_ProfileTimeRamp für die

Name	Typ	Beschreibung
		Zielanfahrt verwendet, aber auch von verschiedenen Stellwertgeneratoren zur Reaktion auf nicht geplante Stopp-Anforderungen (wegfallende Vorschubfreigabe, Fehlerfall, Bausteinaufruf) benutzt.
fTargetClamping	LREAL	[v] Ist dieser Parameter auf einen Wert größer als Null gesetzt, wird beim Erreichen eines Fahrziels dieser Stellwert vorzeichenrichtig ausgegeben. Eine Lageregelung wird unterdrückt.
fVeloAmp	LREAL	Die Kp-Verstärkung des unterlagerten Geschwindigkeitsreglers.
fVeloAmpDx	LREAL	Das Ansprechfenster des unterlagerten Geschwindigkeitsreglers.
fVeloAmpTi	LREAL	Die Integrationszeit des unterlagerten Geschwindigkeitsreglers.
fVeloAmpOutL	LREAL	Die Ausgabebegrenzung des unterlagerten Geschwindigkeitsreglers.
fVeloAmpWuL	LREAL	Die Begrenzung des I-Anteils des unterlagerten Geschwindigkeitsreglers.
fValve_BendPointOutput	LREAL	[1] Bei Ventilen mit Kennlinienknick kann dieser Parameter für eine einfache Linearisierung genutzt werden.
fValve_BendPointVelo	LREAL	[1] Bei Ventilen mit Kennlinienknick kann dieser Parameter für eine einfache Linearisierung genutzt werden.
fValve_OverlapCompM	LREAL	[1] Ab V3.0.8: Kompensation einer (bei bAsymmetricalTargeting = TRUE) für die negative Richtung verwendete Ventilüberdeckung.
fValve_OverlapCompP	LREAL	[1] Ab V3.0.8: Kompensation einer nicht richtungsabhängigen oder (bei bAsymmetricalTargeting = TRUE) für die positive Richtung verwendeten Ventilüberdeckung.
fValve_ResponseTime	LREAL	[s] Mit diesem Parameter kann eine Totzeitkompensation des Stellglieds vorgenommen werden.
fZeroCompensation	LREAL	[V] Hier kann ein Analog-Offset der Geschwindigkeitsausgabe kompensiert werden.
nEnc_OverrunMask	DWORD	
nEnc_PositionMask	DWORD	
nEnc_ZeroSwap	DINT	reserviert.
nDigInReversed	DINT	
nCycleDivider	INT	reserviert.



Name	Typ	Beschreibung
nDrive_Type	E_TcMcDriveType	Hier wird der <u>Antriebstyp</u> [ <a href="#">▶ 90</a> ] festgelegt.
nEnc_HomingType	E_TcMcHomingType	Hier wird die Referenziermethode festgelegt, die ein <u>MC Home BkPlcMc</u> [ <a href="#">▶ 68</a> ]() Baustein verwendet, wenn <u>MC DefaultHomingMode BkPlcMc</u> [ <a href="#">▶ 115</a> ] als HomingMode übergeben wird.
nEnc_Type	E_TcMcEncoderType	Hier wird der <u>Encodertyp</u> [ <a href="#">▶ 93</a> ] festgelegt.
nJerkEnabled	WORD	Diese Bitmaske legt fest, an welchen Übergängen im Profil eine Ruckbegrenzung zu erfolgen hat. Dieser Wert wird verwendet, wenn als Profiltyp iTcMc_ProfileJerkBased eingestellt ist.
nProfileType	E_TcMcProfileType	Hier wird der <u>Stellwertgenerator</u> [ <a href="#">▶ 110</a> ] festgelegt.
nControllerType	WORD	Dieser Parameter wird vom aufgerufenen Lageregler automatisch gesetzt. Er wird nicht gespeichert.
nOverlapDefMode	WORD	reserviert.
bAsymmetricalTargeting	BOOL	Ab V3.0.8: Wenn dieser Parameter auf TRUE steht, werden richtungsabhängige Parameter bei der Zielannäherung und der Überdeckungskompensation wirksam.
bDrive_AbsoluteOutput	BOOL	Wenn dieser Parameter auf TRUE steht, werden Stellwerte richtungsunabhängig immer positiv ausgegeben.
bDrive_DefaultPowerOk	BOOL	Ist dieser Parameter gesetzt, wird die PowerOk-Rückmeldung in der <u>ST TcPlcDeviceInput</u> [ <a href="#">▶ 138</a> ] Struktur der Achse ignoriert.
bDrive_Reversed	BOOL	Ist dieser Parameter gesetzt, wird der Stellwert negiert ausgegeben.
bEnableAutoBrakeDistance	BOOL	Wenn dieser Parameter auf TRUE steht, werden fCreepDistanceM und fCreepDistanceP automatisch aus fCreepSpeedM bzw. fCreepSpeedP und fLagAmp errechnet.
bEnableControlLoopOnFault	BOOL	Wenn dieser Parameter auf TRUE steht, wird der Stillstands-Lageregler der Achse auch im Fehlerfall aktiv. Voraussetzung: Seine Parameter sind dafür geeignet und die Achse ist in einem dafür geeigneten Zustand.
bEnc_RefIndexPositive	BOOL	Ist dieser Parameter gesetzt wird bei einer Referenzfahrt bei der Suche des Referenzindex

Name	Typ	Beschreibung
		(Nocken) ein positiver, andernfalls ein negativer Stellwert ausgegeben.
bEnc_RefSyncPositive	BOOL	Ist dieser Parameter gesetzt, wird bei einer Referenzfahrt bei der Suche des Referenzpulses (Sync-Puls, Null-Impuls) ein positiver, andernfalls ein negativer Stellwert ausgegeben.
bEnc_Reversed	BOOL	Ist dieser Parameter gesetzt, wird der Positionswert negiert ausgewertet.
bMaxLagEna	BOOL	Dieser Parameter aktiviert die Schlepp-Überwachung.
bPEH_Enable	BOOL	Mit diesem Parameter wird die PEH-Überwachung aktiviert.
bPosCtrlAccEna	BOOL	obsolet, wird in Kürze entfernt.
bSoftEndMaxEna	BOOL	Dieser Parameter aktiviert den oberen Software-Endschalter.
bSoftEndMinEna	BOOL	Dieser Parameter aktiviert den unteren Software-Endschalter.
bTimeBased	BOOL	Wenn dieser Parameter auf TRUE steht, werden Profilberechnungen zeitgesteuert ausgeführt. Der Lageregler ist ständig aktiv.
bLinTabAvailable	BOOL	Ein TRUE hier bedeutet, dass bei der Initialisierung eine Linearisierungstabelle per Zeiger angebunden wurde, die eine erfolgreich ermittelte Kennlinie enthält.
bEnc_NoUpLoad	BOOL	Ist dieser Parameter gesetzt werden auch bei Feldbus-Encodern keine Parameter aus dem Gerät gelesen.
bDrive_NoUpLoad	BOOL	Ist dieser Parameter gesetzt werden auch bei Feldbus-Antrieben und -Ventilen keine Parameter aus dem Gerät gelesen.
bDrivelsHybrid	BOOL	Mit diesem Parameter wird eine servoelektrisch-hydraulische Hybrid-Achse gekennzeichnet. Die erweiterten Parameter in stHybrid werden wirksam und erscheinen im PlcMcManager.
bAlignedStart	BOOL	Ab V3.x.y: Wenn dieser Parameter auf TRUE steht, wird bei einem Start aus einem Schleppfehler ein Sprung der Ausgabe vermieden.
bEncModuloMode	BOOL	reserviert.
stHybrid	ST_TcHybridAxParam	In dieser Struktur sind Parameter für hybride elektrisch-hydraulische Achsen zusammengefasst.
stPctrl	ST_TcPctrlParam	In dieser Struktur sind Parameter zusammengefasst, die für einen Kraft- oder Druckregler verwendet werden können.

**fBrakeDistance:** [mm] Bis zu V3.0.7: Bremsstrecke: In dieser nicht richtungsabhängigen Entfernung zum Ziel wird die aktive profilgesteuerte Stellwertgenerierung eingestellt und optional ein Stillstandslageregler oder ein anderer im Ziel geltender Mechanismus aktiviert.

**fBrakeDeadTime:** [s] Bis zu V3.0.7: Dieser Parameter ermöglicht es, die eingestellte Bremsstrecke einen zur Istgeschwindigkeit proportionalen Anteil zu erweitern.

**fCreepSpeed:** [mm/s] Bis zu V3.0.7: Diese Geschwindigkeit wird nicht richtungsabhängig für die letzte Phase der profilgesteuerten Stellwertgenerierung verwendet.

**fCreepDistance:** [mm] Bis zu V3.0.7: Ab dieser nicht richtungsabhängigen Entfernung zum Ziel wird für die letzte Phase der profilgesteuerten Stellwertgenerierung **fCreepSpeed** als Stellwert verwendet.



Dieser Parameter wird nur von MC\_AxRtPosPiControllerEx\_BkPlcMc() verwendet.

**fValve\_OverlapComp:** [1] Bis zu V3.0.7: Kompensation einer nicht richtungsabhängigen Ventilüberdeckung.

Weitere Informationen zur Achsinbetriebnahme finden Sie unter Setup.

### 4.3.25 ST\_TcHydAxRtData (ab V3.0)

Die Variablen in dieser Struktur geben den Laufzeit-Zustand der Achse wieder.



Die Reihenfolge der Daten ist nicht garantiert.

#### Syntax

```

TYPE ST_TcHydAxRtData :
(* last modification: 02.07.2018 *)
STRUCT
(*-----*)
fActForce:           LREAL := 0.0;
fActiveOverlap:     LREAL := 0.0;
fActPos:             LREAL := 0.0;
fActPosDelta:       LREAL := 0.0;
fActPosOffset:      LREAL := 0.0;
fActPressure:       LREAL := 0.0;
fActPressureA:      LREAL := 0.0;
fActPressureB:      LREAL := 0.0;
fActVelo:           LREAL := 0.0;
fBrakeOffTimer:     LREAL := 0.0;
fBrakeOnTimer:      LREAL := 0.0;
fBrakeSafetyTimer: LREAL := 0.0;
fClampingOutput:    LREAL := 0.0;
fDestAcc:           LREAL := 0.0;
fDestCreepDistanceM: LREAL := 0.0;
fDestCreepDistanceP: LREAL := 0.0;
fDestCreepSpeedM:   LREAL := 0.0;
fDestCreepSpeedP:   LREAL := 0.0;
fDestDec:           LREAL := 0.0;
fDestJerk:          LREAL := 0.0;
fDestPos:           LREAL := 0.0;
fDestRampEnd:       LREAL := 0.0;
fDestSpeed:         LREAL := 0.0;
fDistanceToTarget:  LREAL := 0.0;
fEnc_RefShift:      LREAL := 0.0;
fEnc_ZeroSwap:      LREAL := 0.0;
fGearActive:        LREAL := 0.0;
fGearSetting:       LREAL := 0.0;
fLagCtrlOutput:     LREAL := 0.0;
fLatchedPos:        LREAL := 0.0;
fOilRequirred_A:    LREAL := 0.0;
fOilRequirred_B:    LREAL := 0.0;
fOilUsed_A:         LREAL := 0.0;
fOilUsed_B:         LREAL := 0.0;
fOutput:            LREAL := 0.0;
fOverride:          LREAL := 1.0;
fParamAccTime:      LREAL := 0.0;
    
```

```

fPosError:          LREAL := 0.0;
fSetAcc:            LREAL := 0.0;
fSetPos:           LREAL := 0.0;
fSetPressure:      LREAL := 0.0;
fSetSpeed:         LREAL := 0.0;
fSetSpeedOld:      LREAL := 0.0;
fSetVelo:          LREAL := 0.0;
fStartPos:         LREAL := 0.0;
fStartRamp:        LREAL := 0.0;
fStartRampAnchor: LREAL := 0.0;
fSupplyPressure:   LREAL := 0.0;
fTargetPos:        LREAL := 0.0;
fTimerPEH:         LREAL := 0.0;
fTimerTPM:         LREAL := 0.0;
fValvePressure:    LREAL := 0.0;
fVeloError:        LREAL := 0.0;
fBlockDetectDelay: LREAL := 2.0;
(*-----*)
nAxisState:        DWORD := 0;
nCalibrationState: DWORD := 0;
nDeCtrlDWord:     DWORD := 0;
nErrorCode:        DWORD := 0;
nStateDWord:       DWORD := 0;
udiAmpErrorCode:   UDINT;
(*-----*)
iCurrentStep: E_TcMcCurrentStep;
wEncErrMask:       WORD:=0;
wEncErrMaskInv:    WORD:=0;
nDrvWcCount:       INT:=0;
(**)
nEncWcCount:       INT:=0;
nDrvDeviceState:   UINT:=0;
nEncDeviceState:   INT:=0;
(*-----*)
bActPosCams:       BYTE := 0;
bBrakeOff:          BOOL := FALSE;
bBrakeOffInverted: BOOL := FALSE;
bControllable:     BOOL := FALSE;
bCountedCycles:    BYTE := 1;
bCycleCounter:     BYTE := 0;
bDriveResponse:    BOOL := FALSE;
bEncDoLatch:       BOOL := FALSE;
(**)
bEncoderResponse:  BOOL := FALSE;
bEncLatchValid:   BOOL := FALSE;
bLocked_Estop:    BOOL := FALSE;
bParamsUnsave:    BOOL := FALSE;
bReloadParams:     BOOL := FALSE;
bTargeting:        BOOL := FALSE;
bUnalignedOverlap: BOOL := FALSE;
bActPosOffsetEnable: BOOL := FALSE; (* starting with 09.03.2015 *)
(**)
bDriveStartup:     BOOL := FALSE;
bEncAlignRefShift: BOOL := FALSE;
bDrvWcsError:      BOOL := FALSE;
bEncWcsError:      BOOL := FALSE;
bFirstWcs:         BOOL := FALSE;
bChangeCount:      BYTE := 0;
bStartAutoIdent:   BOOL := FALSE;
bParamFileComplete: BOOL := FALSE;
(*-----*)
pMasterRtData:     POINTER TO BYTE;
pMasterParam:      POINTER TO BYTE;
(*-----*)
udiSercDeviceID:   UDINT := 0;
uiSercBoxAddr:     UINT := 0;
uiSercPort:        UINT := 0;
(*-----*)
stPosCtrlr: stbkplcinternal_cplxctrl;
stVeloCtrlr: stbkplcinternal_cplxctrl;
(*-----*)
sTopBlockName:     STRING(87) := '';

stHybrid:          ST_TcHybridAxRtData;
(*-----*)
END_STRUCT
END_TYPE

```

**Parameter**

Name	Typ	Beschreibung
fActForce	LREAL	[N, kN] Istkraft des Zylinders. Dieser Wert wird üblicherweise durch einen Erfassungsbaustein-Baustein für die Erfassung von <u>Kraft- oder Druckistwerten</u> [► 20] ermittelt.
fActiveOverlap	LREAL	[1] Die aktuelle Ausgabe der Überdeckungs-Kompensation. Eine Ausgabe-Variable der Profilgeneratoren.
fActPos	LREAL	[mm] Die aktuelle Istposition der Achse. Dieser Wert wird üblicherweise durch einen Encoder-Baustein ermittelt.
fActPosDelta	LREAL	[mm] Die Änderung der Istposition relativ zum vorangehenden Zyklus.
fActPosOffset	LREAL	[mm] Der Offset für die Istwert-Beeinflussung. Wenn bActPosOffsetEnable auf TRUE steht wird dieser Offset in die fActPos addiert. Wenn sich fActPosOffset ändert wird dadurch die fActVelo nicht beeinflusst.  Durch bActPosOffsetEnable:=TRUE wird fActPosOffset unmittelbar und ohne Rampe wirksam.  <u>Beachten Sie die Informationen.</u> [► 137]  Beispiel: Bei einer Referenzposition von 100.0 mm und einem Offset von 1.0 mm wird die Istposition am Ort des Null-Impulses auf 101.0 mm gesetzt. Wenn anschließend die Beeinflussung deaktiviert oder auf 0.0 mm gesetzt wird zeigt die Istposition am Ort des Null-Impulses den Wert 100.0 mm, wie sie es auch beim Homing ohne Beeinflussung gezeigt hätte.
fActPressure	LREAL	[bar] Istdruck im Zylinder. Dieser Wert wird üblicherweise durch einen Erfassungsbaustein-Baustein für die Erfassung von <u>Kraft- oder Druckistwerten</u> [► 20] ermittelt.
fActPressureA	LREAL	[bar] Istdruck auf der A-Seite des Zylinders. Dieser Wert wird üblicherweise durch einen Erfassungsbaustein-Baustein für die Erfassung von <u>Kraft- oder Druckistwerten</u> [► 20] ermittelt.
fActPressureB	LREAL	[bar] Istdruck auf der B-Seite des Zylinders. Dieser Wert wird üblicherweise durch einen Erfassungsbaustein-Baustein für die Erfassung von <u>Kraft- oder Druckistwerten</u> [► 20] ermittelt.
fActVelo	LREAL	[mm/s] Die aktuelle Istgeschwindigkeit der Achse. Dieser Wert wird üblicherweise durch einen Encoder-Baustein ermittelt.
fBrakeOffTimer	LREAL	
fBrakeOnTimer	LREAL	
fBrakeSafetyTimer	LREAL	
fClampingOutput	LREAL	[V] Eine Ausgabe-Variable der Profilgeneratoren.
fDestAcc	LREAL	[mm/s <sup>2</sup> ] Die vom aktuellen oder dem zuletzt abgearbeiteten Bewegungsauftrag vorgegebene Beschleunigung.
fDestCreepDistance	LREAL	[mm] Bis V3.0.7: Die Schleichstrecke.

Name	Typ	Beschreibung
fDestCreepDistanceM	LREAL	[mm] Ab V3.0.8: Die Schleichstrecke in negativer Richtung.
fDestCreepDistanceP	LREAL	[mm] Ab V3.0.8: Die Schleichstrecke in positiver Richtung.
fDestCreepSpeed	LREAL	[mm/s] Bis V3.0.7: Die Schleichgeschwindigkeit.
fDestCreepSpeedM	LREAL	Ab V3.0.8: Die Schleichgeschwindigkeit in negativer Richtung.
fDestCreepSpeedP	LREAL	[mm/s] Ab V3.0.8: Die Schleichgeschwindigkeit in positiver Richtung.
fDestDec	LREAL	[mm/s <sup>2</sup> ] Die vom aktuellen oder dem zuletzt abgearbeiteten Bewegungsauftrag vorgegebene Verzögerung.
fDestJerk	LREAL	[mm/s <sup>3</sup> ] Der vom aktuellen oder dem zuletzt abgearbeiteten Bewegungsauftrag vorgegebene Ruck.
fDestPos	LREAL	[mm] Die aktuell wirksame Zielposition.
fDestRampEnd	LREAL	
fDestSpeed	LREAL	[mm/s] Die vom aktuellen oder dem zuletzt abgearbeiteten Bewegungsauftrag vorgegebene Geschwindigkeit.
fDistanceToTarget	LREAL	[mm] Der aktuelle Restweg der Achse. Dieser Wert wird üblicherweise durch einen Generator-Baustein ermittelt.
fEnc_RefShift	LREAL	[mm] Der Offset zwischen dem umgerechneten (eventuell intern verlängerten) Zählerstand einer Inkrementalencoder-Eingangsklemme und der Istposition der Achse. Dieser Offset wird durch eine Referenzfahrt z.B. mit einem <a href="#">MC_Home BkPlcMc</a> [▶ 68] Baustein ermittelt oder mit einem <a href="#">MC_SetPosition BkPlcMc</a> [▶ 43] Baustein manipuliert.
fEnc_ZeroSwap	LREAL	
fGearActive	LREAL	
fGearSetting	LREAL	
fLagCtrlOutput	LREAL	[1] Die normierte Ausgabe des Lagereglers. Eine Ausgabe-Variable der Profilgeneratoren.
fLatchedPos	LREAL	[mm] Die Position (unter Berücksichtigung von geltenden Offsets) an der ein Homing erfolgte oder die Komponenten der Istwerterfassung (Encoder, E/A-Elektronik) eingeschaltet wurden.
fOilRequirred_A	LREAL	[l/min] Der unter Berücksichtigung der Sollgeschwindigkeit berechnete Ölbedarf der A-Seite.
fOilRequirred_B	LREAL	[l/min] Der unter Berücksichtigung der Sollgeschwindigkeit berechnete Ölbedarf der B-Seite.
fOilUsed_A	LREAL	[l/min] Der unter Berücksichtigung der Istgeschwindigkeit berechnete Ölverbrauch der A-Seite.
fOilUsed_B	LREAL	[l/min] Der unter Berücksichtigung der Istgeschwindigkeit berechnete Ölverbrauch der B-Seite.

Name	Typ	Beschreibung
fOutput	LREAL	[1] Die auszugebende Stellgröße. Über diese Variable kommunizieren der MC_AxRtFinish_BkPlcMc [▶ 230] und der MC_AxRtDrive_BkPlcMc [▶ 172] Baustein.
fOverride	LREAL	[1] Der aktuelle Achs-Geschwindigkeitsoverride.
fParamAccTime	LREAL	
fPosError	LREAL	[mm] Der aktuelle Lagefehler der Achse.
fSetAcc	LREAL	[mm/s <sup>2</sup> ] Der aktuelle Beschleunigungsstellwert. Eine Ausgabe-Variable der Profilgeneratoren.
fSetPos	LREAL	[mm] Der aktuelle Lagesollwert der Achse.
fSetPressure	LREAL	[bar] Hier ist der Sollwert einer optionalen Druck- oder Kraftregelung abzulegen.
fSetSpeed	LREAL	[mm/s] Die normierte Sollgeschwindigkeit der Achse. Eine Ausgabe-Variable der Profilgeneratoren.
fSetSpeedOld	LREAL	
fSetVelo	LREAL	
fStartPos	LREAL	[mm] Die Startposition des aktuellen oder des zuletzt abgearbeiteten Bewegungsauftrag.
fStartRamp	LREAL	
fStartRampAnchor	LREAL	
fSupplyPressure	LREAL	[bar] Versorgungsdruck. Dieser Wert wird üblicherweise durch einen Erfassungsbaustein-Baustein für die Erfassung von <u>Kraft- oder Druckistwerten</u> [▶ 20] ermittelt.
fTargetPos	LREAL	[mm] Die vom aktuellen oder dem zuletzt abgearbeiteten Bewegungsauftrag vorgegebene Zielposition.
fTimerPEH	LREAL	
fTimerTPM	LREAL	
fValvePressure	LREAL	[bar] Druckabfall am Ventil. Dieser Wert wird üblicherweise durch einen Erfassungsbaustein-Baustein für die Erfassung von <u>Kraft- oder Druckistwerten</u> [▶ 20] ermittelt.
fVeloError	LREAL	
fBlockDetectDelay	LREAL	[s] Die Verzögerungszeit für das Erkennen des Blocks beim Homing on Block. Dieser Wert ist mit 2.0 Sekunden initialisiert um das Default-Verhalten früherer Versionen abzubilden. Wird eine andere Zeit gewünscht muss sie vor dem Start des Homings aktualisiert sein. Wird beim Starten des Homings ein Wert kleiner Zykluszeit erkannt wird automatisch der Default-Wert 2.0 Sekunden eingetragen. Dieser Wert wird nicht als Parameter gespeichert. Diese Variable steht unter TC2 ab 12.10.2017 mit einer V3.0.41 zur Verfügung.
nAxisState	DWORD	Der Bewegungszustand der Achse.
nCalibrationState	DWORD	Der aktuelle Zustand einer Referenzfahrt.
nDeCtrlDWord	DWORD	Die <u>Kontroll-Signale</u> [▶ 320] der Achse.
nErrorCode	DWORD	Der aktuelle <u>ErrorCode</u> [▶ 321] der Achse.
nStateDWord	DWORD	Die <u>Status-Signale</u> [▶ 320] der Achse.
udiAmpErrorCode	UDINT	

Name	Typ	Beschreibung
iCurrentStep	E_TcMcCurrentStep	Der interne Zustand der Stellwertgeneratoren. Werte aus <a href="#">E_TcMcCurrentStep</a> [ <a href="#">▶ 88</a> ].
wEncErrMask	WORD	
wEncErrMaskInv	WORD	
nDrvWcCount	INT	
nEncWcCount	INT	
nDrvDeviceState	UINT	
nEncDeviceState	INT	
bActPosCams	BYTE	Die aktuellen Positionsnocken der Achse. Dieser Wert wird nur verwendet, wenn als Encoder-Typ <a href="#">iTcMc_EncoderDigCam</a> eingestellt ist.
bBrakeOff	BOOL	Das Steuersignal für eine externe Bremse. Eine Ausgabe-Variable der Profilgeneratoren.
bBrakeOffInverted	BOOL	Das invertierte bBrakeOff Signal.
bControllable	BOOL	
bCountedCycles	BYTE	
bCycleCounter	BYTE	
bDriveResponse	BOOL	
bEncDoLatch	BOOL	Mit diesem Signal kommunizieren der <a href="#">MC_Home_BkPlcMc</a> [ <a href="#">▶ 68</a> ] und der <a href="#">MC_AxRtEncoder_BkPlcMc</a> [ <a href="#">▶ 183</a> ] Baustein der Achse während einer Referenzfahrt.
bEncoderResponse	BOOL	
bEncLatchValid	BOOL	Mit diesem Signal kommunizieren der <a href="#">MC_Home_BkPlcMc</a> [ <a href="#">▶ 68</a> ] und der <a href="#">MC_AxRtEncoder_BkPlcMc</a> [ <a href="#">▶ 183</a> ] Baustein der Achse während einer Referenzfahrt.
bLocked_Estop	BOOL	Durch ein TRUE in dieser Variablen werden die Stellwertgeneratoren daran gehindert, den Zustand <a href="#">iTcHydStateEmergencyBreak</a> / <a href="#">McState_Errorstop</a> zu verlassen, obwohl die Ausgaben an den Antrieb auf 0 abgebaut werden. Verwendet von <a href="#">MC_EmergencyStop_BkPlcMc</a> [ <a href="#">▶ 56</a> ] und <a href="#">MC_ImmediateStop_BkPlcMc</a> [ <a href="#">▶ 71</a> ].
bParamsUnsave	BOOL	Die Bausteine <a href="#">MC_WriteParameter_BkPlcMc</a> [ <a href="#">▶ 47</a> ] und <a href="#">MC_WriteBoolParameter_BkPlcMc</a> [ <a href="#">▶ 45</a> ] setzen dieses Flag wenn sie den Wert eines Parameters verändern. Ein <a href="#">MC_AiParamSave_BkPlcMc</a> [ <a href="#">▶ 271</a> ] Baustein löscht das Flag beim erfolgreichen Speichern der Parameter. Im Online Modus des <a href="#">PlcMcManagers</a> [ <a href="#">▶ 350</a> ] wird dieses Flag für die Status-Anzeige genutzt.
bReloadParams	BOOL	
bTargeting	BOOL	
bUnalignedOverlap	BOOL	Hier wird die Charakteristik der Überdeckungs-Kompensation festgelegt.
bActPosOffsetEnable	BOOL	Durch ein TRUE in dieser Variablen wird die Istwert-Beeinflussung aktiviert. Siehe auch unter <a href="#">fActPosOffset</a> .
bDriveStartup	BOOL	
bEncAlignRefShift	BOOL	reserviert.
bDrvWcsError	BOOL	



Name	Typ	Beschreibung
bEncWcsError	BOOL	
bFirstWcs	BOOL	
bChangeCount	BYTE	Dieser Wert wird bei jeder Parameteränderung erhöht.
bStartAutolident	BOOL	
bParamFileComplete	BOOL	Dieses Flag wird gesetzt, wenn beim Laden der Parameter am Dateiende eine entsprechende Kennung gefunden und der CRC-Check erfolgreich war.
pMasterRtData	POINTER TO BYTE	
pMasterParam	POINTER TO BYTE	
udiSercDeviceID	UDINT	
uiSercBoxAddr	UINT	
uiSercPort	UINT	
stPosCtrlr	stbkplcinternal_cplxctrl	
stVeloCtrlr	stbkplcinternal_cplxctrl	
sTopBlockName	STRING	Die meisten von der Applikation direkt aufgerufenen Bausteine der Bibliothek tragen hier eine Debug-Kennung ein.
stHybrid	ST_TcHybridAxRtData	Die erweiterten Zustandsdaten für servoelektrisch-hydraulische Hybrid-Achsen.

**i Informationen für fActPosOffset**

- Ist die Istwert-Beeinflussung während eines Homings aktiv wird fActPosOffset beim Setzen der Ist-Position berücksichtigt.
- Diese Funktion ist nur für die folgenden Encoder-Typen realisiert: iTcMc\_EncoderCoE\_DS406, iTcMc\_EncoderEL3255, iTcMc\_EncoderSim, iTcMc\_EncoderEL5101, iTcMc\_EncoderKL5101, iTcMc\_EncoderKL5111, iTcMc\_EncoderEL5001, iTcMc\_EncoderKL5001, iTcMc\_EncoderKL3002, iTcMc\_EncoderEL3102, iTcMc\_EncoderKL3042, iTcMc\_EncoderKL3062, iTcMc\_EncoderEL3142, iTcMc\_EncoderEM8908\_A, iTcMc\_EncoderEL3162, iTcMc\_EncoderKL3162.
- Ist für ein zu einem dieser Typen kompatibles E/A-Gerät einer der aufgeführten Typen eingestellt, wird die beschriebene Funktion ebenfalls realisiert.

**i** Alle anderen Elemente dieser Struktur sind für eine interne Verwendung reserviert. Sie sind nicht garantiert und dürfen nicht von der Applikation verwendet oder verändert werden.

**4.3.26 ST\_TcMcAuxDataLabels (ab V3.0)**

In dieser Struktur werden die Beschriftungstexte der kundenspezifischen Achsparameter abgelegt. Eine Struktur dieses Typs kann durch einen [MC\\_AxUtiStandardInit\\_BkPlcMc \[► 238\]](#) Baustein durch einen Pointer in der [Axis\\_Ref\\_BkPlcMc \[► 84\]](#) Struktur mit der Achse verbunden werden.

**Syntax**

```

TYPE ST_TcMcAuxDataLabels :
STRUCT
    stLabel:          ARRAY [1..20] OF STRING(20);
END_STRUCT
END_TYPE
    
```

**Parameter**

Name	Typ	Beschreibung
stLabel	ARRAY	Die Beschriftungstexte

### 4.3.27 ST\_TcPlcDeviceInput (ab V3.0)

Diese Struktur enthält die Eingangsabbild-Variablen einer Achse.

#### Syntax

```

TYPE ST_TcPlcDeviceInput :
STRUCT
  uiCount:          UINT:=0;
  uiLatch:          UINT:=0;
  usiStatus:        USINT:=0;

  uiPZDL_RegDaten:  UINT:=0;
  uiPZDH:           UINT:=0;
  usiRegStatus:     USINT:=0;

  udiCount:         UDINT:=0;
  uiStatus:         UINT:=0;

  bTerminalState:   BYTE:=0;
  uiTerminalData:   WORD:=0;
  uiTerminalState2:WORD:=0;

  bDigInA:          BOOL:=FALSE;
  bDigInB:          BOOL:=FALSE;

  bDigCamMM:        BOOL:=FALSE;
  bDigCamM:         BOOL:=FALSE;
  bDigCamP:         BOOL:=FALSE;
  bDigCamPP:        BOOL:=FALSE;

  DriveError:       UDINT:=0;
  ActualPos:        ARRAY [0..1] OF UINT:=0;
  DriveState:       ARRAY [0..3] OF BYTE:=0;

  S_iReserve:       INT:=0;
  S_DiReserve:      ARRAY [1..9] OF DINT:=0;

  CiA_Reserve:      ARRAY [1..8] OF UINT:=0;

  bPowerOk:         BOOL:=FALSE;
  bEnAck:           BOOL:=FALSE;

  wDriveDevState:   WORD:=0;
  wDriveWcState:    BYTE:=0;
  wEncDevState:     WORD:=0;
  wEncWcState:      BYTE:=0;
  uiDriveBoxState:  UINT:=0;
  uiEncBoxState:    UINT:=0;

  sEncAdsAddr:      ST_TcPlcAdsAddr;
  nEncAdsChannel:   BYTE:=0;
  sDrvAdsAddr:      ST_TcPlcAdsAddr;
  nDrvAdsChannel:   BYTE:=0;

  nReserve:         ARRAY [1..20] OF BYTE;
END_STRUCT
END_TYPE

```

#### Parameter

Name	Typ	Beschreibung
uiCount	UINT	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderEL3102, iTcMc_EncoderEL3142, iTcMc_EncoderEL5101, iTcMc_EncoderKL2521, iTcMc_EncoderKL2531, iTcMc_EncoderKL2541, iTcMc_EncoderKL3002, iTcMc_EncoderKL3042, iTcMc_EncoderKL3062, iTcMc_EncoderKL3162, iTcMc_EncoderKL5101, iTcMc_EncoderKL5111, iTcMc_EncoderM2510, iTcMc_EncoderM3120, iTcMc_DriveKL2531, iTcMc_DriveKL2541.

Name	Typ	Beschreibung
uiLatch	UINT	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderEL5101, iTcMc_EncoderKL5101, iTcMc_EncoderKL5111.
usiStatus	USINT	Verwendet für Gerätezustandsinformation. Benutzt bei iTcMc_EncoderEL5101, iTcMc_EncoderKL3002, iTcMc_EncoderKL3042, iTcMc_EncoderKL3062, iTcMc_EncoderKL3162, iTcMc_EncoderKL5101, iTcMc_EncoderKL5111, iTcMc_EncoderM3120.
uiPZDL_RegDaten	UINT	Verwendet für Positionserfassung und Parameter-Kommunikation. Benutzt bei iTcMc_EncoderKL5001.
uiPZDH	UINT	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderKL5001.
usiRegStatus	USINT	Verwendet für Gerätezustandsinformation. Benutzt bei iTcMc_EncoderEL5001, iTcMc_EncoderKL5001.
udiCount	UDINT	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderEL5001.
uiStatus	UINT	Verwendet für Gerätezustandsinformation. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110.
bTerminalState	BYTE	Verwendet für Parameter-Kommunikation. Benutzt bei iTcMc_EncoderKL2521, iTcMc_EncoderKL2531, iTcMc_EncoderKL2541, iTcMc_DriveEL4132, iTcMc_DriveKL2521, iTcMc_DriveKL2531, iTcMc_DriveKL2541, iTcMc_DriveKL4032.
uiTerminalData	WORD	reserviert.
uiTerminalState2	WORD	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderKL2541.
bDigInA	BOOL	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderDigIncrement.
bDigInB	BOOL	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderDigIncrement.
bDigCamMM	BOOL	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderDigCam.
bDigCamM	BOOL	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderDigCam.
bDigCamP	BOOL	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderDigCam.
bDigCamPP	BOOL	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderDigCam.
DriveError	UDINT	Verwendet für Gerätezustandsinformation. Benutzt bei iTcMc_EncoderAx2000_B200, iTcMc_EncoderAx2000_B900.
ActualPos	ARRAY	Verwendet für Positionserfassung. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_EncoderAx2000_B200, iTcMc_EncoderAx2000_B900.
DriveState	ARRAY	Verwendet für Gerätezustandsinformation. Benutzt bei iTcMc_EncoderAx2000_B200, iTcMc_EncoderAx2000_B900.
S_iReserve	INT	reserviert.
S_DiReserve	ARRAY	reserviert.
CiA_Reserve	ARRAY	reserviert.

Name	Typ	Beschreibung
bPowerOk	BOOL	Optional verwendet für die Überwachung eines Netz-Schützes. Benutzt bei iTcMc_DriveAx2000_B110, iTcMc_EncoderAx2000_B200, iTcMc_EncoderAx2000_B900.
bEnAck	BOOL	reserviert.
wDriveDevState	WORD	reserviert.
wDriveWcState	BYTE	Verwendet für die Überwachung der Verbindung zum Steller. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110.
wEncDevState	WORD	reserviert.
wEncWcState	BYTE	Verwendet für die Überwachung der Verbindung zum Encoder. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110, iTcMc_EncoderEL3102, iTcMc_EncoderEL3142, iTcMc_EncoderEL5001, iTcMc_EncoderEL5101.
uiDriveBoxState	UINT	Verwendet für die Überwachung der Verbindung zum Steller. Benutzt bei iTcMc_DriveAx2000_B200, iTcMc_DriveAx2000_B900.
uiEncBoxState	UINT	Verwendet für die Überwachung der Verbindung zum Encoder. Benutzt bei iTcMc_EncoderAx2000_B200, iTcMc_EncoderAx2000_B900.
sEncAdsAddr	ST_TcPlcAdsAddr	Verwendet für Parameter-Kommunikation. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110, iTcMc_EncoderEL3102, iTcMc_EncoderEL3142, iTcMc_EncoderEL5001, iTcMc_EncoderEL5101.
nEncAdsChannel	BYTE	Verwendet für Parameter-Kommunikation. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110.
sDrvAdsAddr	ST_TcPlcAdsAddr	Verwendet für Parameter-Kommunikation. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110.
nDrvAdsChannel	BYTE	Verwendet für Parameter-Kommunikation. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110.
nReserve	ARRAY	reserviert.

### 4.3.28 ST\_TcPlcDeviceOutput (ab V3.0)

Diese Struktur enthält die Ausgangsabbild-Variablen einer Achse.

#### Syntax

```

TYPE ST_TcPlcDeviceOutput :
STRUCT
  nDacOut:          INT:=0;
  bDigOutAp:        BOOL:=FALSE;
  bDigOutAn:        BOOL:=FALSE;
  bDigOutBp:        BOOL:=FALSE;
  bDigOutBn:        BOOL:=FALSE;
  uiCount:          UINT:=0;
  uiDacOutA:        UINT:=0;
  uiDacOutB:        UINT:=0;
  bMovePos:         BOOL:=FALSE;
  bMoveNeg:         BOOL:=FALSE;
  bBrakeOff:        BOOL:=FALSE;
  bBrakeOffInverted:BOOL:=FALSE;
  DriveCtrl:        ARRAY [0..3] OF BYTE:=0;
  NominalVelo:      DINT:=0;
  uiDriveCtrl:      UINT:=0;
  S_iReserve:       ARRAY [1..2] OF INT:=0;
  S_DiReserve:      ARRAY [1..7] OF DINT:=0;

```

```

CiA_Reserve:    ARRAY [1..7] OF UINT:=0;
bPowerOn:      BOOL:=FALSE;
bEnable:       BOOL:=FALSE;
bEnablePos:    BOOL:=FALSE;
bEnableNeg:    BOOL:=FALSE;
nResetState:   BYTE:=0;
usiCtrl:       USINT:=0;
uiTerminalData: WORD:=0;
bTerminalCtrl: BYTE:=0;
uiTerminalCtrl2: WORD:=0;
nReserve:      ARRAY [1..20] OF BYTE;
END_STRUCT
END_TYPE

```

**Parameter**

Name	Typ	Beschreibung
nDacOut	INT	Verwendet für Stellwertausgabe oder Parameter-Kommunikation. Benutzt bei iTcMc_EncoderKL2531, iTcMc_EncoderKL2541, iTcMc_DriveEL4132, iTcMc_DriveKL2521, iTcMc_DriveKL2531, iTcMc_DriveKL2541, iTcMc_DriveKL4032, iTcMc_DriveM2400_Dn.
bDigOutAp	BOOL	Verwendet für Stellwertausgabe. Benutzt bei iTcMc_DriveLowCostStepper.
bDigOutAn	BOOL	Verwendet für Stellwertausgabe. Benutzt bei iTcMc_DriveLowCostStepper.
bDigOutBp	BOOL	Verwendet für Stellwertausgabe. Benutzt bei iTcMc_DriveLowCostStepper.
bDigOutBn	BOOL	Verwendet für Stellwertausgabe. Benutzt bei iTcMc_DriveLowCostStepper.
uiCount	UINT	reserviert.
uiDacOutA	UINT	Verwendet für Stellwertausgabe. Benutzt bei iTcMc_EncoderIx2512_1Coil, iTcMc_EncoderIx2512_2Coil.
uiDacOutB	UINT	Verwendet für Stellwertausgabe. Benutzt bei iTcMc_EncoderIx2512_2Coil.
bMovePos	BOOL	reserviert.
bMoveNeg	BOOL	reserviert.
bBrakeOff	BOOL	reserviert.
bBrakeOffInverted	BOOL	reserviert.
DriveCtrl	ARRAY	Verwendet für Gerätesteuersignale. Benutzt bei iTcMc_EncoderAx2000_B200, iTcMc_DriveAx2000_B200, iTcMc_EncoderAx2000_B900, iTcMc_DriveAx2000_B900.
NominalVelo	DINT	Verwendet für Stellwertausgabe. Benutzt bei iTcMc_DriveAx2000_B110, iTcMc_EncoderAx2000_B200, iTcMc_EncoderAx2000_B900.
uiDriveCtrl	UINT	Verwendet für Gerätesteuersignale. Benutzt bei iTcMc_EncoderAx2000_B110, iTcMc_DriveAx2000_B110.
S_iReserve	ARRAY	reserviert.
S_DiReserve	ARRAY	reserviert.
CiA_Reserve	ARRAY	reserviert.
bPowerOn	BOOL	Optional verwendet für die Steuerung eines Netz-Schützes. Benutzt bei iTcMc_DriveAx2000_B110, iTcMc_EncoderAx2000_B200, iTcMc_EncoderAx2000_B900.
bEnable	BOOL	reserviert.
bEnablePos	BOOL	reserviert.
bEnableNeg	BOOL	reserviert.
nResetState	BYTE	reserviert.

Name	Typ	Beschreibung
usiCtrl	USINT	Verwendet für Gerätesteuersignale oder Parameter-Kommunikation. Benutzt bei iTcMc_EncoderEL5101, iTcMc_EncoderKL3002, iTcMc_EncoderKL3042, iTcMc_EncoderKL3062, iTcMc_EncoderKL3162, iTcMc_EncoderKL5101, iTcMc_EncoderKL5111, iTcMc_EncoderM3120.
uiTerminalData	WORD	Verwendet für Parameter-Kommunikation. Benutzt bei iTcMc_EncoderKL2521, iTcMc_EncoderKL5001, iTcMc_EncoderKL5101, iTcMc_EncoderKL5111, iTcMc_DriveEL4132, iTcMc_DriveKL2521, iTcMc_DriveKL4032.
bTerminalCtrl	BYTE	Verwendet für Parameter-Kommunikation. Benutzt bei iTcMc_EncoderKL2521, iTcMc_EncoderKL2531, iTcMc_EncoderKL2541, iTcMc_DriveEL4132, iTcMc_DriveKL2521, iTcMc_DriveKL2531, iTcMc_DriveKL2541, iTcMc_DriveKL4032.
uiTerminalCtrl2	WORD	Verwendet für Gerätesteuersignale. Benutzt bei iTcMc_EncoderKL2541, iTcMc_DriveKL2531, iTcMc_DriveKL2541.
nReserve	ARRAY	reserviert.

### 4.3.29 ST\_TcPlcMcLogBuffer (ab V3.0)

Eine Variable mit dieser Struktur bildet den LogBuffer der Library. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der [Knowledge Base](#) [► 302].



Die Daten in dieser Struktur dürfen nicht durch die Applikation verändert werden.

#### Syntax

```

TYPE ST_TcMcLogBuffer:
STRUCT
  ReadIdx:      INT:=0;
  WriteIdx:     INT:=0;
  MessageArr:   ARRAY [0..19] OF ST_TcPlcMcLogEntry;
END_STRUCT
END_TYPE

```

#### Parameter

Name	Typ	Beschreibung
ReadIdx	INT	Der Leseindex des Buffers.
WriteIdx	INT	Der Schreibindex des Buffers.
MessageArr	ARRAY	Die aktuell gespeicherten Meldungen.

[ST\\_TcPlcMcLogEntry](#) [► 142]

### 4.3.30 ST\_TcPlcMcLogEntry (ab V3.0)

Eine Variable mit dieser Struktur enthält eine Meldung des LogBuffer der Library. Sie wird als Bestandteil in [ST\\_TcPlcMcLogBuffer](#) [► 142] verwendet. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der [Knowledge Base](#) [► 302].



Die Daten in dieser Struktur dürfen nicht durch die Applikation verändert werden.

**Syntax**

```

TYPE ST_TcPlcMcLogEntry:
STRUCT
  TimeLow:      UDINT:=0;
  TimeHigh:     UDINT:=0;
  LogLevel:     DWORD:=0;
  Source:       DWORD:=0;
  Msg:          STRING(255);
  ArgType:      INT:=0;
  diArg:        DINT:=0;
  lrArg:        LREAL:=0;
  sArg:         STRING(255);
END_STRUCT
END_TYPE
    
```

**Parameter**

Name	Typ	Beschreibung
TimeLow	UDINT	Der Timestamp der Meldung. Hier wird der Zeitpunkt festgehalten, zu dem die Meldung erzeugt wurde.
TimeHigh	UDINT	
LogLevel	DWORD	Eine Kennzeichnung der Dringlichkeit der Meldung. Hier sollten nur Werte aus einem festgelegten Zahlenvorrat erscheinen.
Source	DWORD	Eine Kennzeichnung der Quelle der Meldung. Hier sollten nur Werte aus einem festgelegten Zahlenvorrat erscheinen.
Msg	STRING	Der Meldungstext mit einem optionalen Platzhalter für einen variablen Bestandteil.
ArgType	INT	Der Typ des optionalen Bestandteils.
diArg	DINT	Wenn ein optionaler Bestandteil vom Typ DINT verwendet wird, befindet sich sein Wert hier.
lrArg	LREAL	Wenn ein optionaler Bestandteil vom Typ LREAL verwendet wird, befindet sich sein Wert hier.
sArg	STRING	Wenn ein optionaler Bestandteil vom Typ STRING verwendet wird, befindet sich sein Wert hier.

**4.3.31 ST\_TcPlcRegDataItem (ab V3.0.7)**

Diese Struktur enthält einen Parameter für eine KL-Klemme. Ein ARRAY von Elementen dieses Typs bildet den Typ [ST\\_TcPlcRegDataTable](#) [[▶ 144](#)].

**Syntax**

```

TYPE ST_TcPlcRegDataItem :
STRUCT
  Access:      INT:=0;
  Select:      INT:=-1;
  RegData:     WORD:=0;
END_STRUCT
END_TYPE
    
```

**Parameter**

Name	Typ	Beschreibung
Access	INT	Hier wird die Art der auszuführenden Operation kodiert. Details sind unter <a href="#">MC_AxUtiUpdateRegDriveTerm_BkPlcMc</a> [ <a href="#">▶ 278</a> ] oder <a href="#">MC_AxUtiUpdateRegEncTerm_BkPlcMc</a> [ <a href="#">▶ 280</a> ] zu finden.
Select	INT	Die Adresse des Registers in der Klemme.
RegData	WORD	Der für die auszuführende Operation zu verwendende Parameter.

### 4.3.32 ST\_TcPlcRegDataTable (ab V3.0.7)

Diese Struktur enthält einen Parametersatz für eine KL-Klemme. Eine solche Tabelle wird von [MC\\_AxUtilUpdateRegDriveTerm\\_BkPlcMc \[► 278\]](#) oder [MC\\_AxUtilUpdateRegEncTerm\\_BkPlcMc \[► 280\]](#) Bausteinen verarbeitet.

#### Syntax

```
TYPE ST_TcPlcRegDataTable :
STRUCT
  RegDataItem:    ARRAY [1..64] OF ST_TcPlcRegDataItem;
END_STRUCT
END_TYPE
```

#### Parameter

Name	Typ	beschreibung
RegDataItem	ARRAY	

### 4.3.33 ST\_TcHybridAxParam (ab V3.0.44)

Diese Struktur enthält zusätzliche Parameter der servoelektrisch-hydraulischen Achse. Unter Setup (teilweise in Vorbereitung) werden geeignete Vorgehensweisen zur Achsinbetriebnahme vorgestellt.



Die Reihenfolge der Parameter ist nicht garantiert.

#### Syntax

```
TYPE ST_TcHybridAxParam :
(* last modification: 20.02.2019 *)
STRUCT
  fPump_N_max:    LREAL;
  fPump_N_min:    LREAL;

  fPump_P_max:    LREAL;
  fPump_P_min:    LREAL;

  fPump_Q_fast_P: LREAL;
  fPump_Q_slow_P: LREAL;

  fPump_Q_fast_M: LREAL;
  fPump_Q_slow_M: LREAL;

  fPump_Q_leak:   LREAL;

  fPump_Enc_Offset: LREAL;

  fCylinder_A_addP: LREAL;
  fCylinder_A_addM: LREAL;

  fRampTime:     LREAL;

  fAside_PrsScaling: LREAL;
  fBside_PrsScaling: LREAL;
  fSystem_PrsScaling: LREAL;

  nPumpCavities: DINT;
  nConcept:      DINT;

  nPump_EncType: E_TcMcEncoderType:=iTcMc_EncoderSim;

  bRegenerative:  BOOL;
  bVirtual_A_addP: BOOL;
  bVirtual_A_addM: BOOL;
  bAside_PrsHiResADC: BOOL;
  bBside_PrsHiResADC: BOOL;
  bSystem_PrsHiResADC: BOOL;

END_STRUCT
END_TYPE
```



**Parameter**

Name	Typ	Beschreibung
fPump_N_max	LREAL	[rpm] Die minimal und maximal zulässigen Drehzahlen der Pumpe.
fPump_N_min	LREAL	
fPump_P_max	LREAL	[bar] Die minimal und maximal zulässigen Betriebsdruck der Pumpe.
fPump_P_min	LREAL	
fPump_Q_fast_P	LREAL	[cm <sup>3</sup> /U] Die umdrehungsbezogene Fördermenge der Pumpe in der Eilgang- bzw. Kraftgang-Schaltung am in positiver Richtung wirkenden Zylinderanschluß.
fPump_Q_slow_P	LREAL	
fPump_Q_fast_M	LREAL	[cm <sup>3</sup> /U] Die umdrehungsbezogene Fördermenge der Pumpe in der Eilgang- bzw. Kraftgang-Schaltung am in negativer Richtung wirkenden Zylinderanschluß.
fPump_Q_slow_M	LREAL	
fPump_Q_leak	LREAL	reserviert.
fPump_Enc_Offset	LREAL	reserviert.
fCylinder_A_addP	LREAL	Wenn situationsabhängig in positiver Wirkungsrichtung eine für den Ölbedarf wirksame Fläche zugeschaltet wird, ist sie hier anzugeben. Hierbei kann es sich auch um einen Ölbedarf handeln, der von einer scheinbaren Fläche gefordert wird, tatsächlich aber am Zylinder vorbeigeleitet wird. In diesem Fall ist die Fläche als „virtuell“ zu kennzeichnen.
fCylinder_A_addM	LREAL	Wenn situationsabhängig in negativer Wirkungsrichtung eine für den Ölbedarf wirksame Fläche zugeschaltet wird, ist sie hier anzugeben. Hierbei kann es sich auch um einen Ölbedarf handeln, der von einer scheinbaren Fläche gefordert wird, tatsächlich aber am Zylinder vorbeigeleitet wird. In diesem Fall ist die Fläche als „virtuell“ zu kennzeichnen.
fRampTime	LREAL	Bei einem Schaltvorgang zwischen Eilgang und Kraftgang werden der Gewichtungsfaktor für die Geschwindigkeitsausgabe und die maximal erreichbare Geschwindigkeit geändert. Wenn dies nicht sprunghaft erfolgen soll, kann hier eine Rampe festgelegt werden.
fAside_PrsScaling	LREAL	Hier sind die Skalierungsdrücke für die A-seitige, die B-seitige und die System-Druckerfassung einzustellen.
fBside_PrsScaling	LREAL	
fSystem_PrsScaling	LREAL	
nPumpCavities	DINT	Hier ist die Anzahl der Kammern der Pumpe anzugeben. Bei Kolbenpumpen ist die Anzahl der Kolben einzustellen. Bei Innenzahnradpumpen ist die Anzahl der Zähne auf dem inneren Ritzel einzustellen.
nConcept	DINT	Hier ist das verwendete Schaltungs-Konzept der servoelektrisch-hydraulischen Achse anzugeben.
nPump_EncType	E_TcMcEncoderType	Hier wird der <u>Encodertyp</u> [► 93] des Pumpenantriebs festgelegt. Es steht nur eine kleine Auswahl von Encodertypen zur Verfügung. <b>Dies ist nicht der Encoder am Zylinder.</b>
bRegenerative	BOOL	Dieser Parameter signalisiert, dass die kleinere Zylinderfläche im Ölaustausch mit der größeren Zylinderfläche betrieben wird.

Name	Typ	Beschreibung
bVirtual_A_addP	BOOL	Wenn eine in positiver Wirkungsrichtung zuschaltbare Fläche für den Ölbedarf wirksam ist, für den Kraftaufbau jedoch nicht, ist sie hier zu kennzeichnen.
bVirtual_A_addM	BOOL	Wenn eine in negativer Wirkungsrichtung zuschaltbare Fläche für den Ölbedarf wirksam ist, für den Kraftaufbau jedoch nicht, ist sie hier zu kennzeichnen.
bAside_PrsHiResADC	BOOL	Dieser Parameter signalisiert, dass der Drucksensor der Fläche mit positiver Wirkungsrichtung mit einer 24-Bit-Eingangsklemme eingelesen wird.
bBside_PrsHiResADC	BOOL	Dieser Parameter signalisiert, dass der Drucksensor der Fläche mit negativer Wirkungsrichtung mit einer 24-Bit-Eingangsklemme eingelesen wird.
bSystem_PrsHiResADC	BOOL	Dieser Parameter signalisiert, dass der Drucksensor am vorgespannten Speicher mit einer 24-Bit-Eingangsklemme eingelesen wird.

Weitere Informationen zur Achsinbetriebnahme finden Sie unter Setup.

### 4.3.34 ST\_TcHybridAxRtData (ab V3.0.44)

Diese Struktur enthält zusätzliche Laufzeitwerte der servoelektrisch-hydraulischen Achse.



Die Reihenfolge der Werte ist nicht garantiert.

#### Syntax

```

TYPE ST_TcHybridAxRtData :
(* last modification: 05.12.2018 *)
STRUCT
  fPump_Angle:          LREAL;
  fPump_ModuloAngle:   LREAL;
  fPump_Speed:         LREAL;
  fPump_Torque:        LREAL;

  fMotor_N_max:        LREAL;
  fMotor_RefCurrent:   LREAL;
  fMotor_RefTorque:    LREAL;
  fMotor_PeekCurrent:  LREAL;
  fMotor_PeekTorque:   LREAL;
  fMotor_NomCurrent:   LREAL;
  fMotor_NomTorque:    LREAL;

  fActive_Area_P:      LREAL;
  fActive_Area_M:      LREAL;
  fActive_Qmax_P:      LREAL;
  fActive_Qmax_M:      LREAL;

  fActive_Feed_P:      LREAL;
  fActive_Feed_M:      LREAL;

  fActive_N_max:       LREAL;

  fActive_Vmax_P:      LREAL;
  fActive_Vmax_M:      LREAL;

  fFeed_RampRate_P:    LREAL;
  fFeed_RampRate_M:    LREAL;
  fRamping_Feed_P:     LREAL;
  fRamping_Feed_M:     LREAL;

  bPump_Switched:      BOOL;

```

```

bPump_AreaSwitched: BOOL;
bMotor_EnablePwrMon:BOOL;
bReRamp_FeedFactor: BOOL;
bHydActualCall:      BOOL;
END_STRUCT
END_TYPE

```

**Parameter**

Name	Typ	Beschreibung
fPump_Angle	LREAL	[°] Der umdrehungsbezogene Istwinkel des Motors und somit auch der Pumpe im Bereich 0 ... 360°.
fPump_ModuloAngle	LREAL	[°] Der kavitätsbezogene Istwinkel des Motors und somit auch der Pumpe im Bereich 0 ... 360°/ Anzahl_der_Kavitäten.
fPump_Speed	LREAL	[°/s] Die Winkelgeschwindigkeit des Motors und somit auch der Pumpe. Dieser Wert entspricht dem 6-fachen der Drehzahl in prm.
fPump_Torque	LREAL	[%] Das von der Pumpe vom Motor abgerufene Drehmoment, bezogen auf dessen Nennmoment.
fMotor_N_max	LREAL	[rpm] Die Maximal-Drehzahl des Motors.
fMotor_RefCurrent	LREAL	[A] Der Strom-Bezugswert des Motors.
fMotor_RefTorque	LREAL	[Nm] Der Drehmoment-Bezugswert des Motors.
fMotor_PeekCurrent	LREAL	[A] Der Strom-Spitzenwert des Motors.
fMotor_PeekTorque	LREAL	[Nm] Der Drehmoment- Spitzenwert des Motors.
fMotor_NomCurrent	LREAL	[A] Der Strom-Nennwert des Motors.
fMotor_NomTorque	LREAL	[Nm] Der Drehmoment- Nennwert des Motors.
fActive_Area_P	LREAL	[mm <sup>2</sup> ] Die momentan wirksame Fläche auf der P-Seite des Zylinders.
fActive_Area_M	LREAL	[mm <sup>2</sup> ] Die momentan wirksame Fläche auf der M-Seite des Zylinders.
fActive_Qmax_P	LREAL	[cm <sup>3</sup> /U] Die momentan verfügbare Menge auf der P-Seite des Zylinders.
fActive_Qmax_M	LREAL	[cm <sup>3</sup> /U] Die momentan verfügbare Menge auf der M-Seite des Zylinders.
fActive_Feed_P	LREAL	[mm/U] Der momentan verfügbare Umdrehungsvorschub auf der P-Seite des Zylinders.
fActive_Feed_M	LREAL	[mm/U] Der momentan verfügbare Umdrehungsvorschub auf der M-Seite des Zylinders.
fActive_N_max	LREAL	[rpm] Die momentan verfügbare Maximal-Drehzahl von Motor und Pumpe.
fActive_Vmax_P	LREAL	[mm/s] Die momentan verfügbare Maximal-Geschwindigkeit in positiver Richtung.
fActive_Vmax_M	LREAL	[mm/s] Die momentan verfügbare Maximal-Geschwindigkeit in negativer Richtung.
fFeed_RampRate_P	LREAL	[mm/U/cycle] Die zyklusbezogene Vorschubfaktor-Änderung einer aktuellen oder bereits abgearbeiteten Rampe in positiver Bewegungsrichtung des Zylinders.
fFeed_RampRate_M	LREAL	[mm/U/cycle] Die zyklusbezogene Vorschubfaktor-Änderung einer aktuellen oder bereits abgearbeiteten Rampe in negativer Bewegungsrichtung des Zylinders.
fRamping_Feed_P	LREAL	[mm/U] Der aktuelle verrampte Vorschubfaktor in positiver Bewegungsrichtung des Zylinders.
fRamping_Feed_M	LREAL	[mm/U] Der aktuelle verrampte Vorschubfaktor in positiver Bewegungsrichtung des Zylinders.
bPump_Switched	BOOL	Dieses Signal zeigt eine aktive Umschaltung der Pumpe in den Kraftgang an.

Name	Typ	Beschreibung
bPump_AreaSwitched	BOOL	Dieses Signal zeigt eine aktive Zuschaltung der Flächen für den Kraftgang an.
bMotor_EnablePwrMon	BOOL	Dieses Signal zeigt an, dass die Strom- und Drehmoment-Parameter des Antriebs gelesen wurden und eine genaue Drehmomentberechnung verfügbar ist.
bReRamp_FeedFactor	BOOL	Dieses Signal startet die Rampe für die Umschaltung zwischen Eil- und Kraftgang an.
bHydActualCall	BOOL	Dieses Signal zeigt an, dass eine Instanz des Bausteins MC_AxRtHybridAxisActuals_BkPlcMc () für die servoelektrisch-hydraulische Achse aufgerufen wurde. Andernfalls ist nicht sichergestellt, dass die Istwerte der Achse vollständig ermittelt und die Auswirkungen einer Pumpen- oder Flächenumschaltung berücksichtigt werden. In diesem Fall wird die Achse in den Fehlerzustand gesetzt und eine Meldung in das Log geschrieben.

### 4.3.35 ST\_TcPlcInputAnalog (ab V3.0.44)

Diese Struktur enthält Variablen für die Auswertung von analogen Eingängen.

#### Syntax

```

TYPE ST_TcPlcInputAnalog :
(* last modification: 20.02.2019 *)
STRUCT
    nADC:      DINT;
    nOpState:  INT;
    bWcState:  BOOL;
END_STRUCT
END_TYPE

```

#### Parameter

Name	Typ	Beschreibung
nADC	DINT	nADC: Hier wird der Istwert dargestellt.  <b>Wenn dieser Wert mit einer 16-Bit-Klemme ermittelt wird, ist er anzupassen. Handelt es sich um einen vorzeichenbehafteten Wert (z.B. von einer ±10 V-Klemme) muss er mit einer Typ-Umwandlung INT_TO_DINT() zugewiesen werden. Dadurch wird das Vorzeichen automatisch typrichtig in den oberen 16 Bit erweitert. Andernfalls werden negative Werte als sehr große positive Werte interpretiert. Treten nur positive Werte auf kann darauf verzichtet werden. In diesem Fall kann auch ein direktes Mapping 16→32 Bit genutzt werden, da die oberen 16 Bit dabei unberührt bleiben.</b>
nOpState	INT	nOpState: Dieses Signal meldet den Betriebszustand der Klemme.
bWcState	BOOL	bWcState: Dieses Signal meldet ein Problem beim kontinuierlichen Datenaustausch mit der Klemme.

### 4.3.36 ST\_TcPctrlParam

Diese Struktur enthält zusätzliche Parameter, die für einen Kraft- oder Druck-Regler genutzt werden können. Die Versorgung eines solchen Bausteins ist durch die Applikation durchzuführen.



Die Reihenfolge der Parameter ist nicht garantiert

**Syntax**

```

TYPE ST_TcPctrlParam :
(* last modification: 30.07.2019 *)
STRUCT
  kP:          LREAL;
  fTn:        LREAL;
  fTv:        LREAL;
  fPreset:    LREAL;
  fWuLimit:   LREAL;

  nNf:        INT;

  bAlignAreas: BOOL;
END_STRUCT
END_TYPE
    
```

**Parameter**

Name	Typ	Beschreibung
kP	LREAL	Die Proportional-Verstärkung des Reglers.
fTn	LREAL	Die Integrations-Zeitkonstante des Reglers. Wird sie auf 0.0 gesetzt ist der I-Anteil abgeschaltet.
fTv	LREAL	Die Vorhalte-Zeitkonstante des Reglers. Wird sie auf 0.0 gesetzt ist der D-Anteil abgeschaltet.
fPreset	LREAL	Mit diesem Wert wird der I-Anteil bei seiner Aktivierung initialisiert.
fWuLimit	LREAL	Die Begrenzung für den I-Anteil.
nNf	INT	Die Reaktion des D-Anteils erzeugt in der Regel ein raues Signal, das eine Achse unruhig werden lässt. Mit diesem Parameter kann ein Gleitender Mittelwertfilter aktiviert werden, der bis zu 100 Werte mittelt.
bAlignAreas	BOOL	Wenn dieser Parameter TRUE ist wird die Ausgabe des Reglers richtungsabhängig an das Verhältnis der Wirkungsflächen eines Zylinders angepasst. Dies kann zu einer stabileren Regelung beitragen, wenn die Achse in beiden Richtungen regeln muss.

**HINWEIS**

**Unerwünschte Schwingung**  
 Eine starke Filterung erzeugt einen Phasenfehler, der zu einer Schwingung führen kann.

**4.3.37 MC\_Ref\_Signal\_Ref\_BkPlcMc**

Eine Variable dieses Typs wird an einen [MC StepAbsoluteSwitch\\_BkPlcMc \[► 290\]](#) bzw. [MC StepAbsoluteSwitchDetection\\_BkPlcMc \[► 292\]](#) Baustein übergeben.

**Syntax**

```

TYPE MC_Ref_Signal_Ref_BkPlcMc:
STRUCT
  SignalSource:    E_SignalSource_BkPlcMc := E_SignalSource_BkPlcMc.SignalSource_Default;
  Level:          BOOL;
END_STRUCT
END_TYPE

TYPE E_SignalSource_BkPlcMc:
  SignalSource_Default := 0;
  (**)
END_TYPE
    
```

**Parameter**

Name	Typ	Beschreibung
SignalSource	E_SignalSource_BkPlcMc	SignalSource: Auswahl der Signalquelle durch E_SignalSource_BkPlcMc.
Level	BOOL	Level:Eingangssignal der Referenznocke.

### 4.3.38 E\_TcMcJogMode

Die Konstanten in dieser Auflistung werden zur Umschaltung zwischen verschiedenen Jog Betriebsarten verwendet.

#### Syntax

```

TYPE E_TcMcJogMode :
(
MC_JOGMODE_STANDARD_SLOW, (* motion with standard jog parameters for slow motion *)
MC_JOGMODE_STANDARD_FAST, (* motion with standard jog parameters for fast motion *)
MC_JOGMODE_CONTINUOUS, (* axis moves as long as the jog button is pressed using parameterized
dynamics *)
MC_JOGMODE_INCHING, (* axis moves for a certain relative distance *)
MC_JOGMODE_INCHING_MODULO (* axis moves for a certain relative distance - stop position is rounded
to the distance value *));
END_TYPE

```

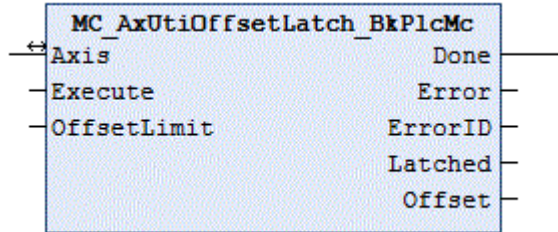
#### Parameter

Name	Beschreibung
MC_JOGMODE_STANDARD_SLOW	Die Achse wird so lange verfahren, wie das Signal an einem der Jog-Eingänge TRUE ist. Dabei wird die in der Axis_Ref_BkPlcMc festgelegte niedrige Geschwindigkeit für Handfunktionen und die Standarddynamik verwendet. In dieser Betriebsart haben die am Funktionsbaustein angelegten Positions-, Geschwindigkeits- und Dynamikdaten keine Bedeutung.
MC_JOGMODE_STANDARD_FAST	Die Achse wird so lange verfahren, wie das Signal an einem der Jog-Eingänge TRUE ist. Dabei wird die in der Axis_Ref_BkPlcMc festgelegte höhere Geschwindigkeit für Handfunktionen und die Standarddynamik verwendet. In dieser Betriebsart haben die am Funktionsbaustein angelegten Positions-, Geschwindigkeits- und Dynamikdaten keine Bedeutung.
MC_JOGMODE_CONTINUOUS	Die Achse wird so lange verfahren, wie das Signal an einem der Jog-Eingänge TRUE ist. Dabei werden die vom Anwender angegebenen Geschwindigkeits- und Dynamikdaten verwendet. Die Position hat keine Bedeutung.
MC_JOGMODE_INCHING	Die Achse wird mit steigender Flanke an einem der Jog-Eingänge um eine bestimmte Distanz verfahren, die über den Positions-Eingang festgelegt wird. Die Achse stoppt automatisch, unabhängig vom Zustand der Jog-Eingänge. Erst mit einer weiteren steigenden Flanke wird ein neuer Bewegungsschritt ausgeführt. Mit jedem Start werden die vom Anwender angegebenen Geschwindigkeits- und Dynamikdaten verwendet.
MC_JOGMODE_INCHING_MODULO	reserviert

## 4.4 System

### 4.4.1 Controller

#### 4.4.1.1 MC\_AxCtrlAutoZero\_BkPlcMc (ab V3.0)



Der Funktionsbaustein führt einen automatischen Nullpunktgleich aus. Dieser Baustein darf nur bei Nullschnitt-Ventilen verwendet werden.

#### Eingänge

```
VAR_INPUT
  Enable:          BOOL:=FALSE;
  EnableOnMoving: BOOL:=FALSE;
  OffsetLimit:    LREAL:=0.0;
  Tn:             LREAL:=0.0;
  Threshold:      LREAL:=0.1;
  Filter:         LREAL:=0.1;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Dieser Eingang kontrolliert die Aktivität der Kompensation.
EnableOnMoving	BOOL	Dieser Eingang kontrolliert die Aktivität der Kompensation.
OffsetLimit	LREAL	[V] Der Wert in fZeroCompensation wird auf diesen Wert begrenzt.
Tn	LREAL	[s] Die Nachstellzeit der Kompensation. Dies ist die Zeit für eine Änderung um 10V. Es werden Werte >100s empfohlen.
Threshold	LREAL	[V] Parameter für das Done Signal.
Filter	LREAL	[s] Parameter für das Done Signal.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [► 84]</a> zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
  Active:         BOOL;
  Limiting:       BOOL;
  Done:          BOOL;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

Name	Typ	Beschreibung
Active	BOOL	Hier wird signalisiert, dass der Baustein den Wert von fZeroCompensation in <a href="#">ST_TcHydAxParam [▶ 120]</a> aktiv verstellt.
Limiting	BOOL	Hier wird signalisiert, dass der Wert von fZeroCompensation in <a href="#">ST_TcHydAxParam [▶ 120]</a> die von OffsetLimit festgelegte Grenze erreicht hat.
Done	BOOL	Hier wird ein Einpendeln des Offsetabgleichs signalisiert.

### Aufgabe des Bausteins

Wenn ein Hydraulik-Zylinder bei abgeschaltetem Lageregler ( $kP=0.0$ ) driftet oder bei aktivem Lageregler mit einem permanentem Schleppabstand neben dem Ziel steht kann dies bei Verwendung eines Nullschnitt-Ventils mit einer Offset-Kompensation korrigiert werden.

Ein Hydraulik-Zylinder steht dann still wenn er sich im Kraftgleichgewicht befindet. Im einfachsten Fall (Zylinder mit gleichen Flächen, keine äußeren Kräfte durch Gravitation oder einen Prozess) ist dieses Gleichgewicht dann erfüllt, wenn auf beiden Flächen gleiche Drücke anstehen. Bei einem Differential-Zylinder müssen sich hierzu die Drücke umgekehrt zu den Flächen verhalten. Bei äußeren Kräften sind diese einzubeziehen. Damit die erforderlichen Druckverhältnisse zustande kommen wird als Druckdifferenz ein Anteil des Systemdrucks benötigt. Dieser wird bei einem Nullschnitt-Ventil über die Druckverstärkungskennlinie definiert.

Eine weitere mögliche Ursache für einen Offset ist eine Differenz zwischen dem hydraulischen Nullpunkt des Ventils und dem logischen Nullpunkt der Ausgabe-Hardware. Hier handelt es sich um unvermeidbare Exemplarstreuungen.

Es wird also eine kleine Ansteuerung des Ventils von bis zu etwa  $\pm 0.5V$  benötigt. Nähere Angaben hierzu sind dem Datenblatt der Ventil- und Hardware-Hersteller zu entnehmen.

### Verhalten des Bausteins: Freigabe-Logik

So lange das **Enable** des Bausteins oder die Reglerfreigabe der Achse FALSE sind wird der Baustein nicht **Active**. Der Vergleichswert für die Beobachtung der Kompensation wird initialisiert und die Zeitmessung für die **Done** Meldung wird zurückgesetzt.

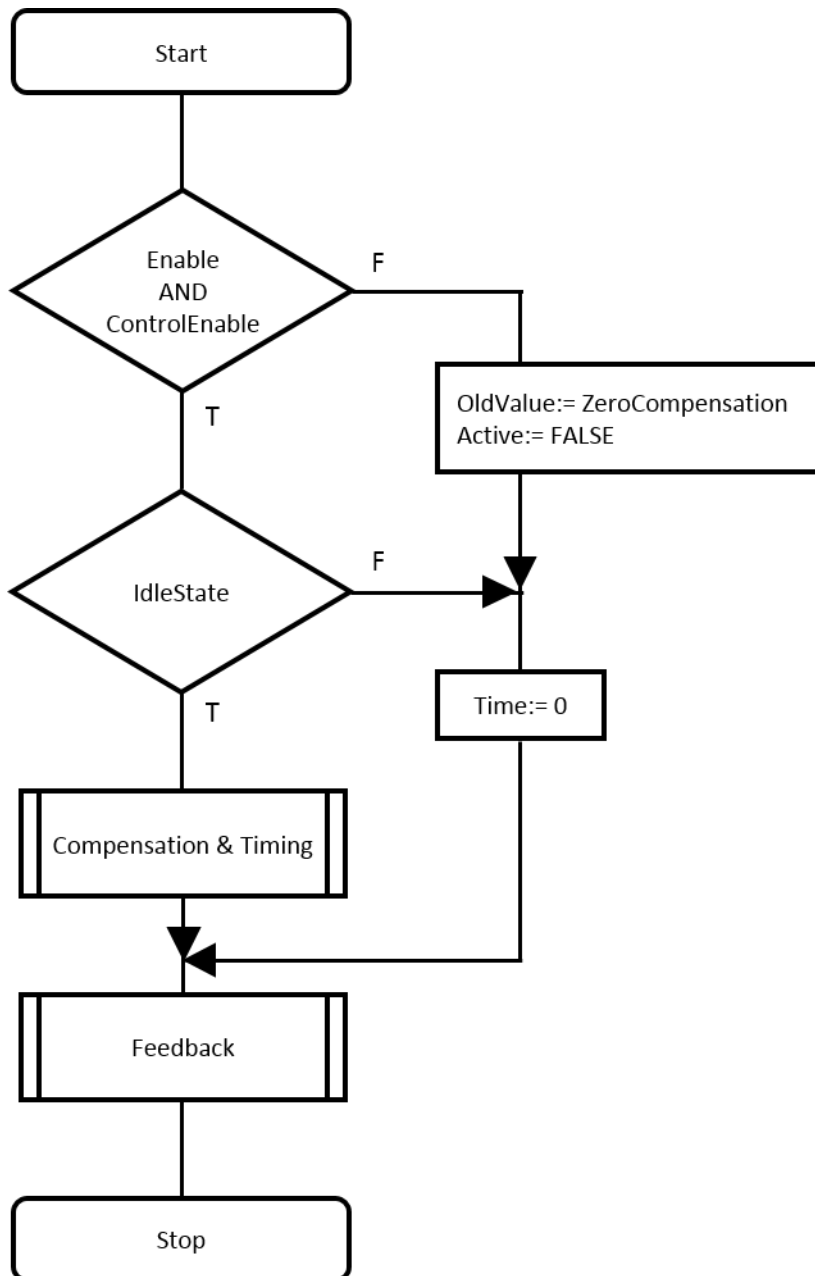
Sind die Freigaben erfüllt und die Achse ist nicht im Idle Zustand (d.h. sie führt eine aktive Bewegung aus) wird nur noch die Zeitmessung für die **Done** Meldung zurückgesetzt.

Sind die Freigaben erfüllt und die Achse ist im Idle-Zustand wird der Block ‚Compensation&Timing‘ durchlaufen.

Unabhängig von diesen Vorbedingungen wird der Block ‚Feedback‘ durchlaufen.

Freigabe Logik:





### Verhalten des Bausteins: Compensation&Timing

Aus dem Schleppabstand und der Reaktion des Reglers wird ein Korrekturwert gebildet. Dabei wird die Bandbreite der möglichen Regler-Parametrierung der Achse berücksichtigt. Aus diesem Korrekturwert und **Tn** wird ein Delta (eine maximale Änderung der **ZeroCompensation** pro Zyklus) gebildet. Dabei legt **Tn** eine Rampenzeit für einen Anstieg um 10V fest. Das Delta wird so begrenzt, dass diese Rampensteigung nicht überschritten wird. So kann eine störend schnelle Änderung vermieden werden, bei der die Korrektur instabil wird. Es sind Werte >100 Sekunden zu empfehlen.

Für die Kompensation wird eine Toleranzschwelle verwendet. Hier wird **LagAmpDx** (Schwellwert des I-Anteils im Lageregler) verwendet.

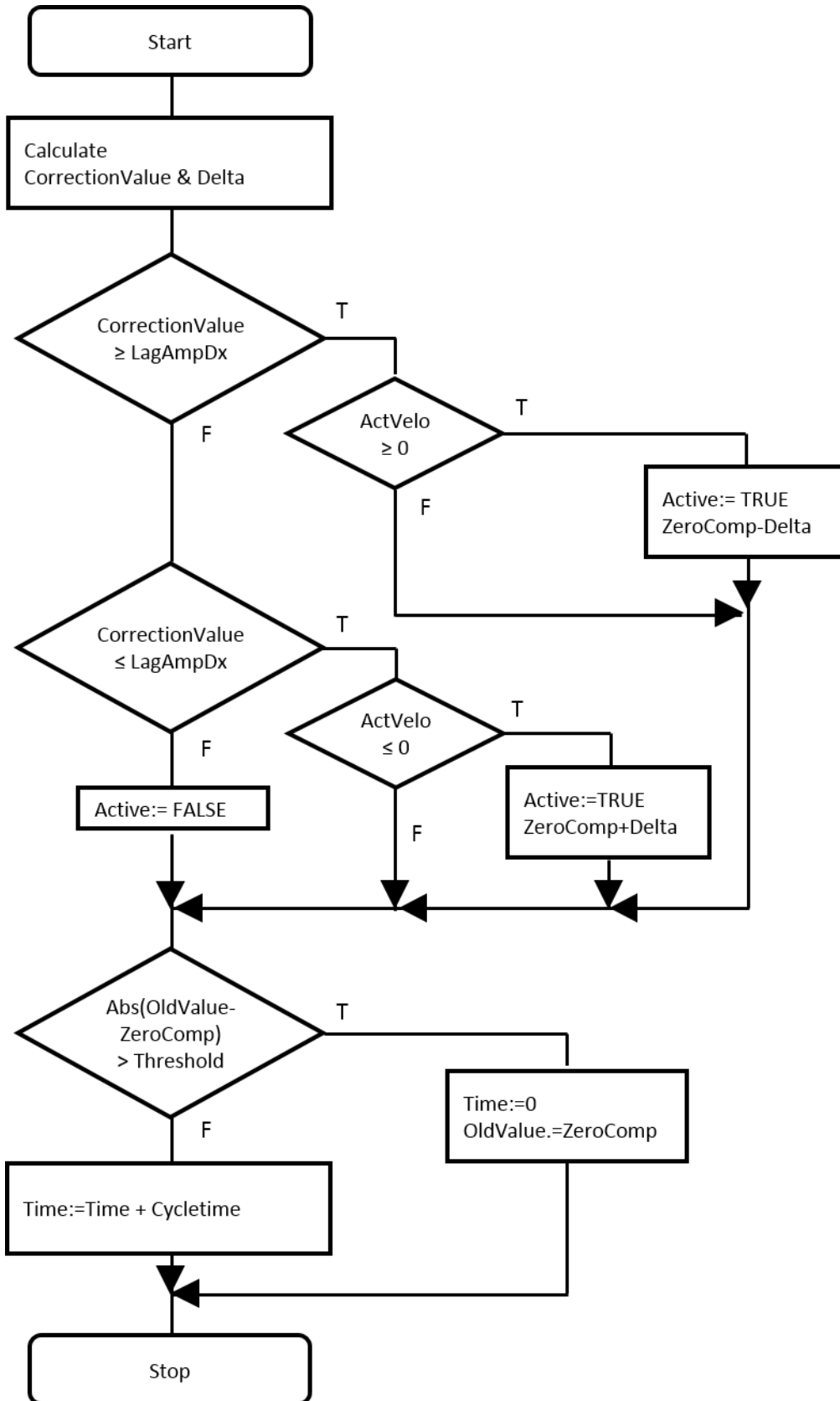
Wenn der Korrekturwert größer/gleich der Toleranzschwelle ist und die Istgeschwindigkeit größer/gleich Null ist (d.h. der verbleibende Korrekturwert wird nicht bereits abgebaut) wird der Baustein **Active** und die Kompensation wird in jedem Zyklus um das beschriebene Delta reduziert.

Wenn der Korrekturwert kleiner/gleich der Toleranzschwelle ist und die Istgeschwindigkeit kleiner/gleich Null ist (d.h. der verbleibende Korrekturwert wird nicht bereits abgebaut) wird der Baustein **Active** und die Kompensation wird in jedem Zyklus um das beschriebene Delta erhöht.

Wenn der Korrekturwert absolut kleiner als die Toleranzschwelle ist wird **Active FALSE**.

Ist die Kompensation um mehr als **Threshold** vom Vergleichswert OldValue verschieden wird die Zeitmessung zurückgesetzt und die aktuelle Kompensation als neuer Vergleichswert aktualisiert. Andernfalls wird die Zeitmessung mit der Zykluszeit erhöht. So wird die Zeit erfasst, die benötigt wird um eine Änderung der Kompensation um mindestens **Threshold** anzusammeln.

Compensation&Timing:

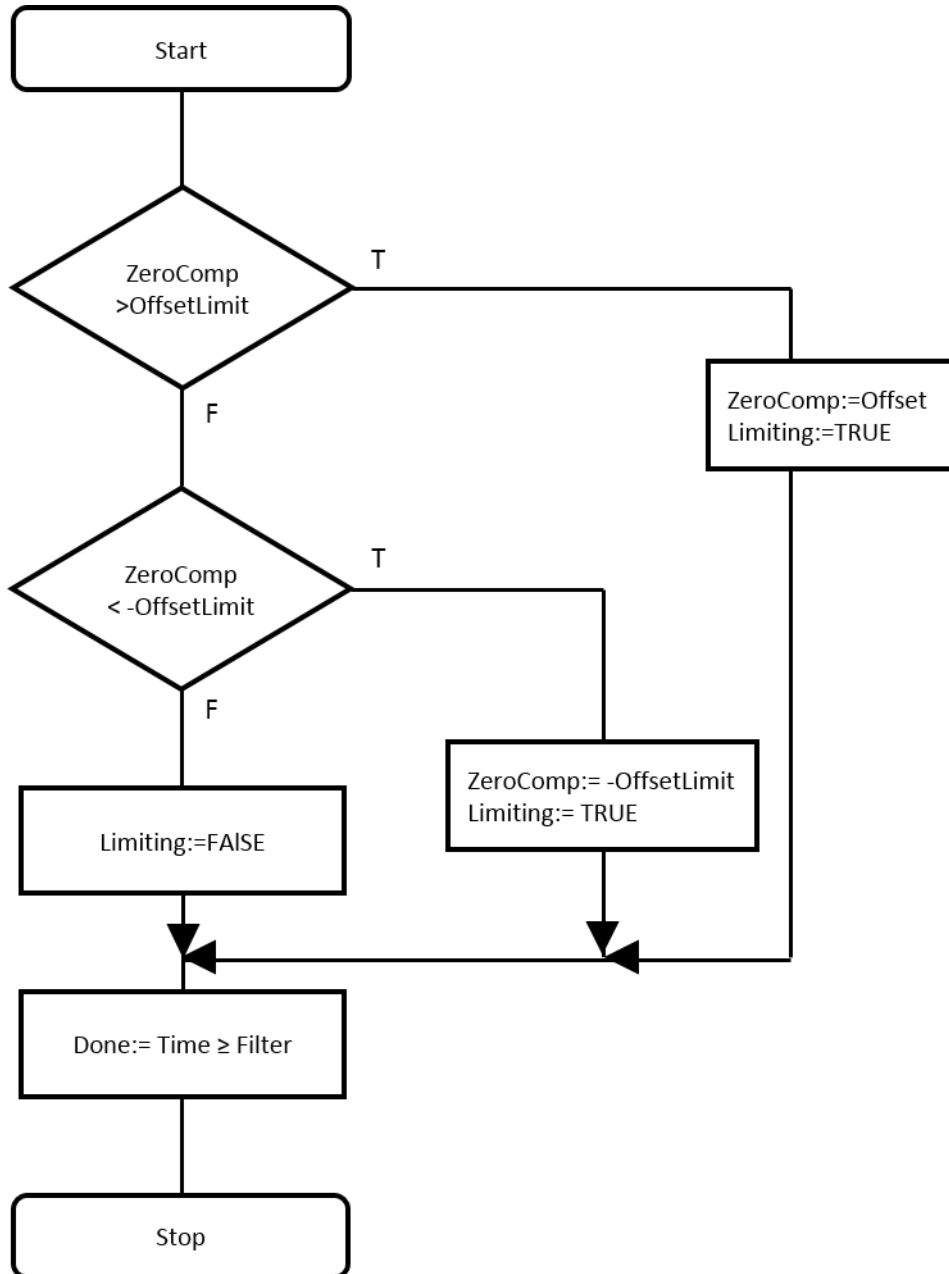


**Verhalten des Bausteins: Feedback**

Es wird eine Begrenzung der Kompensation auf  $\pm$ OffsetLimit vorgenommen und an **Limiting** signalisiert.

Wenn die Zeitmessung bei aktivem Baustein die in **Filter** eingestellte Zeit erreicht wird **Done** gemeldet. Beispiel: Wenn als **Threshold** 0.05 und als **Filter** 2.0 eingestellt sind wird **Done** gemeldet, wenn die Kompensation innerhalb der letzten 2 Sekunden um weniger als 0.05V nachjustiert wurde.

Feedback



Die Begrenzung auf den von OffsetLimit festgelegten Wertebereich ist auch dann aktiv, wenn der Baustein nicht aktiv ist. Der Ausgang Limiting wird aktualisiert.

Der Wert **OffsetLimit** und `ST_TcHydAxParam` [► 120].**fZeroCompensation** werden als Offsetspannung betrachtet. Somit entspricht der Wert 10.0 einer Vollaussteuerung. In der Regel ist für **OffsetLimit** je nach Einsatzfall ein Wert zwischen 0.1 und 1.0 sinnvoll.

## Integration des Bausteins in die Applikation

In der Aufruffreihenfolge der Bausteine einer Achse sollte ein `MC_AxCtrlAutoZero_BkPlcMc` Baustein unmittelbar vor dem `MC_AxRtFinish_BkPlcMc` [► 230] Baustein stehen. Wird an Stelle der Einzelbausteine ein `MC_AxStandardBody_BkPlcMc` [► 237] Baustein aufgerufen sollte der `MC_AxCtrlAutoZero_BkPlcMc` vor diesem Baustein aufgerufen werden.

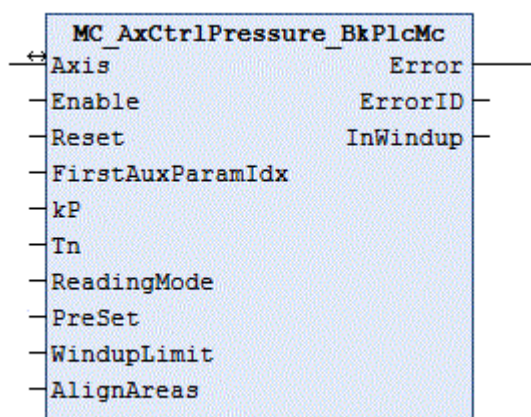
### HINWEIS

#### Gefährliche Achsbewegung

Wenn beim Betrieb der Achse Situationen vorkommen, in denen die Achse eine Reglerfreigabe anstehen hat, aber nicht ihr normales Bewegungsverhalten zeigt ist es zwingend erforderlich, den `MC_AxCtrlAutoZero_BkPlcMc` Bausteins zu disablen. Denkbare Ursachen für eine solche Situation sind zum Beispiel das Anfahren eines Blocks mit oder ohne Übergang in eine Druckregelung oder ein Absenken oder Abschalten der Versorgung. Wird dies nicht beachtet kann der Wert von `fZeroCompensation` in `ST_TcHydAxParam` [► 120] in willkürlicher Richtung bis zur festgelegten Begrenzung laufen. Sobald die Achse zu einem späteren Zeitpunkt wieder reaktionsfähig wird ist eine unter Umständen gefährliche Bewegung unvermeidbar. Das Positionierverhalten wird anschließend stark beeinträchtigt sein. Wird der Baustein ohne **EnableOnMoving** aufgerufen kann er den verstellten Offset dann unter Umständen nicht mehr automatisch korrigieren. Die Achse wird dann außerhalb des Zielfensters stehen bleiben und niemals oder zumindest erst nach einer erheblichen Zeit die Bewegung als abgeschlossen melden.

Bei Kombination mit einem `MC_AxStandardBody_BkPlcMc` [► 237] Baustein werden alle Reaktionen des `MC_AxCtrlAutoZero_BkPlcMc` Bausteins um einen SPS Zyklus verzögert wirksam. In der Regel ist dies kein Problem. Sollte dieser Versatz störend sein müssen die Einzelbausteine für Encoder usw. verwendet und der `MC_AxCtrlAutoZero_BkPlcMc` Baustein unmittelbar vor dem `MC_AxRtFinish_BkPlcMc` [► 230] Baustein aufgerufen werden.

### 4.4.1.2 MC\_AxCtrlPressure\_BkPlcMc (ab V3.0)



Der Funktionsbaustein regelt den an einer Achse wirksamen Druck so, dass in dem durch **ReadingMode** ausgewählten Istwert ein gewünschter Vorgabewert aufgebaut und eingehalten wird.

Die Erfassung des Istdruck kann in den meisten Fällen mit Bausteinen vom Typ `MC_AxRtReadPressureSingle_BkPlcMc` [► 207] oder `MC_AxRtReadPressureDiff_BkPlcMc` [► 204] erfolgen.

#### 🔌 Eingänge

```
VAR_INPUT
  Enable:      BOOL:=FALSE;
  Reset:      BOOL:=TRUE;
  FirstAuxParamIdx: INT:=0;
  kP:        LREAL:=0.0;
  Tn:        LREAL:=0.0;
  ReadingMode: E_TcMcPressureReadingMode:=iTcHydPressureReadingDefault;
  PreSet:    LREAL:=0.0;
  WindupLimit: LREAL:=0.0;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Durch ein TRUE an diesem Eingang wird der Regler aktiviert.
Reset	BOOL	Durch ein TRUE an diesem Eingang wird der Regler zurückgesetzt. Der Speicher des I-Anteils wird abgelöscht.
FirstAuxParamIdx	INT	Hier kann ein Bereich in den <a href="#">Axis_Ref_BkPlcMc</a> [ <a href="#">▶ 84</a> ].ST_TcHydAxParam [ <a href="#">▶ 120</a> ].fCustomerData als Parameter-Interface aktiviert werden.
kP	LREAL	Der Verstärkungsfaktor des P-Anteils.
Tn	LREAL	Die Nachstellzeit des I-Anteils.
ReadingMode	E_TcMcPressureReadingMode	Hier kann die zu regelnde Istgröße festgelegt werden. Als Defaultwert wird <a href="#">Axis_Ref_BkPlcMc</a> [ <a href="#">▶ 84</a> ].ST_TcHydAxRtData [ <a href="#">▶ 131</a> ].fActPressure ausgewählt.
PreSet	LREAL	Hier kann ein Voreinstellwert festgelegt werden, mit dem ein Initialwert für den I-Anteil des Reglers berechnet wird. Auf diesen Wert wird der I-Anteil bei Aktivierung vorgeladen.
WindupLimit	LREAL	Hier kann ein Begrenzungswert für den I-Anteil festgelegt werden. Eine solche Begrenzung verhindert ein extremes Verhalten des I-Anteils in Situationen, in denen die Strecke nicht auf die Ausgaben des Reglers reagiert.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc</a> [ <a href="#">▶ 84</a> ] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
  InWindup:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
InWindup	UDINT	Dieser Ausgang wird TRUE wenn der I-Anteil durch WindupLimit begrenzt wird.

**Verhalten des Bausteins**

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Durch ein TRUE an **Reset** wird der Baustein unabhängig von den anderen Steuersignalen in einen Ruhezustand versetzt. Dann ist sowohl der P- als auch der I-Anteil gelöscht. Durch **Enable** kann festgelegt werden, ob der Baustein den aktiven Zustand annimmt.

Der Eingang **ReadingMode** legt fest, welche Variable in der stAxRtData Struktur die zu regelnde Größe enthält.

- iTcHydPressureReadingDefault, iTcHydPressureReadingActPressure: fActPressure wird geregelt.
- iTcHydPressureReadingActForce: fActForce wird geregelt.

- jeder andere Wert deaktiviert den Regler.



Der Sollwert ist in `fSetPressure` in der `stAxRtData` Struktur der Achse vorzugeben.

Zunächst ermittelt der Baustein, ob er den aktiven Zustand annehmen oder beenden muss. Dazu wird das Signale **Enable** ausgewertet. Bei einer steigenden Flanke wird der I-Anteil mit **PreSet** initialisiert. Wenn der zu `ST_TcHydAxRtData` [► 131].`fSetPressure` passende Ausgabewert bekannt ist kann dies für ein schnelleres Erreichen des ausgeregelten Zustands genutzt werden. Anschließend wird mit **kP** ein P-Anteil und mit **Tn** ein I-Anteil berechnet. Die Summe dieser Regler-Anteile wird als Stellwert in `ST_TcHydAxRtData` [► 131].`fSetSpeed` ausgegeben. Da dieser Regler die Funktion eines Stellwertgenerators übernimmt löscht er `ST_TcHydAxRtData` [► 131].`fLagCtrlOutput`. Der nach dem Reglerbaustein zu platzierende `MC_AxRtFinish_BkPlcMc` [► 230] Baustein berücksichtigt die Reaktion dann automatisch.

Der Übergang in den inaktiven Zustand führt zu einem Löschen der Regleranteile.

### Integration des Bausteins in die Applikation

Ein Baustein dieses Typs muss nach der Istwert- und Istdruck-Erfassung aufgerufen werden. Er übernimmt die volle Kontrolle über die Achse und ersetzt einen eventuell vorhandenen Baustein für die Stellwert-Generierung.

Es steht ein [Programm-Beispiel](#) [► 303] #15 zur Verfügung.



Ist sowohl ein Baustein für die Stellwert-Generierung als auch ein `MC_AxCtrlPressure_BkPlcMc` Baustein vorhanden sind diese Bausteine entweder alternativ aufzurufen oder der `MC_AxCtrlPressure_BkPlcMc` Baustein muss nach dem Stellwert-Baustein stehen, sodass er dessen Ausgaben überschreibt. Beides ist nicht mit jedem Generator-Typ zulässig.



Durch einen Wert größer als 0 in `FirstAuxParamIdx` kann der Baustein veranlasst werden drei aufeinander folgende Werte in den `fCustomerData` der Parameter-Struktur als `Tn`, `kP` und `PreSet` zu verwenden. Ist in `Axis.pStAxAuxLabels` die Adresse eines geeigneten `ARRAY[.] OF STRING()` eingetragen werden die Parameter automatisch mit einer Bezeichnung versehen.

### Inbetriebnahme

Die vier Parameter **kP**, **Tn**, **PreSet** und **WindupLimit** lassen eine Anpassung des Reglers an eine Reihe unterschiedlicher Aufgaben zu.

#### HINWEIS

##### Schwingungen in der Regelung

Während der Inbetriebnahme kann es dazu kommen, dass die Achse mit dem vollen Systemdruck beaufschlagt wird oder dass gedämpfte oder ungedämpfte Schwingungen in einem weiten Frequenzbereich auftreten. Sollte dies für die Achse oder ihre Umgebung gefährlich sein sind Vorkehrungen zu treffen. In jedem Fall sollten Maßnahmen vorgesehen werden, die Regelungen schnell zu deaktivieren.

Zunächst sollten für **Tn** und **PreSet** der Wert 0.0 und für **WindupLimit** der Wert 1.0 eingetragen werden. Der Regler arbeitet jetzt als reiner P-Regler. Nachdem ein Block angefahren und der Regler aktiviert ist (`Enable:=TRUE`, `Reset:=FALSE`, `SetPressure:=Sollwert`) kann jetzt der maximal anwendbare Wert für **kP** festgestellt werden. Dazu ist der Wert schrittweise zu erhöhen, bis sich eine Schwingneigung zeigt. Durch wiederholtes Deaktivieren und Aktivieren sollte überprüft werden, ob der Regler tatsächlich stabil ist. In der Praxis wird der Wert zwischen etwa 0.1 und 0.5 liegen.

Als nächster Parameter ist **Tn** einzustellen. Dazu sollte zunächst ein relativ großer Wert wie 0.5 vorgegeben werden. Jetzt sollte der Istdruck mit großer Trägheit aber recht genau auf den Sollwert eingeregelt werden. Durch schrittweises Verkleinern wird jetzt die maximal mögliche Einstellung ermittelt. Auch dabei ist durch wiederholtes Deaktivieren und Aktivieren zu überprüfen, ob der Regler tatsächlich stabil ist. Zeigt sich eine Neigung zur gedämpften Schwingung beim Aktivieren ist **Tn** bereits zu niedrig eingestellt.

Die Einstellung von **WindupLimit** beeinflusst das Verhalten des Reglers nicht auf direktem Weg. Vielmehr dient dieser Parameter dazu, das Übergangsverhalten zu beeinflussen. Wenn der Regler den Druck unmittelbar aufbauen kann, weil die Achse keinen Weg zurücklegen muss sollte der Wert von **WindupLimit** so gewählt werden, dass der I-Anteil nicht größer als das drei- bis Vierfache des Wertes erreichen kann, der entsprechend der Ventilcharakteristik benötigt wird. Auf diese Weise kann das Einregeln des Drucks deutlich schneller erreicht werden. Hat die Achse noch einen Weg zurückzulegen wird ein zu niedriger Wert dieses Parameters die Bewegung der Achse zum Erreichen der Arbeitsposition bestimmen. Ist der Parameter zu niedrig gewählt wird die Achse sehr langsam oder kann sogar stehen bleiben. Andererseits wird ein zu hoher Wert dazu führen, dass die Achse die Arbeitsposition mit eher hoher Geschwindigkeit erreicht und der Druckanstieg steil ist. Der dabei auftretende Spitzendruck kann dann erhebliche Werte annehmen.

**HINWEIS**

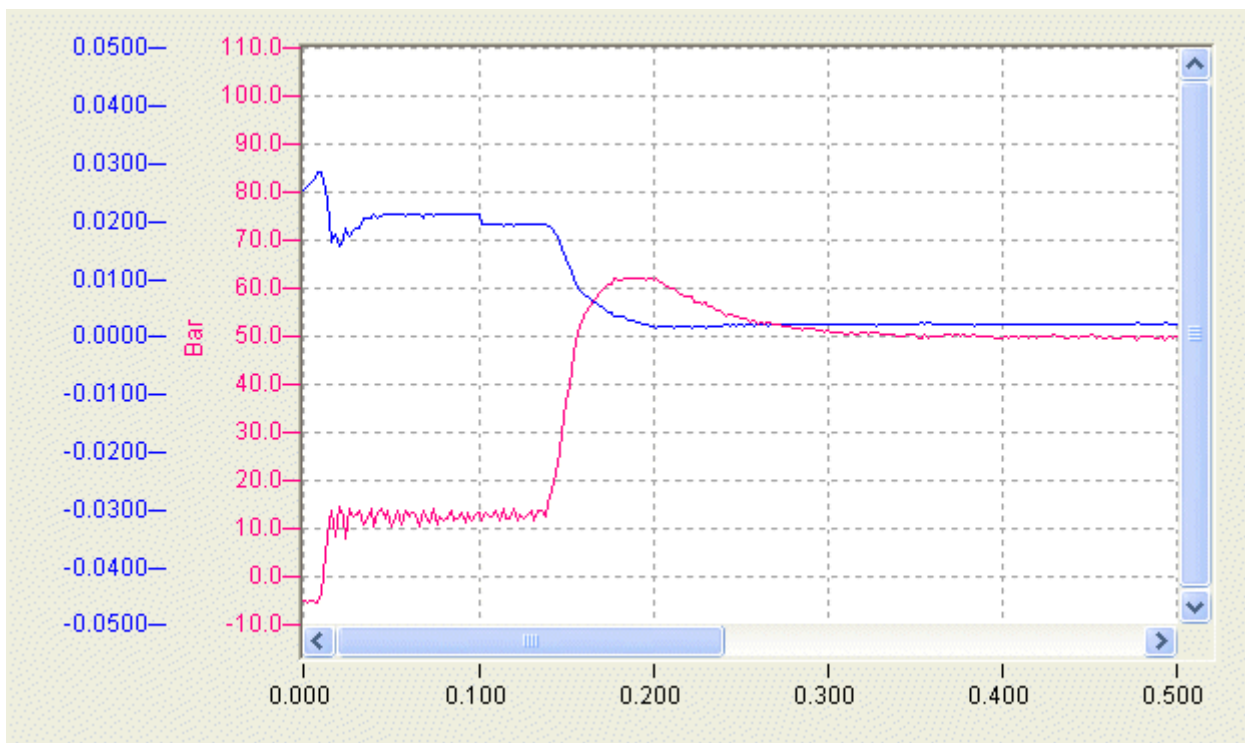
Es ist wenn irgend möglich zu vermeiden, einen Druckregler zu aktivieren, wenn sich die Achse nicht zumindest sehr nah an ihrer Arbeitsposition befindet.

Der Wert für **PreSet** kann für zwei Vorgehensweisen eingesetzt werden. Soll der Druckregler den Stellwert eines anderen Bausteins kontinuierlich fortsetzen kann dessen Stellwert für die Berechnung von **PreSet** vorgegeben werden. So können die Stellwertsprünge beim Aktivieren des Reglers verringert oder völlig vermieden werden.

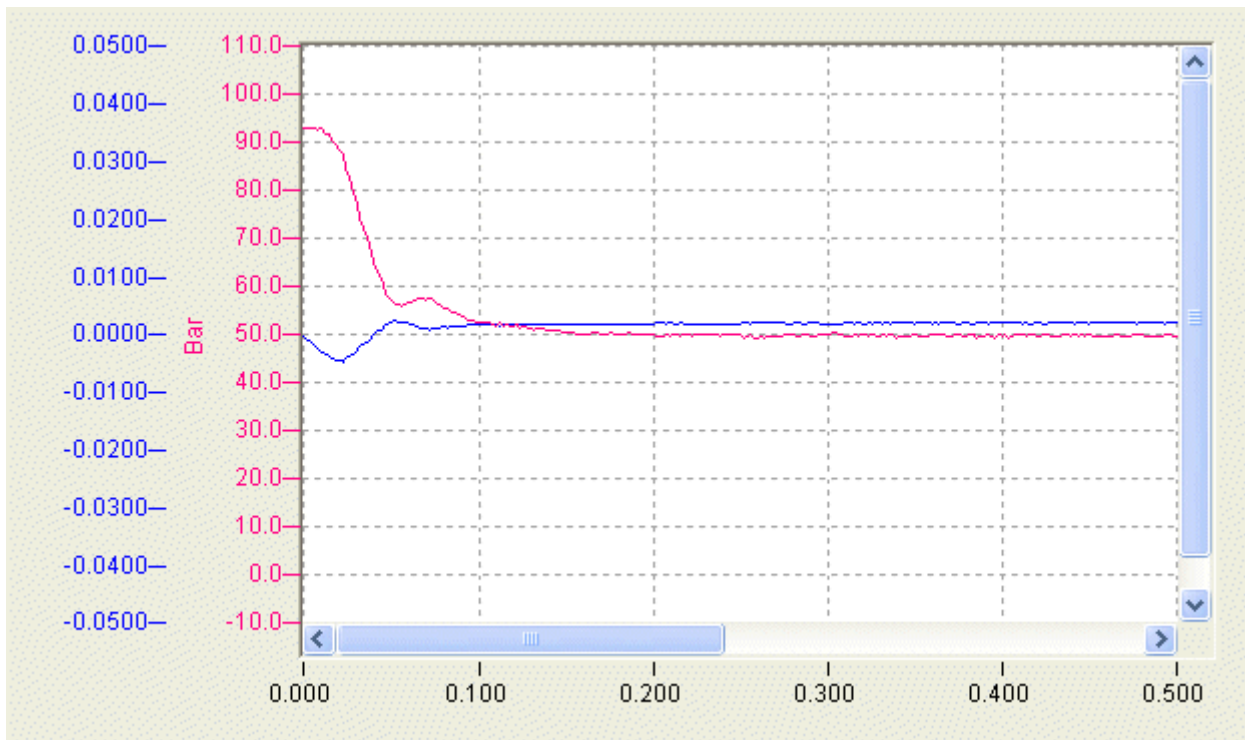
Ist der vom Regler zu erzeugende Stellwert bekannt kann als **PreSet** ein Wert vorgegeben werden, der in der Nähe dieses Wertes liegt. So kann die Zeit verringert werden, die der I-Anteil zum Aufbauen des Stellwerts benötigt. Da auch der P-Anteil wirksam ist sollte jedoch ein Wert eingestellt werden, der höher als der exakte Wert ist.



Letztlich ist bei der Einstellung dieser Parameter durch kleine Veränderungen und Beurteilung des Reglerverhaltens ein der Aufgabe angemessener Satz von Werten zu finden.

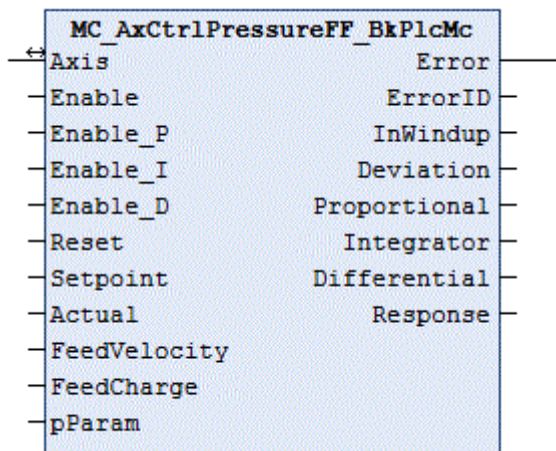


Beispiel für das Verhalten des Reglers, wenn die Achse zunächst einen Weg zurücklegen muss, bevor sie den geforderten Druck aufzubauen vermag.



Beispiel für das Verhalten des Reglers, wenn die Achse unmittelbar den geforderten Druck aufzubauen vermag.

### 4.4.1.3 MC\_AxCtrlPressureFF\_BkPlcMc



Der Funktionsbaustein regelt den an einer Achse wirksamen Druck so, dass in **Actual** der gewünschter Vorgabewert **SetPoint** aufgebaut und eingehalten wird. Alternativ können als Ist- und Sollwerte auch Kräfte verwendet werden.

Die Erfassung eines Istdrucks kann in den meisten Fällen mit Bausteinen vom Typ [MC\\_AxRtReadPressureSingle\\_BkPlcMc \[► 207\]](#) oder [MC\\_AxRtReadPressureDiff\\_BkPlcMc \[► 204\]](#) erfolgen. Für eine Istkraft sind Bausteine vom Typ [MC\\_AxRtReadForceSingle\\_BkPlcMc \[► 202\]](#) oder [MC\\_AxRtReadForceDiff\\_BkPlcMc \[► 199\]](#) geeignet.

#### Eingänge

```
VAR_INPUT
  Enable:          BOOL:=FALSE;
  Enable_P:       BOOL:=TRUE;
  Enable_I:       BOOL:=TRUE;
  Enable_D:       BOOL:=TRUE;
```



```

Reset:          BOOL:=TRUE;
Setpoint:       LREAL:=0.0;
Actual:         LREAL:=0.0;
FeedVelocity:  LREAL:=0.0;
FeedCharge:     LREAL:=0.0;
pParam:        POINTER TO ST_TcPctrlParam;
END_VAR
    
```

Name	Typ	Beschreibung
Enable	BOOL	Durch ein TRUE an diesem Eingang wird der Regler aktiviert.
Enable_P	BOOL	Durch ein TRUE an diesem Eingang wird der Proportional-Anteil des Reglers aktiviert.
Enable_I	BOOL	Durch ein TRUE an diesem Eingang wird der Integrator des Reglers aktiviert, wenn der Proportional-Anteil aktiv ist.
Enable_D	BOOL	Durch ein TRUE an diesem Eingang wird der Differential-Anteil des Reglers aktiviert, wenn der Proportional-Anteil aktiv ist.
Reset	BOOL	Durch ein TRUE an diesem Eingang wird der Regler zurückgesetzt. Der Speicher des I-Anteils wird abgelöscht.
Setpoint	LREAL	Der Sollwert des Reglers.
Actual	LREAL	Der Istwert des Reglers.
FeedVelocity	LREAL	Der Vorgabewert für eine unterlagerte Vorsteuerung.
FeedCharge	LREAL	Eine instantan wirksam werdende und bleibende Veränderung des Integral-Anteils.
pParam	POINTER TO ST_TcPctrlParam	Die Adresse einer Struktur mit den Regler-Parametern. Wird hier Null übergeben verwendet der Regler die Parameter in stAxParams.stPctrl.

 **Ein-/Ausgänge**

```

VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
    
```

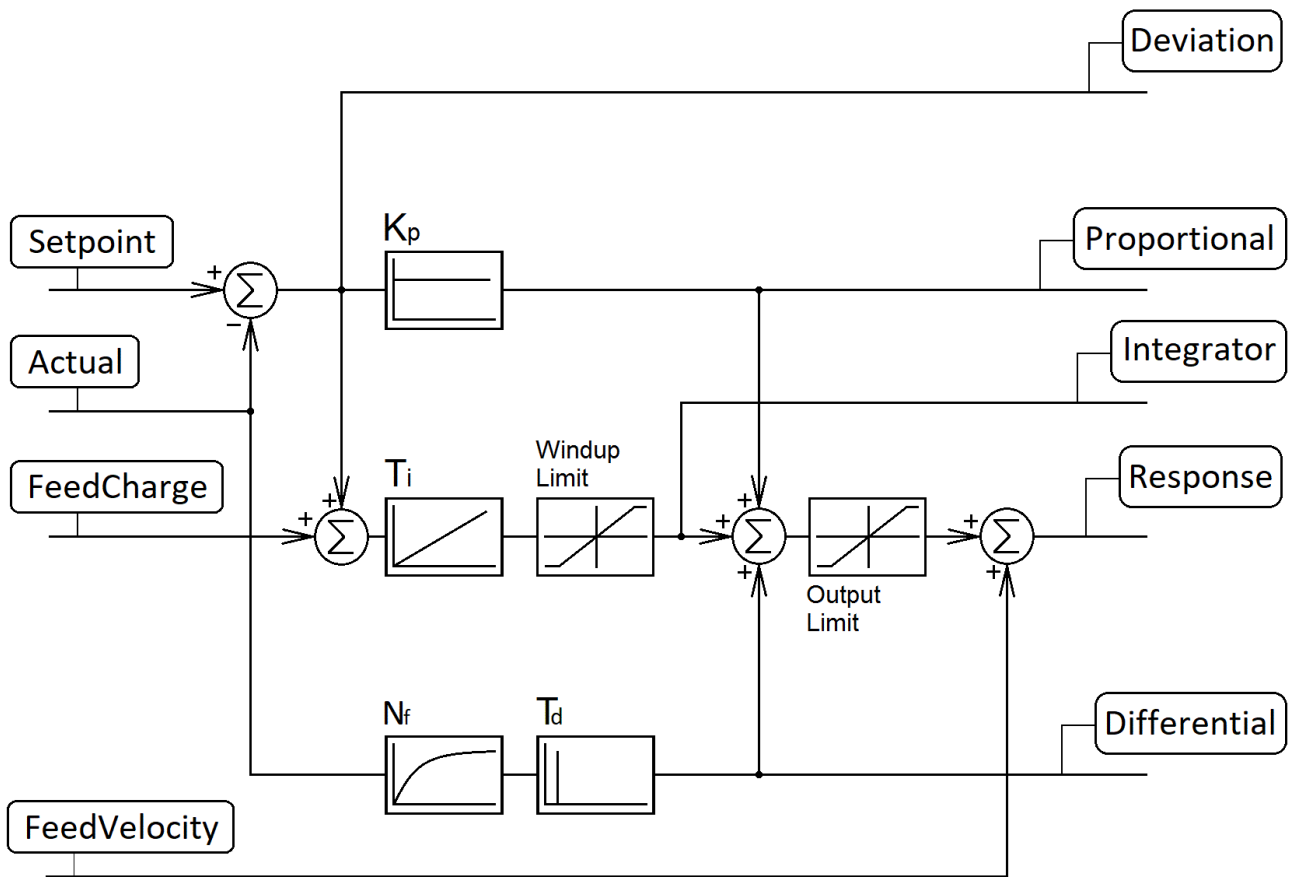
Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

 **Ausgänge**

```

VAR_OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
  InWindup:       UDINT;
  Deviation:      LREAL;
  Proportional:   LREAL;
  Integrator:     LREAL;
  Differential:    LREAL;
  Response:       LREAL;
END_VAR
    
```

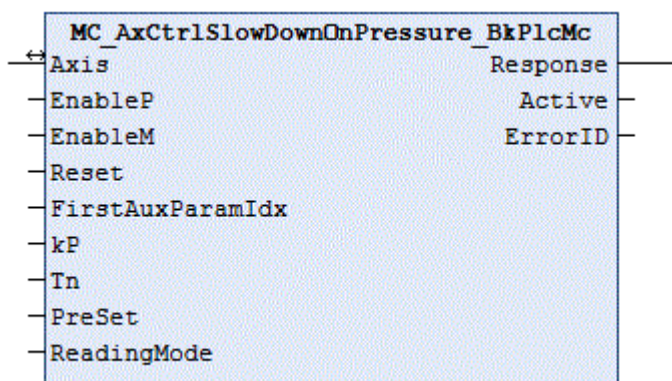
Name	Typ	Beschreibung
Error	BOOL	Ein TRUE hier signalisiert einen Fehler.
ErrorID	UDINT	Eine numerisch kodierte Fehlerangabe.
InWindup	UDINT	Ein TRUE hier signalisiert, das der Integrator durch den WindupLimit begrenzt wird.
Deviation	LREAL	Die aktuelle Reglerabweichung.
Proportional	LREAL	Der aktuelle Proportional-Anteil.
Integrator	LREAL	Der aktuelle Integral-Anteil.
Differential	LREAL	Der aktuelle Differential-Anteil.
Response	LREAL	Der Ausgang des Reglers.



Der Regler verfügt über einen vollständigen PID-Kern, dessen einzelne Anteile über boolsche Eingänge unabhängig von ihren Parametern ein- und ausgeschaltet werden können.

Als Ergänzung steht ein Vorsteuer-Eingang zur Verfügung, der es vereinfacht, die Regelung gegen ein sich bewegendes Objekt an dessen Geschwindigkeit anzupassen. Falls nötig kann der Anteil des Integrierers instantan mit dem Eingang FeedCharge verändert werden.

#### 4.4.1.4 MC\_AxCtrlSlowDownOnPressure\_BkPlcMc (ab V3.0)



Der Funktionsbaustein bremst eine Achse so ab, dass in dem durch **ReadingMode** ausgewählten Istwert ein gewünschter Vorgabewert nicht überschritten wird. Dabei gelten die Regeln der ablösenden Druckregelung.

Die Erfassung des Istdruck kann in den meisten Fällen mit Bausteinen vom Typ [MC\\_AxRtReadPressureSingle\\_BkPlcMc \[ 207 \]](#) oder [MC\\_AxRtReadPressureDiff\\_BkPlcMc \[ 204 \]](#) erfolgen.

 **Eingänge**

```
VAR_INPUT
  EnableP:          BOOL:=FALSE;
  EnableM:          BOOL:=FALSE;
  Reset:            BOOL:=TRUE;
  FirstAuxParamIdx: INT:=0.0;
  kP:               LREAL:=0.0;
  Tn:               LREAL:=0.0;
  PreSet:           LREAL:=0.0;
  ReadingMode:      E_TcMcPressureReadingMode:=iTcHydPressureReadingDefault;
END_VAR
```

E\_TcMcPressureReadingMode [▶ 111]

Name	Typ	Beschreibung
EnableP	BOOL	Durch ein TRUE an diesem Eingang wird dem Regler erlaubt, den Ausgabewert bei einer Bewegung in positiver Richtung zu beeinflussen.
EnableM	BOOL	Durch ein TRUE an diesem Eingang wird dem Regler erlaubt, den Ausgabewert bei einer Bewegung in negativer Richtung zu beeinflussen.
Reset	BOOL	Durch ein TRUE an diesem Eingang wird der Regler zurückgesetzt. Der Speicher des I-Anteils wird abgelöscht.
FirstAuxParamIdx	INT	Hier kann ein Bereich in den <a href="#">Axis_Ref_BkPlcMc [▶ 84].ST_TcHydAxParam [▶ 120].fCustomerData</a> als Parameter-Interface aktiviert werden.
kP	LREAL	Der Verstärkungsfaktor des P-Anteils.
Tn	LREAL	Die Nachstellzeit des I-Anteils.
PreSet	LREAL	Hier kann ein Voreinstellwert festgelegt werden, mit dem ein Initialwert für den I-Anteil des Reglers berechnet wird. Auf diesen Wert wird der I-Anteil bei Aktivierung vorgeladen.
ReadingMode	E_TcMcPressureReadingMode	Hier kann die zu regelnde Istgröße festgelegt werden. Als Defaultwert wird <a href="#">Axis_Ref_BkPlcMc [▶ 84].ST_TcHydAxRtData [▶ 131].fActPressure</a> ausgewählt.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [▶ 84]</a> zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Response:      LREAL;
  Active:        BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Response	LREAL	Der Ausgabewert des Druckreglers.
Active	BOOL	Ein TRUE an diesem Ausgang signalisiert, dass der Baustein einen Response erzeugt, um die Druckregelung zu übernehmen.

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

### Verhalten des Bausteins

Durch ein TRUE an **Reset** wird der Baustein unabhängig von den anderen Steuersignalen in einen Ruhezustand versetzt. Dann ist **Active** FALSE und **Response** := 0.0, weil sowohl der P- als auch der I-Anteil gelöscht sind.

Der Eingang **ReadingMode** legt fest, welche Variable in der `stAxRtData` Struktur die zu regelnde Größe enthält.

- `iTcHydPressureReadingDefault`, `iTcHydPressureReadingActPressure`: `fActPressure` wird geregelt.
- `iTcHydPressureReadingActForce`: `fActForce` wird geregelt.
- jeder andere Wert deaktiviert den Regler.



Der Sollwert ist in `fSetPressure` in der `stAxRtData` Struktur der Achse vorzugeben.

Das Verhalten des Bausteins wird im aktiven Betrieb durch die Eingänge **EnableP** und **EnableM** festgelegt. Sie legen fest, ob der Baustein bei einer Bewegung in positiver oder negativer Richtung eingreifen soll. Es ist dabei zu beachten, dass der Baustein die Aufgabe hat, einer aktiven Fahrbewegung entgegen zu wirken. Wenn also eine Fahrbewegung in positiver Richtung ausgeführt wird, die einen festgelegten Druck nicht überschreiten soll ist **EnableP** zu setzen. Bei entgegengesetzter Fahrtrichtung gibt **EnableM** eine druckbegrenzende Reglerreaktion in positiver Richtung frei.

Zunächst ermittelt der Baustein, ob er den aktiven Zustand annehmen oder beenden muss. Dazu werden die Signale **EnableP**, **EnableM**, das Vorzeichen von `ST_TcHydAxRtData` [▶ 131].`fSetSpeed` und die Differenz zwischen **SetPressure** und dem ausgewählten Istwert ausgewertet.

Beim Übergang in den aktiven Zustand wird der I-Anteil mit **PreSet** initialisiert. Dabei wird er mit einem Startwert geladen, der in Kombination mit `ST_TcHydAxRtData` [▶ 131].`fSetSpeed` den Wert von **PreSet** ergibt. Wenn der zu **SetPressure** passende Ausgabewert bekannt ist kann dies für ein schnelleres Erreichen des ausgeregelten Zustands genutzt werden. In der Praxis sollte die Wahl dieses Parameters vom Verhalten der Regelstrecke abhängig gemacht werden. Diese ist vor allem durch die Nachgiebigkeit des eingepressten Objekts, aber auch durch die gewählte Geschwindigkeit beeinflusst. Ist der Anstieg im Vergleich zur verwendeten **Tn** eher langsam sollte als Vorgabe der aktuelle Stellwert aus `ST_TcHydAxRtData` [▶ 131].`fSetSpeed` verwendet werden. Reagiert der Istdruck mit einem schnellen Anstieg ist ein Wert zu empfehlen, der den Solldruck und die Druckverstärkung des Ventils berücksichtigt.

Anschließend wird mit **kP** ein P-Anteil und mit **Tn** ein I-Anteil berechnet. Die Summe dieser Regler-Anteile wird als **Response** ausgegeben und der Zustand des Reglers mit TRUE an **Active** signalisiert.

Der Übergang in den inaktiven Zustand führt zu einem Löschen der Regleranteile und wird mit FALSE an **Active** markiert.

### Integration des Bausteins in die Applikation

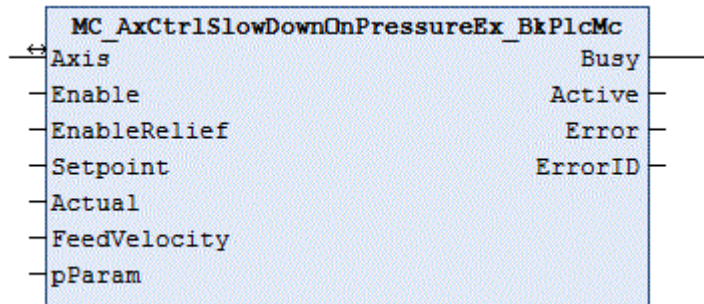
Ein Baustein dieses Typs muss nach der Istwert- und Istdruck-Erfassung und nach der Stellwert-Generierung aufgerufen werden. Werden Bausteine zur Geschwindigkeit- oder Positionsregelung aufgerufen sind diese ebenfalls vor dem Druckregler-Baustein zu platzieren oder die Reaktionen der Regler sind mit entsprechender Sorgfalt zu koordinieren.

Der Druckregler berechnet zwar eine Reaktion, trägt sie aber nicht in die `ST_TcHydAxRtData` [▶ 131] Struktur ein. Dies ist von der Applikation in Abhängigkeit von **Active** und unter eventueller Berücksichtigung von Signalen anderer Regler zu tun. In der Regel wird dabei **Response** auf die Variable `ST_TcHydAxRtData` [▶ 131].`fLagCtrlOutput` zugewiesen. Der nach dem Reglerbaustein zu platzierende `MC_AxRtFinish_BkPlcMc` [▶ 230] Baustein berücksichtigt die Reaktion dann automatisch.



Durch einen Wert größer als 0 in FirstAuxParamIdx kann der Baustein veranlasst werden drei aufeinander folgende Werte in den fCustomerData der Parameter-Struktur als Tn, kP und PreSet zu verwenden. Ist in Axis.pStAxAuxLabels die Adresse eines geeigneten ARRAY[..] OF STRING() eingetragen werden die Parameter automatisch mit einer Bezeichnung versehen.

### 4.4.1.5 MC\_AxCtrlSlowDownOnPressureEx\_BkPlcMc



Der Funktionsbaustein bremst eine Achse so ab, dass der Istwert in **Actual** den in **Setpoint** vorgegebenen Sollwert nicht überschreitet.

Die Erfassung des Istdrucks bzw. der Istkraft kann in den meisten Fällen mit Bausteinen vom Typ [MC\\_AxRtReadPressureSingle\\_BkPlcMc](#) [▶ 207] oder [MC\\_AxRtReadPressureDiff\\_BkPlcMc](#) [▶ 204] bzw. [MC\\_AxRtReadForceSingle\\_BkPlcMc](#) [▶ 202] oder [MC\\_AxRtReadForceDiff\\_BkPlcMc](#) [▶ 199] erfolgen.

#### Eingänge

```
VAR_INPUT
  Enable:          BOOL:=FALSE;
  EnableRelief:   BOOL:=FALSE;
  Setpoint:       LREAL;
  Actual:         LREAL;
  FeedVelocity:   LREAL:=0.0;
  pParam:         POINTER TO ST_TcPctrlParam:=0;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Durch ein TRUE an diesem Eingang wird der Regler freigegeben.
EnableRelief	BOOL	Durch ein TRUE an diesem Eingang wird dem Regler erlaubt, bei Bedarf aktiv zurückzuweichen.
Setpoint	LREAL	Der Sollwert für den zu begrenzenden Istwert.
Actual	LREAL	Der aktuelle Wert der zu begrenzenden Größe.
FeedVelocity	LREAL	Wenn sich das Objekt, gegen das der Druck bzw. die Kraft ausgeübt wird, bewegt, kann dessen Geschwindigkeit hier vorgesteuert werden.
pParam	POINTER TO ST_TcPctrlParam	Hier kann die Adresse einer Struktur vom Typ <a href="#">ST_TcPctrlParam</a> [▶ 148] übergeben werden. Wird dieser Eingang nicht oder mit 0 belegt werden die Regel-Parameter aus den Parametern der Achse verwendet.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc</a> [▶ 84] zu übergeben.

## Ausgänge

```
VAR_OUTPUT
  Busy:          BOOL;
  Active:        BOOL;
  Error:         BOOL;
  ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Ein TRUE an diesem Ausgang signalisiert, dass der Baustein freigegeben ist. Dies bedeutet nicht unbedingt, dass er aktiv in das Verhalten der Achse eingreift.
Active	BOOL	Ein TRUE an diesem Ausgang signalisiert, dass der Baustein freigegeben ist und aktiv in das Verhalten der Achse eingreift.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

### Verhalten des Bausteins

Durch ein FALSE an **Enable** oder bei fehlender Reglerfreigabe der Achse wird der Baustein in einen Ruhezustand versetzt. Dann sind **Busy**, **Active** und **Error** := FALSE.

Bei **Enable** = TRUE wird eine Reihe von Voraussetzungen überprüft.

- **Setpoint** muss sich erkennbar von 0.0 unterscheiden.
- Die Achse muss weggesteuert betrieben werden.
- Die Achse darf nicht mit externer Sollwertgenerierung betrieben werden.
- Das Produkt aus **Setpoint** und kP des Reglers darf nicht unter die Schleichgeschwindigkeit der Achse fallen. In diesem Fall wäre die Achse nicht mehr in der Lage, sich sinnvoll zu bewegen.

Ist eine dieser Voraussetzungen nicht erfüllt sind **Busy** und **Active** = FALSE und **Error** = TRUE. Andernfalls ist **Busy** = TRUE.

Das Vorzeichen von **Setpoint** entscheidet über die Arbeitsrichtung des Bausteins. Ist er positiv greift er ein, wenn sich **Actual** dem Sollwert von unten in steigender Richtung nähert. Nur wenn der Baustein eingreift ist **Active** = TRUE.

Der Baustein ermittelt unter Berücksichtigung der Arbeitsrichtung den Abstand zwischen Sollwert und Istwert und errechnet mit dem kP aus der verwendeten Parameterstruktur die noch zulässige Geschwindigkeit. Übersteigt die Sollgeschwindigkeit diesen Wert wird **Active** = TRUE und die Geschwindigkeit wird limitiert.

Wenn der Istwert den Sollwert erreicht wird die noch zulässige Geschwindigkeit = 0.0 und die Achse sollte spätestens jetzt zum Stehen kommen. Steigt der Istwert weiter wird nur bei **EnableRelief** eine entgegengesetzte Bewegung ausgelöst.

In einigen Anwendungen muss eine Kraft oder ein Druck gegen ein sich bewegendes Objekt ausgeübt werden. In diesem Fall kann die Genauigkeit der Regelung verbessert werden, indem der Regler an **FeedVelocity** eine geeignete Vorsteuerung erhält.

### Integration des Bausteins in die Applikation

Ein Baustein dieses Typs muss nach der Istwert- und Istdruck-Erfassung und nach der Stellwert-Generierung aufgerufen werden. Werden Bausteine zur Geschwindigkeit- oder Positionsregelung aufgerufen sind diese ebenfalls vor dem Druckregler-Baustein zu platzieren. Der nach dem Reglerbaustein zu platzierende [MC\\_AxRtFinish\\_BkPlcMc \[► 230\]](#) Baustein berücksichtigt die Reaktion dann automatisch.

4.4.1.6 MC\_AxCtrlStepperDeStall\_BkPlcMc



Der Funktionsbaustein überwacht die Bewegung einer Schrittmotorachse, die mit einem Encoder betrieben wird.

**i** Der Einsatz eines echten Encoders (nicht Encoder-Emulation durch Impulzzählung einer Ausgabeklemme) ist für die Funktion dieses Bausteins unverzichtbar.

**i** Auch beim Einsatz eines solchen Bausteins kann es zum Stall (Drehmomentabriss) kommen. Es kann also auch dann nicht vorausgesetzt werden, dass die Geschwindigkeit konstant ist.

**Eingänge**

```
VAR_INPUT
  EnableAcc:      BOOL:=FALSE;
  EnableDec:      BOOL:=FALSE;
  Reset:          BOOL:=FALSE;
  UseKL2531State: BOOL:=FALSE;
  ResetRefOnError: BOOL:=FALSE;
  FirstAuxParamIdx: INT:=0;
  VeloLimit:      LREAL:=0.0;
  LimitFilter:    LREAL:=0.0;
  UpdateFilter:   LREAL:=0.0;
END_VAR
```

Name	Typ	Beschreibung
EnableAcc	BOOL	Diese Eingänge legen fest, ob die Überwachung auch in den Beschleunigungs- und Bremsphasen eingreifen darf.
EnableDec	BOOL	
Reset	BOOL	Dieser Eingang kontrolliert die Aktivität des Reglers.
UseKL2531State	BOOL	Wenn hier ein TRUE übergeben wird wertet der Baustein ST_TcPlcDeviceInput [▶ 138].bTerminalState aus.
ResetRefOnError	BOOL	Wenn hier ein TRUE übergeben wird löscht der Baustein das Referenzflag der Achse.
FirstAuxParamIdx	INT	Hier kann ein Bereich in den Axis_Ref BkPlcMc [▶ 84].ST_TcHydAxParam [▶ 120].fCustomerData als Parameter-Interface aktiviert werden.
VeloLimit	LREAL	Die Schwelle für die Geschwindigkeitsabweichung, ab der die Stall-Situation erkannt wird.
LimitFilter	LREAL	Die Zeit, für die eine zu große Geschwindigkeitsabweichung ununterbrochen vorliegen muss, damit die Stall-Situation erkannt wird.
UpdateFilter	LREAL	Die Zeitkonstante, mit der im Baustein der Geschwindigkeits-Stellwert an die Istgeschwindigkeit angepasst wird.

## Ein-/Ausgänge

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

## Ausgänge

```
VAR_OUTPUT
  Active:         BOOL;
  Activated:     BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Active	BOOL	Hier wird signalisiert, dass eine Stall-Situation erkannt ist.
Activated	BOOL	Hier wird signalisiert, dass seit dem letzten Start einer aktiven Achsbewegung eine Stall-Situation erkannt wurde.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

## Verhalten des Bausteins

Bei jedem Aufruf überprüft der Baustein, ob er den Zustand wechseln muss. Er geht in den aktiven Zustand, wenn die interne Bewegungsphase dies nach den Regeln von **EnableAcc**, **EnableDec** zulässt und der Geschwindigkeitsfehler für mindestens **LimitFilter** ununterbrochen den Wert von **VeloLimit** überschreitet. Dabei erlaubt **EnableAcc** dem Baustein, während der Phasen mit konstanter oder im Betrag steigenden Phasen einzugreifen. **EnableDec** gibt die Aktivität des Bausteins für Phasen mit im Betrag sinkender oder konstanter Geschwindigkeit frei. Beim Wechsel in den aktiven Zustand werden **Active** und **Activated** gesetzt.

In den inaktiven Zustand wechselt der Baustein, wenn der Geschwindigkeitsfehler auf den halben Wert von **VeloLimit** abgebaut wurde. Beim Wechsel in den inaktiven Zustand wird **Active** gelöscht.

Im aktiven Zustand wird der Stellwert mit der Zeitkonstanten **UpdateFilter** an die Istgeschwindigkeit angeglichen. Ist die Zeitkonstante auf 0.0 gesetzt, wird die Istgeschwindigkeit direkt übernommen.

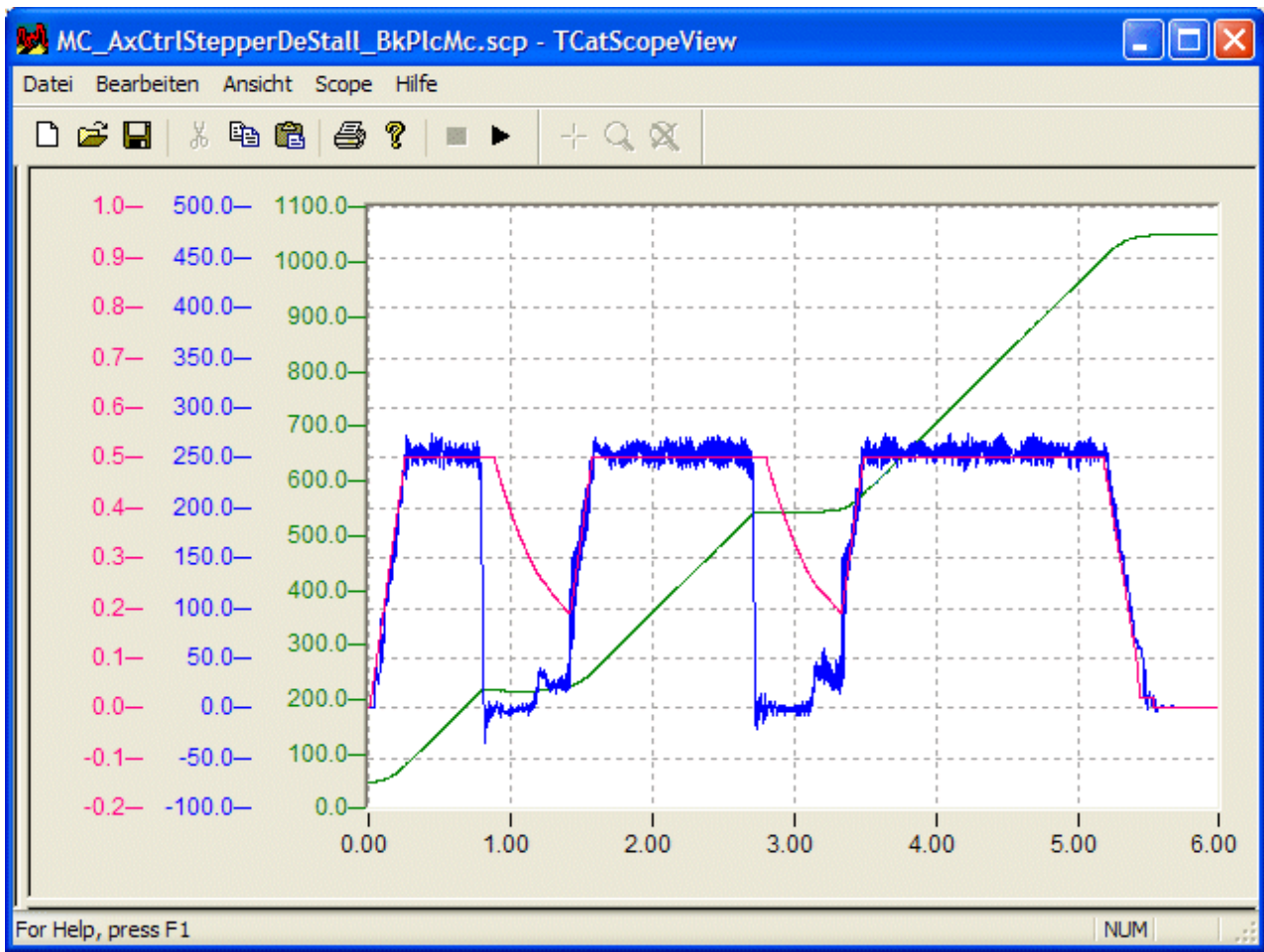
Im inaktiven Zustand wird **Activated** gelöscht, wenn die Achse den Ruhezustand verlässt und eine aktive Bewegung beginnt.



Da der Baustein die Differenz zwischen Soll- und Istgeschwindigkeit auswertet ist die korrekte Einstellung der Referenzgeschwindigkeit beim Einsatz des Bausteins wichtig. Eine zu ungenaue Einstellung dieses Parameters kann dazu führen, dass der Baustein ohne Notwendigkeit in die Bewegung eingreift.

Der nachstehende Scope View zeigt eine Positionierung, in deren Verlauf zwei Mal auf ein Hindernis aufgefahren wurde. Dadurch kam die Achse jeweils vollständig zum Stillstand.



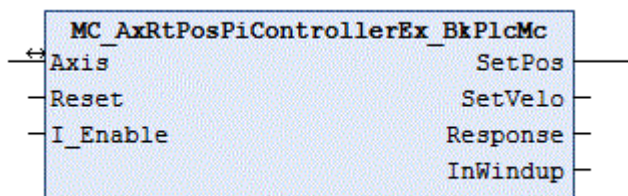


### Integration des Bausteins in die Applikation

Ein Baustein dieses Typs muss nach der Istwert-Erfassung und Stellwert-Generierung aufgerufen werden. Der Baustein überlagert seine Reaktion mit der des Stellwert-Generators und trägt sie in die [ST\\_TcHydAxRtData \[► 131\]](#) Struktur ein. Der nach dem Reglerbaustein zu platzierende [MC\\_AxRtFinish BkPlcMc \[► 230\]](#) Baustein berücksichtigt die Reaktion dann automatisch.

**i** Durch einen Wert größer als 0 in `FirstAuxParamIdx` kann der Baustein veranlasst werden drei aufeinander folgende Werte in den `fCustomerData` der Parameter-Struktur als `VeloLimit`, `LimitFilter` und `UpdateFilter` zu verwenden. Ist in `Axis.pStAxAuxLabels` die Adresse eines geeigneten `ARRAY[.] OF STRING()` eingetragen werden die Parameter automatisch mit einer Bezeichnung versehen.

#### 4.4.1.7 MC\_AxRtPosPiControllerEx\_BkPlcMc (ab V3.0.40)



Der Funktionsbaustein kann alternativ zum Default-Lageregler genutzt werden. Dazu wird er nach dem `MC_AxRuntime_BkPlcMc()` Baustein (Sollwertgenerator und Default-Lageregler) aufgerufen. Durch diese Anordnung überschreibt er die Reaktionen des Default-Lagereglers.

## Eingänge

```
VAR_INPUT
  Reset:          BOOL:=FALSE;
  I_Enable:      BOOL:=FALSE;
END_VAR
```

Name	Typ	Beschreibung
Reset	BOOL	Dieser Eingang löscht alle internen und externen Regler-Reaktionen.
I_Enable	BOOL	Dieser Eingang kontrolliert die Aktivität des I_Anteils.

## Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [ <a href="#">▶ 84</a> ] zu übergeben.

## Ausgänge

```
VAR_OUTPUT
  SetPos:        LREAL;
  SetVelo:       LREAL;
  Response:      LREAL;
  InWindup:     BOOL;
END_VAR
```

Name	Typ	Beschreibung
SetPos	LREAL	[mm] Die am internen Regler wirksam werdende Sollposition.
SetVelo	LREAL	[mm/s] Die am internen Regler wirksam werdende Sollgeschwindigkeit.
Response	LREAL	[mm/s] Die Regler-Reaktion.
InWindup	BOOL	Hier wird die aktiv gewordene Begrenzung des I-Anteils signalisiert.

## Aufgabe des Bausteins

Der im `MC_AxRuntime_BkPlcMc()` [[▶ 221](#)] Baustein integrierte Default-Lageregler kann mit seiner einfachen Struktur die regelungstechnischen Ansprüche einiger Applikationen nicht erfüllen. Für diese Einsatzfälle steht der `MC_AxRtPosPiControllerEx_BkPlcMc()` Baustein zur Verfügung. Er unterstützt folgende regelungstechnische Komponenten:

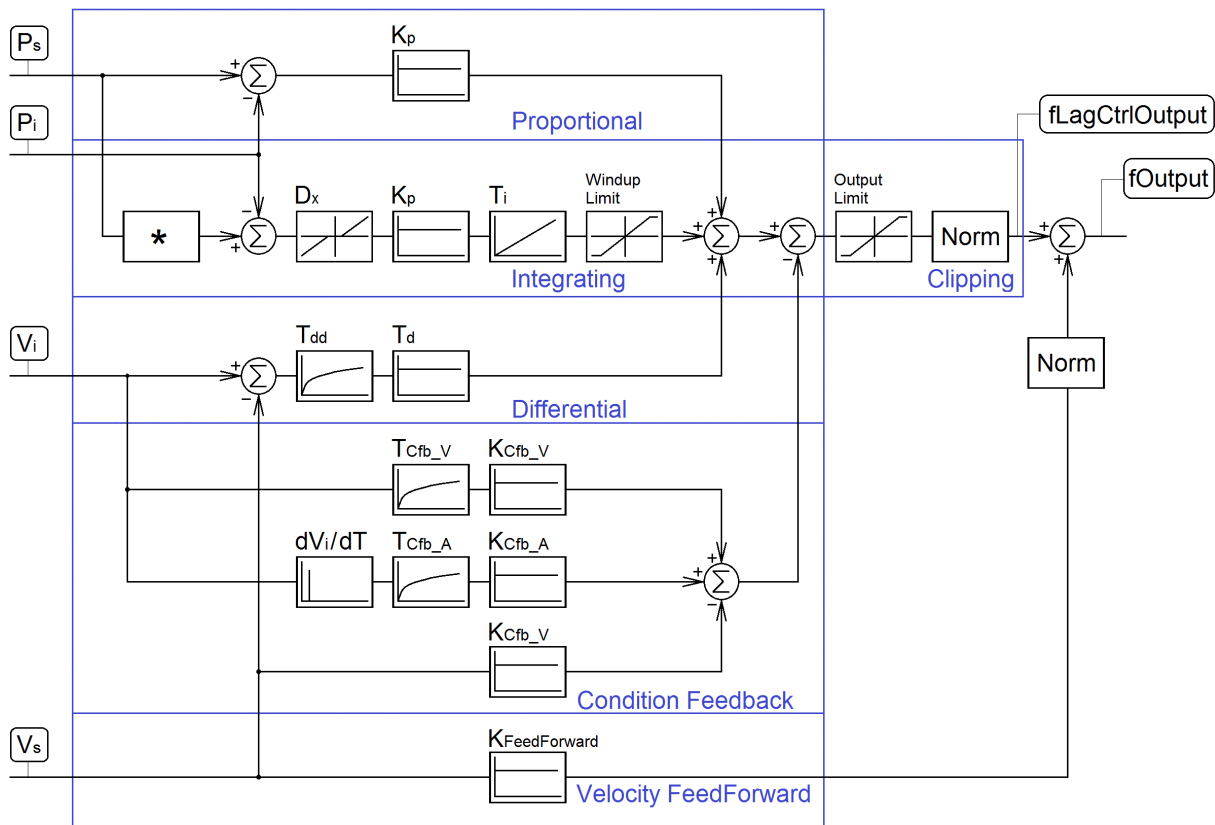
- Positions-P-Regler
- Positions-I-Regler mit Schwellwert und Windup-Limit
- Positions-D-Regler (realisiert als Geschwindigkeits-P-Regler) mit Dämpfungs-Zeit
- Zustandsrückführung für die Ist-Geschwindigkeit
- Zustandsrückführung für die Ist-Beschleunigung
- Kompensation der statischen Wirkung der Zustandsrückführung für die Ist-Geschwindigkeit

Die Umschaltung einer Geschwindigkeits-Vorsteuerung erfolgt nach dem Regler. Gleiches gilt für eventuell aktivierte Linearisierungen.



Der Regler ist mit V3.0.40 freigegeben. Die erweiterten Parameter werden durch den mit dieser Version freigegebenen `PlcMcManager` unterstützt.

Struktur des Reglers



Die mit \* gekennzeichnete Komponente bereitet bei weggesteuerter Sollwertgenerierung den Sollwert für den I-Anteil des Reglers auf. Dies wird dadurch erforderlich, dass die vom Sollwertgenerator bereitgestellte Sollposition beim Erreichen der Bremsstrecke auf die Zielposition springt. Bei zeitgesteuerter Sollwertgenerierung ist die Komponente transparent.



Hier nicht dargestellt: Ein TRUE an Reset oder eine fehlende Reglerfreigabe der Achse löscht sowohl den I-Anteil als auch den Ausgang des Reglers.

Der I-Anteil verfügt über einen Schwellwert  $D_x$ , der eine Reaktion auf kleine Abweichungen unterbindet. Dieser Parameter wird aus technischen Gründen auf mindestens  $2/3$  Inkrementgewichtung des Encoders begrenzt. Soll der I-Anteil inaktiv sein ist  $T_i$  auf Null zu stellen.

Die Implementation des D-Anteils nutzt die Tatsache aus, dass die differenzierte Sollposition vom Sollwertgenerator bereitgestellt wird. Eine Istgeschwindigkeit wird durch Differenzieren der Istposition ermittelt. Unter dieser Voraussetzung wirkt die Differentiations-Zeitkonstante  $T_d$  als Proportionalitätsfaktor. Soll der D-Anteil inaktiv sein ist die Zeitkonstante  $T_d$  auf Null zu stellen.

In der Zustandsrückführung (Condition Feedback) sind drei Zweige implementiert:

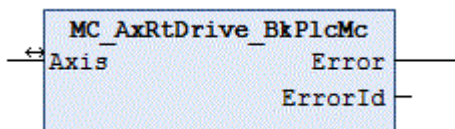
- Geschwindigkeitsaufschaltung: Die Ist-Geschwindigkeit wird gefiltert und mit einem Gewichtungsfaktor aufgeschaltet. Da sie subtrahiert wird erzeugt sie eine dämpfende Wirkung. Soll die Aufschaltung inaktiv sein ist  $K_{c_{fb\_V}}$  auf Null zu setzen.
- Beschleunigungsaufschaltung: Die Ist-Geschwindigkeit wird differenziert, gefiltert und mit einem Gewichtungsfaktor aufgeschaltet. Da sie subtrahiert wird erzeugt sie eine dämpfende Wirkung. Soll die Aufschaltung inaktiv sein ist  $K_{c_{fb\_A}}$  auf Null zu setzen.
- Eine Geschwindigkeitsaufschaltung erzeugt eine statisch wirksame Reduzierung der Geschwindigkeitsvorsteuerung. Bei weggesteuerter Positionierung erzeugt das eine erkennbare Geschwindigkeitsabweichung. Die ständig aktive Lageregelung bei zeitgesteuerter Positionierung kompensiert dies so weit möglich durch die Regelung. Diese unerwünschte Nebenwirkung der Geschwindigkeitsrückführung wird durch eine automatische Anpassung der Vorsteuerung behoben. Ein inaktiv stellen der Aufschaltung inaktiviert auch diese Kompensation.

Die Aufschaltung einer Geschwindigkeits-Vorsteuerung erfolgt nach dem Regler. Die Gewichtung ist bei weggesteuerter Sollwertgenerierung fest auf 1.0 gesetzt und kann nicht reduziert werden.

Eine eventuell aktivierte Linearisierung erfolgt nach dem Regler und wird hier nicht dargestellt.

## 4.4.2 Drive

### 4.4.2.1 MC\_AxRtDrive\_BkPlcMc (in V3.0)



Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einer Hardware-Baugruppe. Dazu wird in Abhängigkeit des in **Axis.ST\_TcHydAxParam** [▶ 120] als **nDrive\_Type** eingestellten Wertes ein Baustein aufgerufen, der die Besonderheiten der Hardware-Baugruppe berücksichtigt.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [▶ 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Error:      BOOL;
  ErrorID:    UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

### Verhalten des Bausteins

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **nDrive\_Type** in **pStAxParams** auf einen nicht zulässigen Wert eingestellt ist reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdDriveType**. Wenn der Pointer **pStAxRtData** der Achse initialisiert ist wird sie in einen Störzustand versetzt werden.
- Wenn einer der spezifischen Unterbausteine ein Problem erkennt wird dieser (sofern möglich) die Achse in einen Störzustand versetzen. Dieser Fehler wird an den Ausgängen des **MC\_AxRtDrive\_BkPlcMc** wiederholt.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird der Stellwert der Achse entsprechend **nDrive\_Type** [▶ 90] in **Axis.ST\_TcHydAxParam** [▶ 120] verarbeitet.

Informationen über die erforderlichen Verknüpfung von E/A-Komponenten mit den Eingangs- und Ausgangsstrukturen der Achse finden Sie in der Knowledge Base [▶ 302] unter FAQ #7.

Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollte zur Vereinfachung ein Baustein des Typs **MC\_AxStandardBody\_BkPlcMc** [▶ 237] verwendet werden.

Für einen asynchronen Datenaustausch mit E/A-Gerät der KL-Reihe stehen die Bausteine [MC\\_AxUtiReadRegDriveTerm\\_BkPlcMc \[► 276\]](#) und [MC\\_AxUtiWriteRegDriveTerm\\_BkPlcMc \[► 285\]](#) zur Verfügung.

### **iTcMc\_DriveAx2000\_B110A**

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines AX2000 Servostellers am EtherCAT Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Absolut-Encoder ausgerüstet ist. Wird ein Motor mit einem Resolver betrieben ist **iTcMc\_DriveAx2000\_B110R** einzustellen.

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Encoder-Bausteins und werden dort beschrieben. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderAX2000\\_B110A \[► 184\]](#).

### **iTcMc\_DriveAx2000\_B110R**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe an einen AX2000 Servoverstärker am EtherCAT Feldbus.

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Encoder-Bausteins und werden dort beschrieben. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderAx2000\\_B110R \[► 185\]](#).

### **iTcMc\_DriveAx2000\_B200R, iTcMc\_DriveAx2000\_B900R**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe an einen AX2000 Servoverstärker am Beckhoff Lightbus (B200) oder RtEthernet-Feldbus (B900).

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Encoder-Bausteins und werden dort beschrieben. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderAx2000\\_B200R \[► 186\]](#).

### **iTcMc\_DriveAx2000\_B750A**

Der Funktionsbaustein übernimmt (ab V3.0.26) die Aufbereitung des Stellwerts der Achse für die Ausgabe an einem AX2000 Servostellers am Sercos Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Absolut-Encoder ausgerüstet ist.

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Encoder-Bausteins und werden dort beschrieben. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderAX2000\\_B750A \[► 186\]](#).

Beachten Sie eine Reihe von Besonderheiten. Näheres hierzu finden Sie in der Knowledge Base.

### **iTcMc\_DriveAx5000\_B110A, iTcMc\_DriveAx5000\_B110SR**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe an einem AX5000 Servostellers am EtherCAT Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Absolut-Encoder ausgerüstet ist. Wird ein Motor mit einem Resolver betrieben **ist iTcMc\_EncoderAx5000\_B110SR einzustellen.**

---

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

---

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Encoder-Bausteins und werden dort beschrieben. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderAX5000\\_B110A \[► 187\]](#).

Unter [iTcMc\\_EncoderAX5000\\_B110A \[► 187\]](#) finden Sie eine Liste mit erfolgreich getesteten kompatiblen Geräten.

Beachten Sie eine Reihe von Besonderheiten. Näheres hierzu finden Sie in der Knowledge Base.

### **iTcMc\_DriveCoE\_DS402**

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines Servostellers mit CoE DS402 Profil am EtherCAT Feldbus.

---

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

---

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Encoder-Bausteins und werden dort beschrieben. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderCoE\\_DS402A \[► 188\]](#) und [iTcMc\\_EncoderCoE\\_DS402SR \[► 189\]](#).

Unter [iTcMc\\_EncoderCoE\\_DS402SR \[► 189\]](#) finden Sie eine Liste mit erfolgreich getesteten kompatiblen Geräten.

---

**i** Es werden derzeit nur Antriebe mit Resolver oder Singleturn-Encodern unterstützt.

---

### **iTcMc\_Drive\_CoE\_DS408**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe an ein Proportionalventil am EtherCAT Feldbus. Das Ventil muss dazu das CiA DS408 Profil unterstützen.

E/A-Variable	Interface.Variable	Verwendung
siehe Hinweis	ST_TcPlcDeviceInput.nDacOut	Ausgabe des Geschwindigkeits-Signals.
siehe Hinweis	ST_TcPlcDeviceOutput.uiDriveCtrl	Steuerung des Geräts.
siehe Hinweis	ST_TcPlcDeviceInput.uiStatus	Status des Geräts

E/A-Variable	Interface.Variable	Verwendung
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Überwachung des Onlinestatus
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Automatische Identifikation.



Die Namen der mit dem Gerät ausgetauschten Prozessdaten werden durch die XML Datei des Herstellers festgelegt.

Die nachstehenden Index.SubIndex-Kombinationen müssen vom Ventil unterstützt werden.

Index	Subindex	Bedeutung
1000	0	Identifikation
1008	0	Gerätebezeichnung (optional)
1018	1	Hersteller-Kennung
1018	2	Gerätetyp

Die nachstehende Liste kompatibler Geräte ist naturgemäß unvollständig. Sie stellt keine Empfehlung dar sondern ist nur als allgemeiner Hinweis zu verstehen. Der problemlose Einsatz der genannten Geräte kann von Beckhoff nicht garantiert werden. Ist ein Hersteller oder eines seiner Geräte nicht aufgeführt kann ein problemloser Betrieb durchaus gegeben sein, wird jedoch nicht zugesichert.

Hersteller	Typ	Beschreibung
Moog	D638Exxx	Proportionalventil
Parker	DxxFP /DxxFE /TDP /TPQ	Proportionalventil

x: Stellt einen Platzhalter für verschiedene Zeichen dar.

### iTcMc\_DriveI2512\_1Coil

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einen PWM-Feldbusmodul IP2512.

E/A-Variable	Interface.Variable	Verwendung
Daten Aus	ST_TcPlcDeviceOutput.uiDacOutA	Ausgabe des PWM-Faktors.

### iTcMc\_DriveI2512\_2Coil

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einen PWM-Feldbusmodul IP2512.

E/A-Variable	Interface.Variable	Verwendung
Daten Aus	ST_TcPlcDeviceOutput.uiDacOutA	Ausgabe des PWM-Faktors für Spule 1.
Daten Aus	ST_TcPlcDeviceOutput.uiDacOutB	Ausgabe des PWM-Faktors für Spule 2.

### iTcMc\_DriveEL2535

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einer stromgeregelten PWM-Ausgabeklemme. Diese Klemme stellt zwei unabhängige Endstufen bereit und kann für die folgenden Ventiltypen eingesetzt werden:



F Proportionalventil mit Federmittelstellung und zwei Spulen ohne Permanentmagnete:  
nDrive\_Type = iTcMc\_DriveEL2535\_2Coilolgen

Hier werden beide Kanäle für ein Ventil benötigt. Die Klemme kann nicht für ein weiteres Ventil mitgenutzt werden.

Bei diesem Ventiltyp wird ein Anteil der Vollbestromung in der richtungswirksamen Spule bei stromloser Gegenspule benötigt um den Schieber in die gewünschte Stellung zu bewegen. Der Klemmenbaustein erzeugt für -100 % .. 0 % .. +100 % Ansteuerung die Ausgabewerte 0 .. 0 .. 32767 in uiDacOutA und 32767 .. 0 .. 0 in uiDacOutB.

E/A-Variable	Interface.Variable	Verwendung
Channel1.PWM Output	ST_TcPlcDeviceOutput.uiDacOutA	Ausgabe des PWM-Faktors für Spule 1.
Channel2.PWM Output	ST_TcPlcDeviceOutput.uiDacOutB	Ausgabe des PWM-Faktors für Spule 2.
	ST_TcPlcDeviceOutput.nDacOut	NICHT VERWENDEN!
Channel1.Status	ST_TcPlcDeviceInput.uiStatus	Status des ersten Kanals des Geräts
Channel2.Status	ST_TcPlcDeviceInput.uiTerminalState2	Status des zweiten Kanals des Geräts
Channel1.Control	ST_TcPlcDeviceOutput.uiDriveCtrl	Steuerung ersten Kanals des Geräts
Channel2.Control	ST_TcPlcDeviceOutput.uiTerminalCtrl2	Steuerung zweiten Kanals des Geräts
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Automatische Identifikation, Parametrierung.



Proportionalventil mit Federendstellung und einer Spule ohne Permanentmagnete:  
nDrive\_Type = iTcMc\_DriveEL2535\_1Coil

Hier wird nur ein Kanal benötigt. Die Klemme kann für ein weiteres Ventil mitgenutzt werden. Dafür sind die E/A-Variablen des zweiten Kanals zu nutzen.

Bei diesem Ventiltyp wird 50% der Vollbestromung benötigt um den Schieber in die Mittelstellung zu bewegen. Der Klemmenbaustein erzeugt für -100 % .. 0 % .. +100 % Ansteuerung die Ausgabewerte 0 .. 16384 .. 32767.

E/A-Variable	Interface.Variable	Verwendung
	ST_TcPlcDeviceOutput.uiDacOutA	NICHT VERWENDEN!
	ST_TcPlcDeviceOutput.uiDacOutB	NICHT VERWENDEN!
Channel1.PWM Output	ST_TcPlcDeviceOutput.nDacOut	Ausgabe des PWM-Faktors.
Channel1.Status	ST_TcPlcDeviceInput.uiStatus	Status des Geräts
Channel1.Control	ST_TcPlcDeviceOutput.uiDriveCtrl	Steuerung des Geräts.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Automatische Identifikation, Parametrierung.



Proportionalventil mit Federmittelstellung und einer Spule mit Permanentmagnete:  
nDrive\_Type = iTcMc\_DriveEL2535\_1Coil



Hier wird nur ein Kanal benötigt. Die Klemme kann für ein weiteres Ventil mitgenutzt werden. Dafür sind die E/A-Variablen des zweiten Kanals zu nutzen.

Bei diesem Ventiltyp wird eine bipolare Bestromung benötigt, die dem Arbeitsprinzip einer ±10V Klemme entspricht. Der vom Klemmenbaustein erzeugte Ausgabewert ist NACH Aufruf des Drive-Bausteins durch die Applikation wie folgt anzupassen:

$ST\_TcPlcDeviceOutput.nDacOut := 2 * (ST\_TcPlcDeviceOutput.nDacOut - 16384);$

E/A-Variable	Interface.Variable	Verwendung
	ST_TcPlcDeviceOutput.uiDacOutA	NICHT VERWENDEN!
	ST_TcPlcDeviceOutput.uiDacOutB	NICHT VERWENDEN!
Channel1.PWM Output	ST_TcPlcDeviceOutput.nDacOut	Ausgabe des PWM-Faktors.
Channel1.Status	ST_TcPlcDeviceInput.uiStatus	Status des Geräts
Channel1.Control	ST_TcPlcDeviceOutput.uiDriveCtrl	Steuerung des Geräts.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Automatische Identifikation, Parametrierung.

**iTcMc\_DriveEL4132**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einer ±10 V Ausgabeklemme.

E/A-Variable	Interface.Variable	Verwendung
Output	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.

**iTcMc\_DriveEL7031**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einer Schrittmotorendstufenklemme EL7031.

E/A-Variable	Interface.Variable	Verwendung
STM Velocity.Velocity	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals.
STM Control.Control	ST_TcPlcDeviceOutput.uiDriveCtrl	Betrieb: Steuerung der Endstufe.
STM Status.Status	ST_TcPlcDeviceInput.uiStatus	Betrieb: Status der Endstufe.
WcState.WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung, Zustandsüberwachung.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Kommunikation.

**iTcMc\_DriveEL7041**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einer Schrittmotorendstufenklemme EL7041.

E/A-Variable	Interface.Variable	Verwendung
STM Velocity.Velocity	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals.
STM Control.Control	ST_TcPlcDeviceOutput.uiDriveCtrl	Betrieb: Steuerung der Endstufe.
STM Status.Status	ST_TcPlcDeviceInput.uiStatus	Betrieb: Status der Endstufe.
ENC Status.Counter Value	ST_TcPlcDeviceInput.uiCount	Betrieb: Einlesen der Ist-Position.
ENC Status.Latch Value	ST_TcPlcDeviceInput.uiLatch	Betrieb: Einlesen der Latch-Position.
ENC Status.Status	ST_TcPlcDeviceInput.uiTerminalState2	Betrieb: Status der Encoder-Schnittstelle.
ENC Control.Control	ST_TcPlcDeviceOutput.uiTerminalCtrl2	Betrieb: Steuerung der Encoder-Schnittstelle.
WcState.WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung, Zustandsüberwachung.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Kommunikation.

### iTcMc\_DriveEL7201

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf eine Servoklemme EL7201.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Encoder-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_EncoderEL7201.

E/A-Variable	Interface.Variable	Verwendung
Target velocity	ST_TcPlcDeviceOutput.NominalVelocity	Betrieb: Ausgabe des Geschwindigkeits-Signals.
Controlword	ST_TcPlcDeviceOutput.uiDriveCtrl	Betrieb: Steuerung der Endstufe.
Position actual value	ST_TcPlcDeviceInput.udiCount	Betrieb: Einlesen der Ist-Position.
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
Statusword	ST_TcPlcDeviceInput.uiStatus	Betrieb: Status der Endstufe.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung, Zustandsüberwachung.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Kommunikation

### iTcMc\_DriveKL2521

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf eine Pulsausgabeklemme KL2521.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Encoder-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_EncoderKL2521 [► 195].

E/A-Variable	Interface.Variable	Verwendung
Daten Ein	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position.

E/A-Variable	Interface.Variable	Verwendung
		Bei <u>Register-Kommunikation</u> [▶ 317]: Schnittstelle für gelesene Daten.
Kontrolle	ST_TcPlcDeviceOutput.bTerminalCtrl	Register-Kommunikation
Status	ST_TcPlcDeviceInput.bTerminalState	Register-Kommunikation
Daten Aus	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Register-Kommunikation: Schnittstelle für geschriebene Daten.

**iTcMc\_DriveKL2531**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf eine Schrittmotorendstufenklemme KL2531.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Encoder-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_EncoderKL2531 [▶ 195].

E/A-Variable	Interface.Variable	Verwendung
Velocity	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Bei <u>Register-Kommunikation</u> [▶ 317]: Schnittstelle für geschriebene Daten.
Position	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Ctrl	ST_TcPlcDeviceOutput.bTerminalCtrl	Steuerung der Endstufe, Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Status der Endstufe, Register-Kommunikation.
ExtStatus	ST_TcPlcDeviceInput.uiTerminalState2	Diagnose von Endstufe und Motor

**iTcMc\_DriveKL2532**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf eine DC-Motorendstufenklemme KL2532.

E/A-Variable	Interface.Variable	Verwendung
Daten ein	ST_TcPlcDeviceInput.uiTerminalData	Bei <u>Register-Kommunikation</u> [▶ 317]: Schnittstelle für gelesene Daten.
Kontrolle	ST_TcPlcDeviceOutput.bTerminalCtrl	Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Register-Kommunikation
Daten aus	ST_TcPlcDeviceOutput.nDacOut	Register-Kommunikation

**iTcMc\_DriveKL2535\_1Coil, iTcMc\_DriveKL2535\_2Coil**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf eine PWM-Endstufenklemme KL2535.

E/A-Variable	Interface.Variable	Verwendung
Daten ein	ST_TcPlcDeviceInput.uiTerminalData	<a href="#">Register-Kommunikation [► 317]</a>
Kontrolle	ST_TcPlcDeviceOutput.bTerminalCtrl	Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Register-Kommunikation
Daten aus	ST_TcPlcDeviceOutput.nDacOut	Register-Kommunikation

**iTcMc\_DriveKL2541**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf eine Schrittmotorendstufenklemme KL2541.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Encoder-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderKL2541 \[► 196\]](#).

E/A-Variable	Interface.Variable	Verwendung
Velocity	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Bei <a href="#">Register-Kommunikation [► 317]</a> : Schnittstelle für geschriebene Daten.
Position	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Ctrl	ST_TcPlcDeviceOutput.bTerminalCtrl	Steuerung der Endstufe, Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Status der Endstufe, Register-Kommunikation.
ExtCtrl	ST_TcPlcDeviceOutput.uiTerminalCtrl2	Latch-Steuerung bei Homing mit dem Synchron-Impuls des Encoders
ExtStatus	ST_TcPlcDeviceInput.uiTerminalState2	Diagnose von Endstufe und Motor, Latch-Status bei Homing mit dem Synchron-Impuls des Encoders

**iTcMc\_DriveKL2542**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf eine DC-Motorendstufenklemme KL2542.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Encoder-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_EncoderKL2542 \[► 196\]](#).

E/A-Variable	Interface.Variable	Verwendung
Daten aus	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals.

E/A-Variable	Interface.Variable	Verwendung
		Bei Register-Kommunikation [▶ 317]: Schnittstelle für geschriebene Daten.
Daten ein	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Kontrolle	ST_TcPlcDeviceOutput.bTerminalCtrl	Steuerung der Endstufe, Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Status der Endstufe, Register-Kommunikation.

**iTcMc\_DriveKL4032**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einer ±10 V Ausgabeklemme.

E/A-Variable	Interface.Variable	Verwendung
Daten Aus	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Bei Register-Kommunikation [▶ 317]: Schnittstelle für geschriebene Daten.
Kontrolle	ST_TcPlcDeviceOutput.bTerminalCtrl	Register-Kommunikation
Status	ST_TcPlcDeviceInput.bTerminalState	Register-Kommunikation
Daten Ein	ST_TcPlcDeviceOutput.uiTerminalData	Register-Kommunikation: Schnittstelle für gelesene Daten.

**iTcMc\_DriveLowCostStepper**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf digitalen Ausgangsklemmen. Für die Emulation einer Istposition wird ein Pulszähler aktualisiert, der mit einem iTcMc\_EncoderLowCostStepper [▶ 198] Encoder ausgewertet werden kann.

E/A-Variable	Interface.Variable	Verwendung
Ausgang	ST_TcPlcDeviceOutput.nDigOutAp	Nichtinvertierte Ansteuerung der A-Phase.
Ausgang	ST_TcPlcDeviceOutput.nDigOutAn	Invertierte Ansteuerung der A-Phase.
Ausgang	ST_TcPlcDeviceOutput.nDigOutBp	Nichtinvertierte Ansteuerung der B-Phase.
Ausgang	ST_TcPlcDeviceOutput.nDigOutBn	Invertierte Ansteuerung der B-Phase.

**iTcMc\_DriveLowCostInverter**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf digitalen Ausgangsklemmen für den Betrieb einer Polwendeschutz-Schaltung oder eines Frequenzumrichters mit Festfrequenzen. Beim Einsatz dieses Antriebstyps ist die Beachtung einiger Besonderheiten erforderlich. Bei der Verknüpfung sind zwei Varianten zu unterscheiden:

**Bremse, Freigabe, Richtung und Geschwindigkeitsstufe**

Nach dem Aufruf des MC\_AxRtFinish\_BkPlcMc [▶ 230]- oder MC\_AxStandardBody\_BkPlcMc [▶ 237]-Bausteins der Achse stehen vier decodierte Signale zur Verfügung. Um die hier benötigten Signale zu erzeugen sind nach dem Bausteinaufruf die nachstehenden Zusammenfassungen der richtungsspezifischen Signale

erforderlich.

**Beispiel:**

stAxDeviceOut.bDigOutAp:=stAxDeviceOut.bDigOutAp OR stAxDeviceOut.bDigOutBp;

stAxDeviceOut.bDigOutAn:=stAxDeviceOut.bDigOutAn OR stAxDeviceOut.bDigOutBn;



Ab V3.0.11 kann auf dem Ventil-Reiter die Ausgabe eines Absolutwerts aktiviert werden. Die oben gezeigte Zusammenfassung der Signale für diesen Einsatzfall wird dann intern vorgenommen.

E/A-Variable	Interface.Variable	Verwendung
Ausgang	ST_TcPlcDeviceOutput.nDigOutAp	Anwahl der Festfrequenz für den Eilgang.
Ausgang	ST_TcPlcDeviceOutput.nDigOutAn	Anwahl der Festfrequenz für den Schleichgang.
Ausgang	ST_TcPlcDeviceOutput.bMovePos	Vorgabe der Bewegungsrichtung: Positiv.
Ausgang	ST_TcPlcDeviceOutput.bMoveNeg	Vorgabe der Bewegungsrichtung: Negativ.
Ausgang	ST_TcPlcDeviceOutput.bPowerOn	Freigabe der Leistungsstufe.
Ausgang	ST_TcPlcDeviceOutput.bBrakeOff	Ansteuerung der Bremse.
Eingang	ST_TcPlcDeviceInput.bPowerOk	Status des Umrichters: Betriebsbereitschaft.

**Bremse, Freigabe und richtungscodierte Geschwindigkeitsstufe**

E/A-Variable	Interface.Variable	Verwendung
Ausgang	ST_TcPlcDeviceOutput.nDigOutAp	Anwahl der Festfrequenz für den Eilgang in positiver Bewegungsrichtung.
Ausgang	ST_TcPlcDeviceOutput.nDigOutAn	Anwahl der Festfrequenz für den Schleichgang in positiver Bewegungsrichtung.
Ausgang	ST_TcPlcDeviceOutput.nDigOutBn	Anwahl der Festfrequenz für den Schleichgang in negativer Bewegungsrichtung.
Ausgang	ST_TcPlcDeviceOutput.nDigOutBp	Anwahl der Festfrequenz für den Eilgang in negativer Bewegungsrichtung.
Ausgang	ST_TcPlcDeviceOutput.bPowerOn	Freigabe der Leistungsstufe.
Ausgang	ST_TcPlcDeviceOutput.bBrakeOff	Ansteuerung der Bremse.
Eingang	ST_TcPlcDeviceInput.bPowerOk	Status des Umrichters: Betriebsbereitschaft.

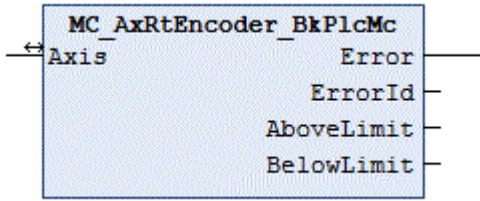
**iTcMc\_DriveM2400\_Dn**

Der Funktionsbaustein übernimmt die Aufbereitung des Stellwerts der Achse für die Ausgabe auf einem von vier Kanälen einer  $\pm 10$  V M2400 Ausgabebox.

E/A-Variable	Interface.Variable	Verwendung
Daten Aus	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals.

### 4.4.3 Encoder

#### 4.4.3.1 MC\_AxRtEncoder\_BkPlcMc (ab V3.0)



Der Funktionsbaustein übernimmt die Ermittlung der Istposition der Achse aus den Eingangsinformationen einer Hardware-Baugruppe. Dazu wird in Abhängigkeit des in **Axis.ST TcHydAxParam** [▶ 120] als **nEnc\_Type** eingestellten Wertes ein Baustein aufgerufen, der die Besonderheiten der Hardware-Baugruppe berücksichtigt.

Für die Ermittlung der wesentlichen Istwerte einer servoelektrisch-hydraulischen Hybrid-Achse steht mit **MC\_AxRtHybridAxisActuals\_BkPlcMc** [▶ 209] ein angepasster Baustein zur Verfügung.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
  AboveLimit:     BOOL;
  BelowLimit:     BOOL;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
AboveLimit	BOOL	Hier wird das Überschreiten des oberen Software-Endschalters durch die Ist-Position signalisiert.
BelowLimit	BOOL	Hier wird das Unterschreiten des unteren Software-Endschalters durch die Ist-Position signalisiert.

#### Verhalten des Bausteins

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **nEnc\_Type** in pStAxParams auf einen nicht zulässigen Wert eingestellt ist reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdEncType**. Die Achse wird in einen Störzustand versetzt.
- Wenn einer der spezifischen Unterbausteine ein Problem erkennt wird dieser (sofern möglich) die Achse in einen Störzustand versetzen. Dieser Fehler wird an den Ausgängen des **MC\_AxRtEncoder\_BkPlcMc** wiederholt.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Istposition der Achse entsprechend `nEnc_Type` [► 93] in `Axis.ST_TcHydAxParam` [► 120] durch Aufruf eines entsprechenden typspezifischen Bausteins ermittelt.

Der Ausgang **AboveLimit** ist TRUE wenn die Istposition einen Wert größer als **fSoftEndMax** in `Axis.ST_TcHydAxParam` [► 120] annimmt. Liegt die Istposition unter **fSoftEndMin** wird dies in **BelowLimit** signalisiert. Diese Signale werden unabhängig von **bSoftEndMaxEna** und **bSoftEndMinEna** erzeugt.

Informationen über die erforderlichen Verknüpfung von E/A-Komponenten mit den Eingangs- und Ausgangsstrukturen der Achse finden Sie in der Knowledge Base unter [FAQ #4](#) [► 306].

Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollte zur Vereinfachung ein Baustein des Typs `MC_AxStandardBody_BkPlcMc` [► 237] verwendet werden.

Für einen asynchronen Datenaustausch mit E/A-Gerät der KL-Reihe stehen die Bausteine `MC_AxUtiReadRegEncTerm_BkPlcMc` [► 277] und `MC_AxUtiWriteRegEncTerm_BkPlcMc` [► 286] zur Verfügung.

### iTcMc\_EncoderAx2000\_B110A

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines AX2000 Servostellers am EtherCAT Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Absolut-Encoder ausgerüstet ist. Wird ein Motor mit einem Resolver betrieben ist **iTcMc\_EncoderAx2000\_B110R** einzustellen.



Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch `iTcMc_DriveAX2000_B110R` [► 173].

E/A-Variable	Interface.Variable	Verwendung
Position actual value	<code>ST_TcPlcDeviceInput.ActualPos[0..1]</code>	Ermittlung der Ist-Position.
Status word	<code>ST_TcPlcDeviceInput.uiStatus</code>	Status des Geräts, der Encodernachbildung.
Control word	<code>ST_TcPlcDeviceOutput.uiDriveCtrl</code>	Steuerung des Geräts.
Velocity demand value	<code>ST_TcPlcDeviceOutput.NominalVelocity</code>	Ausgabe des Geschwindigkeits-Stellwerts.
WcState (siehe Hinweis)	<code>ST_TcPlcDeviceInput.wEncWcState</code>	Verbindungsüberwachung für die Istwerterfassung.
WcState	<code>ST_TcPlcDeviceInput.wDriveWcState</code>	Verbindungsüberwachung für den Antrieb.
InfoData.State	<code>ST_TcPlcDeviceInput.uiDriveBoxState</code>	Überwachung des Onlinestatus
InfoData.AdsAddr (siehe Hinweis)	<code>ST_TcPlcDeviceInput.sEncAdsAddr</code>	Parameterkommunikation.
InfoData.AdsAddr	<code>ST_TcPlcDeviceInput.sDrvAdsAddr</code>	Steuerung des Echtzeitstatus, Parameterkommunikation.
Chn0 (siehe Hinweis)	<code>ST_TcPlcDeviceInput.nEncAdsChannel</code>	Parameterkommunikation.
Chn0	<code>ST_TcPlcDeviceInput.nDrvAdsChannel</code>	Steuerung des Echtzeitstatus, Parameterkommunikation.
Ausgang (auf einer DA-Klemme)	<code>ST_TcPlcDeviceOutput.PowerOn</code>	Optionale Ansteuerung des Netzschütz. Hierfür ist eine Digital-Ausgangsklemme vorzusehen.
Eingang (auf einer DE-Klemme)	<code>ST_TcPlcDeviceInput.PowerOk</code>	Optionale Auswertung des Netzschütz. Hierfür ist eine Digital-Eingangsklemme vorzusehen.



**i** Um die Erstellung der E/A-Verknüpfung zu vereinfachen kann dann auf die Verknüpfung von ST\_TcPlcDeviceInput.sEncAdsAddr, ST\_TcPlcDeviceInput.nEncAdsChannel und ST\_TcPlcDeviceInput.wEncWcState verzichtet werden, wenn wie üblich die Istwerterfassung über dasselbe Gerät erfolgt. Die Bausteine für Parameterkommunikation und Encoderauswertung verwenden dann die entsprechenden Variablen der Antriebs-Verknüpfung.

**iTcMc\_EncoderAx2000\_B110R**

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines AX2000 Servostellers am EtherCAT Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Resolver ausgerüstet ist. Wird ein Motor mit einem Absolut-Encoder betrieben ist **iTcMc\_EncoderAx2000\_B110A** einzustellen.

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_DriveAX2000\_B110R [► 173].

E/A-Variable	Interface.Variable	Verwendung
Position actual value	ST_TcPlcDeviceInput.ActualPos[0..1]	Ermittlung der Ist-Position.
Status word	ST_TcPlcDeviceInput.uiStatus	Status des Geräts, der Encodernachbildung.
Control word	ST_TcPlcDeviceOutput.uiDriveCtrl	Steuerung des Geräts.
Velocity demand value	ST_TcPlcDeviceOutput.NominalVelocity	Ausgabe des Geschwindigkeits-Stellwerts.
WcState (siehe Hinweis)	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung für die Istwerterfassung.
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung für den Antrieb.
uiDriveBoxState	ST_TcPlcDeviceInput.InfoData.State	Überwachung des Onlinestatus
InfoData.AdsAddr (siehe Hinweis)	ST_TcPlcDeviceInput.sEncAdsAddr	Parameterkommunikation.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Parameterkommunikation.
Chn0 (siehe Hinweis)	ST_TcPlcDeviceInput.nEncAdsChannel	Parameterkommunikation.
Chn0	ST_TcPlcDeviceInput.nDrvAdsChannel	Parameterkommunikation.
Ausgang (auf einer DA-Klemme)	ST_TcPlcDeviceOutput.PowerOn	Optionale Ansteuerung des Netzschütz. Hierfür ist eine Digital-Ausgangsklemme vorzusehen.
Eingang (auf einer DE-Klemme)	ST_TcPlcDeviceInput.PowerOk	Optionale Auswertung des Netzschütz. Hierfür ist eine Digital-Eingangsklemme vorzusehen.

**i** Um die Erstellung der E/A-Verknüpfung zu vereinfachen kann dann auf die Verknüpfung von ST\_TcPlcDeviceInput.sEncAdsAddr, ST\_TcPlcDeviceInput.nEncAdsChannel und ST\_TcPlcDeviceInput.wEncWcState verzichtet werden, wenn wie üblich die Istwerterfassung über dasselbe Gerät erfolgt. Die Bausteine für Parameterkommunikation und Encoderauswertung verwenden dann die entsprechenden Variablen der Antriebs-Verknüpfung.

**iTcMc\_EncoderAx2000\_B200R, iTcMc\_EncoderAx2000\_B900R**

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines AX2000 Servostellers mit Lightbus (B200) bzw. RealtimeEthernet (B900).

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_DriveAX2000\_B200R [► 173].

E/A-Variable	Interface.Variable	Verwendung
ActualPos[0..1]	ST_TcPlcDeviceInput.ActualPos[0..1]	Ermittlung der Ist-Position.
DriveError	ST_TcPlcDeviceInput.DriveError	Status des Geräts.
DriveState[0..3]	ST_TcPlcDeviceInput.DriveState[0..3]	Status des Geräts.
BoxState	ST_TcPlcDeviceInput.uiDriveBoxState	Verbindungsüberwachung.
DriveCtrl0	ST_TcPlcDeviceOutput.DriveCtrl[0]	Steuerung des Geräts.
DriveCtrl1	ST_TcPlcDeviceOutput.DriveCtrl[1]	Steuerung des Geräts.
DriveCtrl2	ST_TcPlcDeviceOutput.DriveCtrl[2]	Steuerung des Geräts.
DriveCtrl3	ST_TcPlcDeviceOutput.DriveCtrl[3]	Steuerung des Geräts.
NominalVelo	ST_TcPlcDeviceOutput.NominalVelo	Ausgabe des Geschwindigkeits-Stellwerts.
Ausgang (auf einer DA-Klemme)	ST_TcPlcDeviceOutput.PowerOn	Optionale Ansteuerung des Netzschütz. Hierfür ist eine Digital-Ausgangsklemme vorzusehen.
Eingang (auf einer DE-Klemme)	ST_TcPlcDeviceInput.PowerOk	Optionale Auswertung des Netzschütz. Hierfür ist eine Digital-Eingangsklemme vorzusehen.

**iTcMc\_EncoderAx2000\_B750A**

Der Funktionsbaustein übernimmt (ab V3.0.26) die Auswertung der Istwerte eines AX2000 Servostellers am Sercos Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Absolut-Encoder ausgerüstet ist.

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_DriveAX2000\_B750A [► 173].

E/A-Variable	Interface.Variable	Verwendung
Antriebs-Statuswort	ST_TcPlcDeviceInput.uiStatus	Status des Geräts.
Lage-Istwert Geber 1	ST_TcPlcDeviceInput.udiCount	Ermittlung der Ist-Position.
Master-Steuerwort	ST_TcPlcDeviceOutput.uiDriveCtrl	Steuerung des Geräts.
Geschwindigkeits-Sollwert	ST_TcPlcDeviceOutput.NominalVelo	Ausgabe des Geschwindigkeits-Stellwerts.

E/A-Variable	Interface.Variable	Verwendung
SystemState (vom Sercos Master)	ST_TcPlcDeviceInput.uiDriveBoxState	Überwachung der Sercos-Phase. <b>Diese Variable wird vom Sercos Master (z.B. FC7501) bereitgestellt.</b>
Ausgang (auf einer DA-Klemme)	ST_TcPlcDeviceOutput.PowerOn	Optionale Ansteuerung des Netzschütz. Hierfür ist eine Digital-Ausgangsklemme vorzusehen.
Eingang (auf einer DE-Klemme)	ST_TcPlcDeviceInput.PowerOk	Optionale Auswertung des Netzschütz. Hierfür ist eine Digital-Eingangsklemme vorzusehen.

Beachten Sie eine Reihe von Besonderheiten. Näheres hierzu finden Sie in der [Knowledge Base \[► 302\]](#).

**iTcMc\_EncoderAx5000\_B110A, iTcMc\_EncoderAx5000\_B110SR**

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines AX5000 Servostellers am EtherCAT Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Absolut-Encoder ausgerüstet ist. Wird ein Motor mit einem Resolver betrieben ist **iTcMc\_EncoderAx5000\_B110SR** einzustellen.

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_DriveAX5000\\_B110A \[► 174\]](#).

E/A-Variable	Interface.Variable	Verwendung
Position feedback 1 value	ST_TcPlcDeviceInput.udiCount	Ermittlung der Ist-Position.
Drive status word	ST_TcPlcDeviceInput.uiStatus	Status des Geräts.
Master control word	ST_TcPlcDeviceOutput.uiDriveCtrl	Steuerung des Geräts.
Velocity command value	ST_TcPlcDeviceOutput.NominalVelocity	Ausgabe des Geschwindigkeits-Stellwerts.
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung für den Antrieb.
WcState (siehe Hinweis)	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung für die Istwerterfassung.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Überwachung des Onlinestatus
InfoData.AdsAddr	ST_TcPlcDeviceInput.sDrvAdsAddr	Steuerung des Echtzeitstatus, Parameterkommunikation.
InfoData.AdsAddr (siehe Hinweis)	ST_TcPlcDeviceInput.sEncAdsAddr	Parameterkommunikation.
Chn0 (siehe Hinweis 2)	ST_TcPlcDeviceInput.nDrvAdsChannel	für Einfachgeräte oder den ersten Antrieb eines Doppelgeräts: Steuerung des Echtzeitstatus, Parameterkommunikation.
Chn0 (siehe Hinweise 1,2)	ST_TcPlcDeviceInput.nEncAdsChannel	für Einfachgeräte oder den ersten Antrieb eines Doppelgeräts: Parameterkommunikation.
Chn1 (siehe Hinweis 2)	ST_TcPlcDeviceInput.nDrvAdsChannel	nur für den zweiten Antrieb eines Doppelgeräts: Steuerung des Echtzeitstatus, Parameterkommunikation.
Chn1 (siehe Hinweise 1,2)	ST_TcPlcDeviceInput.nEncAdsChannel	nur für den zweiten Antrieb eines Doppelgeräts: Parameterkommunikation.

E/A-Variable	Interface.Variable	Verwendung
Ausgang (auf einer DA-Klemme)	ST_TcPlcDeviceOutput.PowerOn	Optionale Ansteuerung des Netzschütz. Hierfür ist eine Digital-Ausgangsklemme vorzusehen.
Eingang (auf einer DE-Klemme)	ST_TcPlcDeviceInput.PowerOk	Optionale Auswertung des Netzschütz. Hierfür ist eine Digital-Eingangsklemme vorzusehen.

Die nachstehende Liste kompatibler Geräte ist naturgemäß unvollständig. Sie stellt keine Empfehlung dar sondern ist nur als allgemeiner Hinweis zu verstehen. Der problemlose Einsatz der genannten Geräte kann von Beckhoff nicht garantiert werden. Ist ein Hersteller oder eines seiner Geräte nicht aufgeführt kann ein problemloser Betrieb durchaus gegeben sein, wird jedoch nicht zugesichert.

Hersteller	Typ	Beschreibung
Baumüller	b-maxx	Servoregler mit Singleturn-Absolutencoder

**i** Um die Erstellung der E/A-Verknüpfung zu vereinfachen kann dann auf die Verknüpfung von ST\_TcPlcDeviceInput.sEncAdsAddr, ST\_TcPlcDeviceInput.nEncAdsChannel und ST\_TcPlcDeviceInput.wEncWcState verzichtet werden, wenn wie üblich die Istwerterfassung über dasselbe Gerät erfolgt. Die Bausteine für Parameterkommunikation und Encoderauswertung verwenden dann die entsprechenden Variablen der Antriebs-Verknüpfung.

**i** Die Variablen Chn0 und Chn2 sind für die Unterscheidung der Kanäle eines Doppelgeräts erforderlich. Für den ersten Antrieb des Geräts ist Chn0 zu verbinden, für den zweiten entsprechend Chn1. Bei Einfachgeräten ist wie für den ersten Kanal eines Doppelgeräts zu verfahren...

Beachten Sie eine Reihe von Besonderheiten. Näheres hierzu finden Sie in der Knowledge Base.

**iTcMc\_EncoderCoE\_DS402A**

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines Servostellers mit CoE DS402 Profil am EtherCAT Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Multiturn-Absolutencoder ausgerüstet ist. AX8000 Geräte mit Absolutencoder unterstützen dieses Profil.

**i** Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_DriveCoE\_DS402 [▶ 174].

E/A-Variable	Interface.Variable	Verwendung
siehe Hinweis	ST_TcPlcDeviceInput.udiCount	Ermittlung der Ist-Position.
	ST_TcPlcDeviceInput.uiStatus	
	ST_TcPlcDeviceOutput.uiDriveCtrl	
	ST_TcPlcDeviceOutput.NominalVelo	
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Überwachung des Onlinestatus
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Automatische Identifikation.



Die Namen der mit dem Gerät ausgetauschten Prozessdaten werden durch die XML Datei des Herstellers festgelegt.

Eine Liste mit kompatiblen Geräten finden Sie weiter unten.

Mapping Hinweis AX8000:

E/A-Variable	Interface.Variable	Verwendung
Position actual value	ST_TcPlcDeviceInput.udiCount	Ermittlung der Ist-Position.
Statusword	ST_TcPlcDeviceInput.uiStatus	
Controlword	ST_TcPlcDeviceOutput.uiDriveCtrl	
Target velocity	ST_TcPlcDeviceOutput.NominalVelocity	Sollgeschwindigkeit
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Überwachung des Onlinestatus
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Automatische Identifikation.

### iTcMc\_EncoderCoE\_DS402SR

Der Funktionsbaustein übernimmt die Auswertung der Istwerte eines Servostellers mit CoE DS402 Profil am EtherCAT Feldbus. Dabei wird vorausgesetzt, dass der angeschlossene Motor mit einem Resolver oder Singleturn-Absolutencoder ausgerüstet ist.



Der TwinCAT System Manager wird beim manuellen Einfügen oder bei der automatischen Erkennung eines Antriebsstellers vorschlagen, eine NC-Achse im Projekt einzufügen und mit diesem Steller zu verbinden. Soll dieser Steller mit der Hydraulik Bibliothek kontrolliert werden ist dieser Vorschlag unbedingt abzulehnen.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch iTcMc\_DriveCoE\_DS402 [▶ 174].

E/A-Variable	Interface.Variable	Verwendung
siehe Hinweis	ST_TcPlcDeviceInput.udiCount	Ermittlung der Ist-Position.
	ST_TcPlcDeviceInput.uiStatus	
	ST_TcPlcDeviceOutput.uiDriveCtrl	
	ST_TcPlcDeviceOutput.NominalVelocity	
WcState	ST_TcPlcDeviceInput.wDriveWcState	Verbindungsüberwachung.
InfoData.State	ST_TcPlcDeviceInput.uiDriveBoxState	Überwachung des Onlinestatus
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Automatische Identifikation.



Die Namen der mit dem Gerät ausgetauschten Prozessdaten werden durch die XML Datei des Herstellers festgelegt.

Die nachstehenden Index.SubIndex-Kombinationen müssen vom Encoder unterstützt werden.

Index	Subindex	Bedeutung
1000	0	Identifikation
1008	0	Gerätebezeichnung (optional)

Index	Subindex	Bedeutung
1018	1	Hersteller-Kennung
1018	2	Gerätetyp
6080	0	Maximal-Drehzahl in RPM (optional, wenn dieses Objekt nicht unterstützt wird ist die Bezugsgeschwindigkeit manuell einzugeben).
608F	1	Anzahl Encoder-Inkremente pro Motorumdrehung.
6090	1	Für die Stellwertausgabe verwendete Anzahl der Inkremente pro Motorumdrehung.

Die nachstehende Liste kompatibler Geräte ist naturgemäß unvollständig. Sie stellt keine Empfehlung dar sondern ist nur als allgemeiner Hinweis zu verstehen. Der problemlose Einsatz der genannten Geräte kann von Beckhoff nicht garantiert werden. Ist ein Hersteller oder eines seiner Geräte nicht aufgeführt kann ein problemloser Betrieb durchaus gegeben sein, wird jedoch nicht zugesichert.

Hersteller	Typ	Beschreibung
LTi DRIVES GmbH		Servoregler mit Singleturn-Absolutencoder

### iTcMc\_EncoderCoE\_DS406

Der Funktionsbaustein übernimmt die Auswertung von Encodern mit direktem EtherCAT-Anschluss. Der Encoder muss dazu das CiA DS406 Profil unterstützen.

E/A-Variable	Interface.Variable	Verwendung
siehe Hinweis	ST_TcPlcDeviceInput.udiCount	Ermittlung der Ist-Position.
siehe Hinweise	ST_TcPlcDeviceInput.wEncDevState	Überwachung des Gerätestatus.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Überwachung des Onlinestatus.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Automatische Identifikation.



Die Namen der mit dem Gerät ausgetauschten Prozessdaten werden durch die XML Datei des Herstellers festgelegt.



Die Überwachung des Gerätestatus ist nicht für alle Geräte aller Hersteller garantiert. Bei einigen Geräten wird ein 8-Bit Status bereitgestellt. Eine solche Information ist auf den unteren 8 Bit des wEncDevState Elements abzubilden.

Die nachstehenden Index.SubIndex-Kombinationen müssen vom Encoder unterstützt werden.

Index	Subindex	Bedeutung
1000	0	Identifikation
1008	0	Gerätebezeichnung (optional)
1018	1	Hersteller-Kennung
1018	2	Gerätetyp
6001	0	Rotatorische Encoder: Inkremente pro Umdrehung (obligatorisch)

Index	Subindex	Bedeutung
6002	0	Rotatorische Encoder: Gesamter Zählbereich (Option A, alternativ: Index 6502) Lineare Encoder: Gesamter Zählbereich (obligatorisch)
6005	1	Lineare Encoder: Auflösung (Option A, alternativ: Index 6501)
6501	0	Lineare Encoder: Auflösung (Option B, alternativ: Index 6005)
6502	0	Rotatorische Encoder: Anzahl der gezählten Umdrehungen (Option B, alternativ: Index 6002)
650A	2	Lineare Encoder: Untere Grenze des vorgesehenen Arbeitsfeldes (Option)
650B	3	Lineare Encoder: Obere Grenze des vorgesehenen Arbeitsfeldes (Option)

Die nachstehende Liste kompatibler Geräte ist naturgemäß unvollständig. Sie stellt keine Empfehlung dar sondern ist nur als allgemeiner Hinweis zu verstehen. Der problemlose Einsatz der genannten Geräte kann von Beckhoff nicht garantiert werden. Ist ein Hersteller oder eines seiner Geräte nicht aufgeführt kann ein problemloser Betrieb durchaus gegeben sein, wird jedoch nicht zugesichert.

Abhängig von der Unterstützung der aufgelisteten Objekte können bestimmte Parameter automatisch ermittelt werden. Dies gilt für den Zählbereich bzw. die Überlauf-Erkennung und (für lineare Encoder) die Auflösung. Werden die betreffenden Objekte nicht oder nicht in einer unterstützten Kombination bereitgestellt ist dies nicht möglich. In einem solchen Fall kann ein Betrieb möglich sein. Allerdings sind dann die Parameter manuell bei der Inbetriebnahme einzustellen.

Hersteller	Typ	Beschreibung
Fritz Kübler GmbH	58x8	Multiturn-Absolutencoder.
IVO GmbH & Co. KG	GXMMW_H	Multiturn-Absolutencoder.
MTS	Temposonics R	Linear-Absolutencoder.
TR Electronic GmbH	LMP	Linear-Absolutencoder.
TWK-Electronic GmbH	CRKxx12R12C1xx	Multiturn-Absolutencoder.

### iTcMc\_EncoderDigCam

Der Funktionsbaustein übernimmt die Auswertung von vier digitalen Eingängen als Positionsnocken.

E/A-Variable	Interface.Variable	Verwendung
Eingang	ST_TcPlcDeviceInput.bDigCamPP	Ermittlung der Ist-Position: Positive Zielnocke.
Eingang	ST_TcPlcDeviceInput.bDigCamP	Ermittlung der Ist-Position: Positive Bremsnocke.
Eingang	ST_TcPlcDeviceInput.bDigCamM	Ermittlung der Ist-Position: Negative Bremsnocke.
Eingang	ST_TcPlcDeviceInput.bDigCamMM	Ermittlung der Ist-Position: Negative Zielnocke.

### iTcMc\_EncoderDigIncrement

Der Funktionsbaustein übernimmt die Auswertung von zwei digitalen Eingängen für die Emulation einer Inkremental-Encoderauswertung.

E/A-Variable	Interface.Variable	Verwendung
Eingang	ST_TcPlcDeviceInput.bDigInA	Ermittlung der Ist-Position.
Eingang	ST_TcPlcDeviceInput.bDigInB	Ermittlung der Ist-Position.

### iTcMc\_EncoderEL3102

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme EL3102.

E/A-Variable	Interface.Variable	Verwendung
Value	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Optional: Adress-Informationen für Parameter-Kommunikation via CoE.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.

### iTcMc\_EncoderEL3142

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme EL3142. Das Mapping erfolgt wie bei der schnittstellenkompatiblen EL3102.

### iTcMc\_EncoderEL3162

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme EL3162. Das Mapping erfolgt wie bei der schnittstellenkompatiblen EL3102.

### iTcMc\_EncoderEL3255

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme EL3255.

E/A-Variable	Interface.Variable	Verwendung
AI Standard Channel x.Value	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.
AI Standard Channel x.Status	ST_TcPlcDeviceInput.wEncDevState	Auswertung des Fehlersignals des Encoders.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Adress-Informationen für Parameter-Kommunikation via CoE.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.



Die Klemme unterstützt bis zu fünf Encoder. Die Variablen InfoData.AdsAddr, InfoData.State und WcState sind durch Mehrfach-Mapping auf alle beteiligten Achsen zu verteilen.

### iTcMc\_EncoderEL5001

Der Funktionsbaustein übernimmt die Auswertung der Daten einer SSI-Encoderklemme EL5001.

E/A-Variable	Interface.Variable	Verwendung
Value	ST_TcPlcDeviceInput.udiCount	Einlesen der Ist-Position.
Status	ST_TcPlcDeviceOutput.usiRegStatus	Auswertung des Fehlersignals des Encoders.



E/A-Variable	Interface.Variable	Verwendung
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Adress-Informationen für Parameter-Kommunikation via CoE.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.

**iTcMc\_EncoderEL5021**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Sin/Cos-Encoderklemme EL5021.

E/A-Variable	Interface.Variable	Verwendung
ENC Status.Counter value	ST_TcPlcDeviceInput.udiCount	Einlesen der Ist-Position.
ENC Status.Status	ST_TcPlcDeviceInput.usiRegStatus	Auswertung des Fehlersignals des Encoders.
ENC Status.Latch value	ST_TcPlcDeviceInput.udiLatch	Beim Homing unter Verwendung des Synchron-Impulses des Encoders.
ENC Control.Control	ST_TcPlcDeviceOutput.usiCtrl	Steuerung der Latch-Funktion.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Adress-Informationen für Parameter-Kommunikation via CoE.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.

**iTcMc\_EncoderEL5032 (ab V3.0.40)**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer ENDAT-Encoderklemme EL5032.

Die Klemme EL5032 stellt abhängig von ihrer Einstellung einen 32-Bit oder einen 64-Bit Zähler zur Verfügung. Dadurch ist der höchste darstellbare Wert entweder  $2^{32} - 1$  oder  $2^{64} - 1$ . Multipliziert mit der Encoder-Auflösung ergibt dies den auswertbaren Weg. Bei 10 nm Auflösung ergibt sich ein Wert von 42949 mm. Dies ist für die meisten Anwendung ausreichend, weshalb in der Regel der 32-Bit-Modus der Klemme genutzt werden kann. Dafür ist nur das Mapping auf udiCount durchzuführen. Andernfalls muss der 64-Bit-Modus der Klemme aktiviert und das vollständige Mapping auf udiCount und S\_DiReserve[1] konfiguriert werden.

<b>HINWEIS</b>	
<b>Versorgungsspannung beachten</b>	
Um Beschädigungen des angeschlossenen Geräts zu vermeiden ist vor dem Anschließen die in der EL5032 eingestellte Versorgungsspannung zu überprüfen.	

Bei Feldbusstart und beim Rücksetzen eines Achsfehlers werden bestimmte Parameter des angeschlossenen Geräts gelesen. Dabei wird der Typ des Geräts im Logging dokumentiert. Es werden nur absolute Linear-Maßstäbe und absolute Multiturn-Encoder akzeptiert. Bei Linear-Maßstäben wird die Auflösung automatisch in Encoder-Gewichtung und –Interpolation aktualisiert.

E/A-Variable	Interface.Variable	Verwendung
Position (DWORD bzw. unterer Teil des ULINT)	ST_TcPlcDeviceInput.udiCount	Einlesen der Ist-Position.
Position (oberer Teil des ULINT)	ST_TcPlcDeviceInput.S_DiReserve[1]	Optional: Einlesen der Ist-Position unter TC2.
Position (oberer Teil des ULINT)	ST_TcPlcDeviceInput.udiLatch	Optional: Einlesen der Ist-Position unter TC3.

E/A-Variable	Interface.Variable	Verwendung
Status	ST_TcPlcDeviceInput.uiEncDevState	Auswertung des Fehlersignals des Encoders.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Adress-Informationen für Parameter-Kommunikation via CoE.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.

### iTcMc\_EncoderEL5101

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Inkremental-Encoderklemme EL5101.

E/A-Variable	Interface.Variable	Verwendung
Value	ST_TcPlcDeviceInput.uiCount	Betrieb: Einlesen der Ist-Position.
Latch	ST_TcPlcDeviceInput.uiLatch	Beim Homing unter Verwendung des Synchron-Impulses des Encoders.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	Steuerung der Latch-Funktion usw.
Status	ST_TcPlcDeviceInput.usiStatus	Status des Encoders, der Latchfunktion.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Adress-Informationen für Parameter-Kommunikation via CoE.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.

### iTcMc\_EncoderEL5111

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Inkremental-Encoderklemme EL5111.

E/A-Variable	Interface.Variable	Verwendung
Value	ST_TcPlcDeviceInput.uiCount	Betrieb: Einlesen der Ist-Position.
Latch	ST_TcPlcDeviceInput.uiLatch	Beim Homing unter Verwendung des Synchron-Impulses des Encoders.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	Steuerung der Latch-Funktion usw.
Status	ST_TcPlcDeviceInput.usiStatus	Status des Encoders, der Latchfunktion.
InfoData.AdsAddr	ST_TcPlcDeviceInput.sEncAdsAddr	Adress-Informationen für Parameter-Kommunikation via CoE.
InfoData.State	ST_TcPlcDeviceInput.uiEncBoxState	Verbindungsüberwachung, Zustandsüberwachung.
WcState	ST_TcPlcDeviceInput.wEncWcState	Verbindungsüberwachung.

### iTcMc\_EncoderEL7041

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Schrittmotor-Ausgangsklemme EL7041.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_DriveEL7041 \[► 177\]](#).

**iTcMc\_EncoderEL7201**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Servo-Ausgangsklemme EL7201.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_DriveEL7201](#).

**iTcMc\_EncoderIx5009**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer SSI-Encoderbox IP5009.

E/A-Variable	Interface.Variable	Verwendung
PZDL_RegDaten	ST_TcPlcDeviceInput.uiPZDL_RegDaten	Betrieb: Einlesen der Ist-Position. Bei <u>Register-Kommunikation</u> <a href="#">[► 317]</a> : Schnittstelle für gelesene Daten.
PZDH	ST_TcPlcDeviceInput.uiPZDH	Einlesen der Ist-Position.
RegStatus	ST_TcPlcDeviceInput.usiRegStatus	Diverse Status-Informationen.

**iTcMc\_EncoderKL2521**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Pulsausgabeklemme KL2521. Es werden die ausgegebenen Pulse gezählt und für eine Encoder-Emulation benutzt.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_DriveKL2521 \[► 178\]](#).

E/A-Variable	Interface.Variable	Verwendung
Daten Ein	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position. Bei <u>Register-Kommunikation</u> <a href="#">[► 317]</a> : Schnittstelle für gelesene Daten.
Kontrolle	ST_TcPlcDeviceOutput.bTerminalCtrl	Register-Kommunikation
Status	ST_TcPlcDeviceInput.bTerminalState	Register-Kommunikation
Daten Aus	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Register-Kommunikation: Schnittstelle für geschriebene Daten.

**iTcMc\_EncoderKL2531**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Pulsausgabeklemme KL2531. Es werden die ausgegebenen Pulse gezählt und für eine Encoder-Emulation benutzt.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_DriveKL2531 \[► 179\]](#).

E/A-Variable	Interface.Variable	Verwendung
Velocity	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Bei <u>Register-Kommunikation</u> [► 317]: Schnittstelle für geschriebene Daten.
Position	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Ctrl	ST_TcPlcDeviceOutput.bTerminalCtrl	Steuerung der Endstufe, Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Status der Endstufe, Register-Kommunikation.
ExtStatus	ST_TcPlcDeviceInput.uiTerminalState2	Diagnose von Endstufe und Motor

### iTcMc\_EncoderKL2541

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Pulsausgabeklemme KL2541. Es werden die ausgegebenen Pulse gezählt und für eine Encoder-Emulation benutzt.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_DriveKL2541 \[► 180\]](#).

E/A-Variable	Interface.Variable	Verwendung
Velocity	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Bei <u>Register-Kommunikation</u> [► 317]: Schnittstelle für geschriebene Daten.
Position	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Ctrl	ST_TcPlcDeviceOutput.bTerminalCtrl	Steuerung der Endstufe, Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Status der Endstufe, Register-Kommunikation.
ExtCtrl	ST_TcPlcDeviceOutput.uiTerminalCtrl2	Latch-Steuerung bei Homing mit dem Synchron-Impuls des Encoders
ExtStatus	ST_TcPlcDeviceInput.uiTerminalState2	Diagnose von Endstufe und Motor, Latch-Status bei Homing mit dem Synchron-Impuls des Encoders

### iTcMc\_EncoderKL2542

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Motorendstufenklemme KL2542.

Dieses E/A-Gerät gehört zu einer Gruppe von Geräten, die sowohl für die Stellwert-Ausgabe als auch für die Istwert-Ermittlung benutzt werden. Die benötigten Mapping-Definitionen insbesondere zur Parameter-Kommunikation überschneiden sich mit denen des Drive-Bausteins. Siehe hierzu im Vergleich auch [iTcMc\\_DriveKL2542 \[► 180\]](#).

E/A-Variable	Interface.Variable	Verwendung
Daten aus	ST_TcPlcDeviceOutput.nDacOut	Betrieb: Ausgabe des Geschwindigkeits-Signals. Bei <u>Register-Kommunikation [► 317]</u> : Schnittstelle für geschriebene Daten.
Daten ein	ST_TcPlcDeviceInput.uiTerminalData	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Kontrolle	ST_TcPlcDeviceOutput.bTerminalCtrl	Steuerung der Endstufe, Register-Kommunikation.
Status	ST_TcPlcDeviceInput.bTerminalState	Status der Endstufe, Register-Kommunikation.

**iTcMc\_EncoderKL3002**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme KL3002.

E/A-Variable	Interface.Variable	Verwendung
Daten Ein	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	<u>Register-Kommunikation [► 317]</u>
Status	ST_TcPlcDeviceInput.usiStatus	Register-Kommunikation.

**iTcMc\_EncoderKL3042**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme KL3042.

E/A-Variable	Interface.Variable	Verwendung
Daten Ein	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	<u>Register-Kommunikation [► 317]</u>
Status	ST_TcPlcDeviceInput.usiStatus	Register-Kommunikation.

**iTcMc\_EncoderKL3062**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme KL3062.

E/A-Variable	Interface.Variable	Verwendung
Daten Ein	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	<u>Register-Kommunikation [► 317]</u>
Status	ST_TcPlcDeviceInput.usiStatus	Register-Kommunikation.

**iTcMc\_EncoderKL3162**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsklemme KL3162.

E/A-Variable	Interface.Variable	Verwendung
Daten Ein	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	<u>Register-Kommunikation [► 317]</u>
Status	ST_TcPlcDeviceInput.usiStatus	Register-Kommunikation.

**iTcMc\_EncoderKL5001**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer SSI-Encoderklemme KL5001.

E/A-Variable	Interface.Variable	Verwendung
PZDL_RegDaten	ST_TcPlcDeviceInput.uiPZDL_RegDaten	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation [▶ 317]: Schnittstelle für gelesene Daten.
PZDH	ST_TcPlcDeviceInput.uiPZDH	Einlesen der Ist-Position.
RegStatus	ST_TcPlcDeviceInput.usiRegStatus	Diverse Status-Informationen.
RegDaten	ST_TcPlcDeviceOutput.bTerminalData	Register-Kommunikation.

**iTcMc\_EncoderKL5101**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Inkremental-Encoderklemme KL5101.

E/A-Variable	Interface.Variable	Verwendung
Counter	ST_TcPlcDeviceInput.uiCount	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Latch	ST_TcPlcDeviceInput.uiLatch	Beim Homing unter Verwendung des Synchron-Impulses des Encoders.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	Steuerung der Latch-Funktion usw., Register-Kommunikation [▶ 317]
Status	ST_TcPlcDeviceInput.usiStatus	Diverse Status-Informationen.
RegDaten	ST_TcPlcDeviceOutput.bTerminalData	Register-Kommunikation.

**iTcMc\_EncoderKL5111**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Inkremental-Encoderklemme KL5111.

E/A-Variable	Interface.Variable	Verwendung
Counter	ST_TcPlcDeviceInput.uiCount	Betrieb: Einlesen der Ist-Position. Bei Register-Kommunikation: Schnittstelle für gelesene Daten.
Latch	ST_TcPlcDeviceInput.uiLatch	Beim Homing unter Verwendung des Synchron-Impulses des Encoders.
Ctrl	ST_TcPlcDeviceOutput.usiCtrl	Steuerung der Latch-Funktion usw., Register-Kommunikation [▶ 317]
Status	ST_TcPlcDeviceInput.usiStatus	Diverse Status-Informationen.
RegDaten	ST_TcPlcDeviceOutput.bTerminalData	Register-Kommunikation.

**iTcMc\_EncoderLowCostStepper**

Wenn als nDrive\_Type der Wert iTcMc\_DriveLowCostStepper [▶ 181] eingetragen ist werden in ST\_TcPlcDeviceOutput.uiCount die ausgegebenen Halbschritte gezählt. Daraus wird die Istposition errechnet. Ein Mapping ist für den Encoder nicht erforderlich.



Dieser Encoder-Typ ist nur in Kombination mit einem iTcMc\_DriveLowCostStepper Antrieb nutzbar.

**iTcMc\_EncoderM2510**

Der Funktionsbaustein übernimmt die Auswertung der Daten einer Analog-Eingangsbox M2510.

E/A-Variable	Interface.Variable	Verwendung
Daten Ein	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.

**iTcMc\_EncoderM3120**

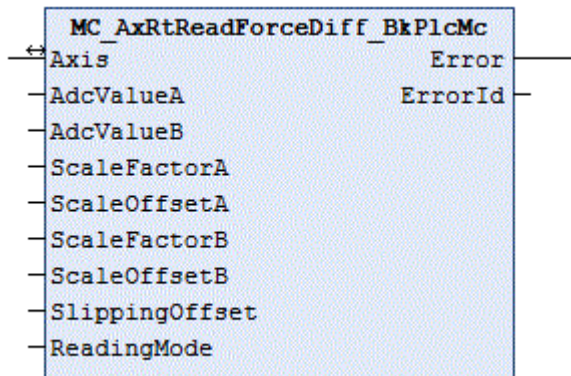
Der Funktionsbaustein übernimmt die Auswertung der Daten einer Inkremental-Encoderbox M3120.

E/A-Variable	Interface.Variable	Verwendung
Value_N	ST_TcPlcDeviceInput.uiCount	Einlesen der Ist-Position.
State_N	ST_TcPlcDeviceInput.usiStatus	Diverse Status-Informationen.
Ctrl_N	ST_TcPlcDeviceOutput.usiCtrl	Steuerung der Latch-Funktion usw.

**iTcMc\_EncoderSim**

Ein Simulationsencoder errechnet die Istposition durch Integration der Sollgeschwindigkeit. Es ist kein Mapping erforderlich.

**4.4.3.2 MC\_AxRtReadForceDiff\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein übernimmt die Ermittlung der Istkraft der Achse aus den Eingangsdaten von zwei Analog-Eingangsklemmen. Dabei wird der Istdruck der A- und B-Seite unter Berücksichtigung der Flächen und der Gleitreibung in die an der Last wirksam werdende Kraft umgerechnet.

Steht nur ein Eingangssignal zur Verfügung sollte ein Baustein des Typs MC\_AxRtReadForceSingle\_BkPlcMc [► 202] verwendet werden. Soll der Istdruck ermittelt werden ist ein Baustein vom Typ MC\_AxRtReadPressureDiff\_BkPlcMc [► 204] einzusetzen.

**🔌 Eingänge**

```

VAR_INPUT
  AdcValueA:      INT:=0;
  AdcValueB:      INT:=0;
  ScaleFactorA:   LREAL:=0.0;
  ScaleOffsetA:   LREAL:=0.0;
  ScaleFactorB:   LREAL:=0.0;
  ScaleOffsetB:   LREAL:=0.0;
  SlippingOffset: LREAL:=0.0;
  ReadingMode:    E_TcMcPressureReadingMode:=iTcHydPressureReadingDefault;
END_VAR
    
```

## E\_TcMcPressureReadingMode [▶ 111]

Name	Typ	Beschreibung
AdcValueA	INT	Hier sind die Eingangsdaten von den Analog-Klemmen zu übergeben.
AdcValueB	INT	
ScaleFactorA	LREAL	[N/ADC_INC] Dieser Wert stellt die Gewichtung dar. Er legt fest, welche Druckerhöhung einer Stufe des AD-Wandlers entspricht.
ScaleOffsetA	LREAL	[N/ADC_INC] Dieser Offset dient dazu, den Nullpunkt der Druckskala zu korrigieren.
ScaleFactorB	LREAL	[N/ADC_INC] Dieser Wert stellt die Gewichtung dar. Er legt fest, welche Druckerhöhung einer Stufe des AD-Wandlers entspricht.
ScaleOffsetB	LREAL	[N/ADC_INC] Dieser Offset dient dazu, den Nullpunkt der Druckskala zu korrigieren.
SlippingOffset	LREAL	[N] Wenn der Baustein für die Berechnung der wirksam werdenden Kraft genutzt wird kann hier die für die Überwindung der Gleitreibung benötigte Kraft eingetragen werden.
ReadingMode	E_TcMcPressureReadingMode	Hier kann die zu ermittelnde Istgröße festgelegt werden. Als Default-Ziel wird <a href="#">Axis_Ref_BkPlcMc [▶ 84].ST_TcHydAxRtData [▶ 131].fActPressure</a> ausgewählt.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [▶ 84]</a> zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Error:      BOOL;
  ErrorID:    UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Der Baustein ermittelt durch Auswertung der Variablen **AdcValueA** und **AdcValueB** den Istdruck bzw. die Istkraft der Achse. Das Ergebnis wird in [ST\\_TcHydAxRtData \[▶ 131\].fActPressure](#) eingetragen.



Die Speicherung von einer Achse zugeordneten Parametern kann zum Beispiel in den [ST\\_TcHydAxParam.fCustomerData\[...\]](#) erfolgen. So ist sichergestellt, dass die Daten zusammen mit den Standard-Parametern der Achse geladen, gespeichert und gesichert und bei Bedarf auch exportiert und importiert werden.

**Ermittlung eines Differenz-Istdrucks**

Für die Inbetriebnahme sind drei Vorgehensweisen üblich.



### Inbetriebnahme-Variante A (bevorzugt für $\pm 10V$ )

Hierbei ist keine Bewegung der Achse erforderlich. Die erreichbare Genauigkeit ist bei hochwertigen Drucksensoren in den meisten Fällen ausreichend.

- Als **ScaleFactorA** und **ScaleFactorB** ist der Bemessungsdruck der Drucksensoren dividiert durch **AdcValueA<sub>MAX</sub>** bzw. **AdcValueB<sub>MAX</sub>** einzutragen.
- Wird der Baustein zur Ermittlung des Istdrucks verwendet sind die Parameter **ScaleArreaA** und **ScaleArreaB** auf den Wert 1.0 einzustellen. Andernfalls sind diese Parameter für eine Istkraft in N (= Newton) in mm<sup>2</sup> anzugeben.

### Inbetriebnahme-Variante B

Hierbei ist es erforderlich, das in beide Richtungen ein Block mit vollem Systemdruck angefahren werden kann. Dabei ist eine echte Bewegung der Achse nicht erforderlich. Also kann das Anfahren der Anschläge dadurch nachgebildet werden, dass die Achse durch Einlegen von provisorischen Begrenzungen nur einen Teil ihres Fahrweges zurücklegen kann oder sogar mechanisch vollständig fixiert wird.

- Es sind alle Bausteine zu deaktivieren, die auf den Wert von **ST\_TcHydAxRtData [► 131].fActPressure** reagieren.
- Mit niedriger Geschwindigkeit wird zunächst der untere (in Richtung sinkender Istposition gelegene) Block angefahren. Dabei wird für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Jetzt sollte auf der A-Seite der Systemdruck und auf der B-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall sind die A- und B-seitigen Drücke durch Messung zu ermitteln.
- Mit niedriger Geschwindigkeit wird anschließend der obere (in Richtung steigender Istposition gelegene) Block angefahren. Dabei wird erneut für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Auch jetzt sind die Drücke festzustellen.
- Die einzutragenden Parameter errechnen sich dann wie folgt:

$$\mathbf{ScaleFactorA} := (\text{DruckA}_{\text{MAX}} - \text{DruckA}_{\text{MIN}}) / (\mathbf{AdcValueA}_{\text{MAX}} - \mathbf{AdcValueA}_{\text{MIN}});$$

$$\mathbf{ScaleFactorB} := (\text{DruckB}_{\text{MAX}} - \text{DruckB}_{\text{MIN}}) / (\mathbf{AdcValueB}_{\text{MAX}} - \mathbf{AdcValueB}_{\text{MIN}});$$

$$\mathbf{ScaleOffsetA} := \text{DruckA}_{\text{MIN}} - \mathbf{ScaleFactorA} * \mathbf{AdcValueA}_{\text{MIN}}$$

$$\mathbf{ScaleOffsetB} := \text{DruckB}_{\text{MIN}} - \mathbf{ScaleFactorB} * \mathbf{AdcValueB}_{\text{MIN}}$$

### Inbetriebnahme-Variante C

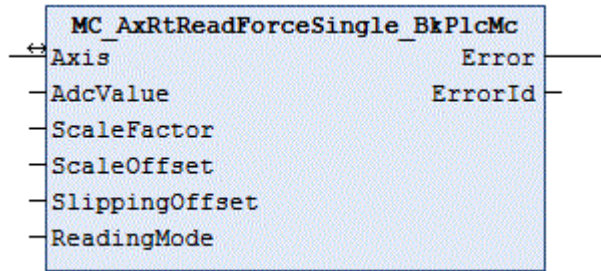
Ersatzweise kann eine Inbetriebnahme der Erfassung ohne Ansteuerung der Achse erfolgen. Dabei wird jedoch eine weitaus geringere Genauigkeit erreicht.

- Zunächst ist die Achse druckfrei zu machen. Dazu ist der Kompressor still zu setzen und der Druckspeicher zu entlasten.
- Es ist sicher zu stellen, dass die Achse keinen Druck aufbaut. Dazu ist eine von außen mit Kräften (Gravitation usw.) beaufschlagte Achse mechanisch abzustützen. Das Ventil ist mehrfach manuell oder elektrisch in beide Richtungen zu öffnen.
- Jetzt wird für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Es sollte sowohl auf der A- als auch auf der B-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall sind die A- und B-seitigen Drücke durch Messung zu ermitteln. Die so gefundenen Werte sind in den oben genannten Gleichungen als **MIN** Werte einzusetzen.
- Aus den Datenblattangaben der Drucksensoren ist der Druck für die obere Grenze des elektrischen Signals (10 Volt, 20 mA) zu entnehmen. Als **AdcValueA** und **AdcValueB** ist der obere Grenzwert für die gewandelte elektrische Größe anzunehmen. Diese Werte sind als **MAX** Werte in die oben genannten Gleichungen einzusetzen.
- Die einzutragenden Parameter errechnen sich dann wie oben beschrieben.

### Ermittlung einer wirksam werden Kraft

Für die Ermittlung einer wirksam werdenden Kraft ist zunächst wie oben beschrieben der Istdruck zu ermitteln. Durch Eintragen der wirksamen Flächen unter **ScaleArreaA** und **ScaleArreaB** wird der Baustein veranlasst, die beidseitigen Drücke unter Berücksichtigung der Flächen in Kräfte umzurechnen.

### 4.4.3.3 MC\_AxRtReadForceSingle\_BkPlcMc (ab V3.0)



Der Funktionsbaustein übernimmt die Ermittlung der Istkraft der Achse aus den Eingangsdaten einer Analog-Eingangsklemme. Dabei wird der Istdruck der A- oder B-Seite unter Berücksichtigung der Fläche und der Gleitreibung in die an der Last wirksam werdende Kraft umgerechnet.

**i** Steht nur ein Eingangssignal zur Verfügung sollte ein Baustein des Typs [MC\\_AxRtReadForceDiff\\_BkPlcMc \[► 199\]](#) verwendet werden. Soll der Istdruck ermittelt werden ist ein Baustein vom Typ [MC\\_AxRtReadPressureDiff\\_BkPlcMc \[► 204\]](#) einzusetzen.

#### Eingänge

```
VAR_INPUT
  AdcValueA:      INT:=0;
  AdcValueB:      INT:=0;
  ScaleFactorA:   LREAL:=0.0;
  ScaleOffsetA:   LREAL:=0.0;
  ScaleFactorB:   LREAL:=0.0;
  ScaleOffsetB:   LREAL:=0.0;
  SlippingOffset: LREAL:=0.0;
  ReadingMode:    E_TcMcPressureReadingMode:=iTcHydPressureReadingDefault;
END_VAR
```

Name	Typ	Beschreibung
AdcValueA	INT	Hier sind die Eingangsdaten von den Analog-Klemmen zu übergeben.
AdcValueB	INT	
ScaleFactorA	LREAL	[N/ADC_INC] Dieser Wert stellt die Gewichtung dar. Er legt fest, welche Druckerhöhung einer Stufe des AD-Wandlers entspricht.
ScaleOffsetA	LREAL	[N/ADC_INC] Dieser Offset dient dazu, den Nullpunkt der Druckskala zu korrigieren.
ScaleFactorB	LREAL	[N/ADC_INC] Dieser Wert stellt die Gewichtung dar. Er legt fest, welche Druckerhöhung einer Stufe des AD-Wandlers entspricht.
ScaleOffsetB	LREAL	[N/ADC_INC] Dieser Offset dient dazu, den Nullpunkt der Druckskala zu korrigieren.
SlippingOffset	LREAL	[N] Wenn der Baustein für die Berechnung der wirksam werdenden Kraft genutzt wird, kann hier die für die Überwindung der Gleitreibung benötigte Kraft eingetragen werden.
ReadingMode	E_TcMcPressureReadingMode	Hier kann die zu ermittelnde Istgröße festgelegt werden. Als Defaultziel wird <a href="#">Axis_Ref_BkPlcMc [► 84].ST_TcHydAxRtData [► 131].fActPressure</a> ausgewählt.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Error: Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	ErrorID: Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Der Baustein ermittelt durch Auswertung der Variablen **AdcValueA** den Istdruck bzw. die Istkraft der Achse. Das Ergebnis wird in ST\_TcHydAxRtData [► 131].fActPressure eingetragen.

Die Speicherung von einer Achse zugeordneten Parametern kann zum Beispiel in den ST\_TcHydAxParam [► 120].fCustomerData[...] erfolgen. So ist sicher gestellt, dass die Daten zusammen mit den Standard-Parametern der Achse geladen, gespeichert und gesichert und bei Bedarf auch exportiert und importiert werden.

**Ermittlung eines Differenz-Istdrucks**

Wird der Baustein zur Ermittlung des Istdrucks benutzt sind die Parameter **ScaleArreaA** und **ScaleArreaB** auf den Wert 1.0 und **SlippingOffset** auf den Wert 0.0 einzustellen.

**Inbetriebnahme-Variante A**

Hierbei ist keine Bewegung der Achse erforderlich. Die erreichbare Genauigkeit ist bei hochwertigen Drucksensoren in den meisten Fällen ausreichend.

- Als **ScaleFactorA** ist der Bemessungsdruck der Drucksensoren dividiert durch **AdcValueA<sub>MAX</sub>** einzutragen.

**Inbetriebnahme-Variante B**

Hierbei ist es erforderlich, das in beide Richtungen ein Block mit vollem Systemdruck angefahren werden kann. Dabei ist eine echte Bewegung der Achse nicht erforderlich. Also kann das Anfahren der Anschläge dadurch nachgebildet werden, dass die Achse durch Einlegen von provisorischen Begrenzungen nur einen Teil ihres Fahrweges zurücklegen kann oder sogar mechanisch vollständig fixiert wird.

- Es sind alle Bausteine zu deaktivieren, die auf den Wert von ST\_TcHydAxRtData [► 131].fActPressure reagieren.
- Mit niedriger Geschwindigkeit wird zunächst der untere (in Richtung sinkender Istposition gelegene) Block angefahren. Dabei wird für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Jetzt sollte auf der A-Seite der Systemdruck und auf der B-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall sind die A- und B-seitigen Drücke durch Messung zu ermitteln.
- Mit niedriger Geschwindigkeit wird anschließend der obere (in Richtung steigender Istposition gelegene) Block angefahren. Dabei wird erneut für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Auch jetzt sind die Drücke festzustellen.
- Die einzutragenden Parameter errechnen sich dann wie folgt:

$$\text{ScaleFactorA} := (\text{DruckA}_{\text{MAX}} - \text{DruckA}_{\text{MIN}}) / (\text{AdcValueA}_{\text{MAX}} - \text{AdcValueA}_{\text{MIN}});$$

$$\text{ScaleFactorB} := (\text{DruckB}_{\text{MAX}} - \text{DruckB}_{\text{MIN}}) / (\text{AdcValueB}_{\text{MAX}} - \text{AdcValueB}_{\text{MIN}});$$

$$\text{ScaleOffsetA} := \text{DruckA}_{\text{MIN}} - \text{ScaleFactorA} * \text{AdcValueA}_{\text{MIN}}$$

$$\text{ScaleOffsetB} := \text{DruckB}_{\text{MIN}} - \text{ScaleFactorB} * \text{AdcValueB}_{\text{MIN}}$$

## Inbetriebnahme-Variante C

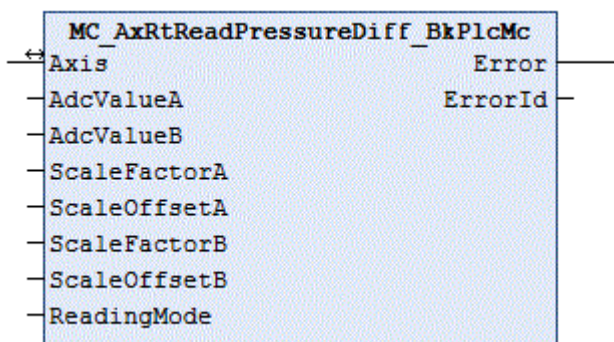
Ersatzweise kann eine Inbetriebnahme der Erfassung ohne Ansteuerung der Achse erfolgen. Dabei wird jedoch eine weitaus geringere Genauigkeit erreicht.

- Zunächst ist die Achse druckfrei zu machen. Dazu ist der Kompressor still zu setzen und der Druckspeicher zu entlasten.
- Es ist sicher zu stellen, dass die Achse keinen Druck aufbaut. Dazu ist eine von außen mit Kräften (Gravitation usw.) beaufschlagte Achse mechanisch abzustützen. Das Ventil ist mehrfach manuell oder elektrisch in beide Richtungen zu öffnen.
- Jetzt wird für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Es sollte sowohl auf der A- als auch auf der B-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall sind die A- und B-seitigen Drücke durch Messung zu ermitteln. Die so gefundenen Werte sind in den oben genannten Gleichungen als **MIN** Werte einzusetzen.
- Aus den Datenblattangaben der Drucksensoren ist der Druck für die obere Grenze des elektrischen Signals (10 Volt, 20 mA) zu entnehmen. Als **AdcValueA** und **AdcValueB** ist der obere Grenzwert für die gewandelte elektrische Größe anzunehmen. Diese Werte sind als **MAX** Werte in die oben genannten Gleichungen einzusetzen.
- Die einzutragenden Parameter errechnen sich dann wie oben beschrieben.

### Ermittlung einer wirksam werden Kraft

Für die Ermittlung einer wirksam werdenden Kraft ist zunächst wie oben beschrieben der Istdruck zu ermitteln. Durch Eintragen der wirksamen Fläche unter **ScaleArreaA** wird der Baustein veranlasst, den einseitigen Druck unter Berücksichtigung der Fläche in eine Kraft umzurechnen.

#### 4.4.3.4 MC\_AxRtReadPressureDiff\_BkPlcMc (ab V3.0)



Der Funktionsbaustein übernimmt die Ermittlung des Istdrucks der Achse aus den Eingangsdaten von zwei Analog-Eingangsklemmen.



Steht nur ein Eingangssignal zur Verfügung sollte ein Baustein des Typs [MC\\_AxRtReadPressureSingle\\_BkPlcMc \[► 207\]](#) verwendet werden. Soll nicht der Druck sondern die Kraft ermittelt werden ist ein Baustein vom Typ [MC\\_AxRtReadForceDiff\\_BkPlcMc \[► 199\]](#) einzusetzen.

### Eingänge

```
VAR_INPUT
  AdcValueA:    INT:=0;
  AdcValueB:    INT:=0;
  ScaleFactorA: LREAL:=0.0;
  ScaleOffsetA: LREAL:=0.0;
  ScaleFactorB: LREAL:=0.0;
  ScaleOffsetB: LREAL:=0.0;
  ReadingMode:  E_TcMcPressureReadingMode:=iTcHydPressureReadingDefault;
END_VAR
```

Name	Typ	Beschreibung
AdcValueA	INT	Hier sind die Eingangsdaten von den Analog-Klemmen zu übergeben.
AdcValueB	INT	
ScaleFactorA	LREAL	[bar/ADC_INC] Dieser Wert stellt die Gewichtung dar. Er legt fest, welche Druckerhöhung einer Stufe des AD-Wandlers entspricht.
ScaleOffsetA	LREAL	[bar] Dieser Offset dient dazu, den Nullpunkt der Druckskala zu korrigieren.
ScaleFactorB	LREAL	[bar/ADC_INC] Dieser Wert stellt die Gewichtung dar. Er legt fest, welche Druckerhöhung einer Stufe des AD-Wandlers entspricht.
ScaleOffsetB	LREAL	[bar] Dieser Offset dient dazu, den Nullpunkt der Druckskala zu korrigieren.
ReadingMode	E_TcMcPressureReadingMode	Mit diesem Parameter wird festgelegt, wo das Ergebnis der Auswertung abzulegen ist.

 Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 Ausgänge

```
VAR_OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei kann ein Problem erkannt und gemeldet werden:

- Ist der Pointer pStAxRtData in Axis\_Ref\_BkPlcMc [► 84] nicht initialisiert reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdPtrMcPlc**. Die Achse kann in diesem Fall nicht in einen Störzustand versetzt werden.

Wenn diese Überprüfung ohne Problem durchgeführt werden konnte wird der Istdruck der Achse durch Auswertung der Variablen **AdcValueA** und **AdcValueB** ermittelt. Das Ergebnis wird in ST\_TcHydAxRtData [► 131].fActPressure eingetragen.

Die Speicherung von einer Achse zugeordneten Parametern kann zum Beispiel in den ST\_TcHydAxParam [► 120].fCustomerData[...] erfolgen. So ist sichergestellt, dass die Daten zusammen mit den Standard-Parametern der Achse geladen, gespeichert und gesichert und bei Bedarf auch exportiert und importiert werden.

**Inbetriebnahme-Variante A**

Hierbei ist keine Bewegung der Achse erforderlich. Die erreichbare Genauigkeit ist bei hochwertigen Drucksensoren in den meisten Fällen ausreichend.

- Als **ScaleFactorA** und **ScaleFactorB** ist der Bemessungsdruck der Drucksensoren dividiert durch **AdcValueA<sub>MAX</sub>** bzw. **AdcValueB<sub>MAX</sub>** einzutragen.

### Inbetriebnahme-Variante B

Hierbei ist keine Bewegung der Achse erforderlich. Die erreichbare Genauigkeit ist bei hochwertigen Drucksensoren in den meisten Fällen ausreichend.

- Als **ScaleFactorA** und **ScaleFactorB** ist der Bemessungsdruck der Drucksensoren dividiert durch **AdcValueA<sub>MAX</sub>** bzw. **AdcValueB<sub>MAX</sub>** einzutragen.

### Inbetriebnahme-Variante C

Hierbei ist es erforderlich, das in beide Richtungen ein Block mit vollem Systemdruck angefahren werden kann. Dabei ist eine echte Bewegung der Achse nicht erforderlich. Also kann das Anfahren der Anschläge dadurch nachgebildet werden, dass die Achse durch Einlegen von provisorischen Begrenzungen nur einen Teil ihres Fahrweges zurücklegen kann oder sogar mechanisch vollständig fixiert wird.

- Es sind alle Bausteine zu deaktivieren, die auf den Wert von **ST\_TcHydAxRtData [► 131].fActPressure** reagieren.
- Mit niedriger Geschwindigkeit wird zunächst der untere (in Richtung sinkender Istposition gelegene) Block angefahren. Dabei wird für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Jetzt sollte auf der B-Seite der Systemdruck und auf der A-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall sind die A- und B-seitigen Drücke durch Messung zu ermitteln.
- Mit niedriger Geschwindigkeit wird anschließend der obere (in Richtung steigender Istposition gelegene) Block angefahren. Dabei wird erneut für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Auch jetzt sind die Drücke festzustellen.
- Die einzutragenden Parameter errechnen sich dann wie folgt:

$$\mathbf{ScaleFactorA} := (\text{DruckA}_{\text{MAX}} - \text{DruckA}_{\text{MIN}}) / (\mathbf{AdcValueA}_{\text{MAX}} - \mathbf{AdcValueA}_{\text{MIN}});$$

$$\mathbf{ScaleFactorB} := (\text{DruckB}_{\text{MAX}} - \text{DruckB}_{\text{MIN}}) / (\mathbf{AdcValueB}_{\text{MAX}} - \mathbf{AdcValueB}_{\text{MIN}});$$

$$\mathbf{ScaleOffsetA} := \text{DruckA}_{\text{MIN}} - \mathbf{ScaleFactorA} * \mathbf{AdcValueA}_{\text{MIN}}$$

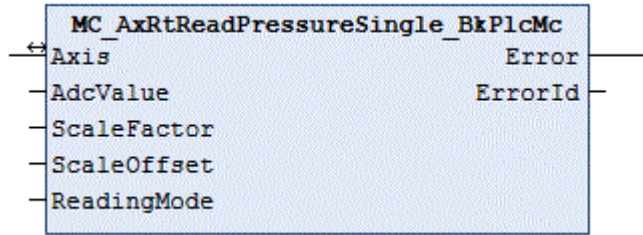
$$\mathbf{ScaleOffsetB} := \text{DruckB}_{\text{MIN}} - \mathbf{ScaleFactorB} * \mathbf{AdcValueB}_{\text{MIN}}$$

### Inbetriebnahme-Variante D

Ersatzweise kann eine Inbetriebnahme der Erfassung ohne Ansteuerung der Achse erfolgen. Dabei wird jedoch eine weitaus geringere Genauigkeit erreicht.

- Zunächst ist die Achse druckfrei zu machen. Dazu ist der Kompressor still zu setzen und der Druckspeicher zu entlasten.
- Es ist sicher zu stellen, dass die Achse keinen Druck aufbaut. Dazu ist eine von außen mit Kräften (Gravitation usw.) beaufschlagte Achse mechanisch abzustützen. Das Ventil ist mehrfach manuell oder elektrisch in beide Richtungen zu öffnen.
- Jetzt wird für **AdcValueA** und **AdcValueB** der Wert ermittelt und festgehalten. Es sollte sowohl auf der A- als auch auf der B-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall sind die A- und B-seitigen Drücke durch Messung zu ermitteln. Die so gefundenen Werte sind in den oben genannten Gleichungen als **MIN** Werte einzusetzen.
- Aus den Datenblattangaben der Drucksensoren ist der Druck für die obere Grenze des elektrischen Signals (10 Volt, 20 mA) zu entnehmen. Als **AdcValueA** und **AdcValueB** ist der obere Grenzwert für die gewandelte elektrische Größe anzunehmen. Diese Werte sind als **MAX** Werte in die oben genannten Gleichungen einzusetzen.
- Die einzutragenden Parameter errechnen sich dann wie oben beschrieben.

### 4.4.3.5 MC\_AxRtReadPressureSingle\_BkPlcMc (ab V3.0)



Der Funktionsbaustein übernimmt die Ermittlung des Istdrucks der Achse aus den Eingangsdaten einer Analog-Eingangsklemme.



Stehen zwei Eingangssignale für die A- und B-Seite zur Verfügung sollte ein Baustein des Typs [MC\\_AxRtReadPressureDiff\\_BkPlcMc](#) [[▶ 204](#)] verwendet werden.

#### Eingänge

```
VAR_INPUT
  AdcValue:      INT:=0;
  ScaleFactor:   LREAL:=0.0;
  ScaleOffset:   LREAL:=0.0;
  ReadingMode:   E_TcMcPressureReadingMode:=iTcHydPressureReadingDefault;
END_VAR
```

Name	Typ	Beschreibung
AdcValue	INT	Hier sind die Eingangsdaten von der Analog-Klemme zu übergeben.
ScaleFactor	LREAL	[bar/ADC_INC] Dieser Wert stellt die Gewichtung dar. Er legt fest, welche Druckerhöhung einer Stufe des AD-Wandlers entspricht.
ScaleOffset	LREAL	[bar] Dieser Offset dient dazu, den Nullpunkt der Druckskala zu korrigieren.
ReadingMode	E_TcMcPressureReadingMode	Hier kann die zu ermittelnde Istgröße festgelegt werden. Als Defaultwert wird <a href="#">Axis_Ref_BkPlcMc</a> [ <a href="#">▶ 84</a> ]. <a href="#">ST_TcHydAxRtData</a> [ <a href="#">▶ 131</a> ].fActPressure ausgewählt.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc</a> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

## Verhalten des Bausteins

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei kann ein Problem erkannt und gemeldet werden:

- Ist der Pointer `pStAxRtData` in [Axis\\_Ref\\_BkPlcMc \[► 84\]](#) nicht initialisiert reagiert der Baustein mit **Error** und **ErrorID:=dwTcHydErrCdPtrMcPlc**. Die Achse kann in diesem Fall nicht in einen Stöorzustand versetzt werden.

Wenn diese Überprüfung ohne Problem durchgeführt werden konnte wird der Istdruck der Achse durch Auswertung der Variablen **AdcValue** ermittelt. Das Ergebnis wird in [ST\\_TcHydAxRtData \[► 131\].fActPressure](#) eingetragen.



Die Speicherung von einer Achse zugeordneten Parametern kann zum Beispiel in den [ST\\_TcHydAxParam \[► 120\].fCustomerData\[...\]](#) erfolgen. So ist sichergestellt, dass die Daten zusammen mit den Standard-Parametern der Achse geladen, gespeichert und gesichert und bei Bedarf auch exportiert und importiert werden.

## Inbetriebnahme-Variante A

Hierbei ist es erforderlich, das in beide Richtungen ein Block mit vollem Systemdruck angefahren werden kann. Dabei ist eine echte Bewegung der Achse nicht erforderlich. Also kann das Anfahren der Anschläge dadurch nachgebildet werden, dass die Achse durch Einlegen von provisorischen Begrenzungen nur einen Teil ihres Fahrweges zurücklegen kann oder sogar mechanisch vollständig fixiert wird.

- Es sind alle Bausteine zu deaktivieren, die auf den Wert von [ST\\_TcHydAxRtData \[► 131\].fActPressure](#) reagieren.
- Mit niedriger Geschwindigkeit wird zunächst der untere (in Richtung sinkender Istposition gelegene) Block angefahren. Dabei wird für **AdcValue** der Wert zu ermittelt und festgehalten. Jetzt sollte auf der B-Seite der Systemdruck und auf der A-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall sind die A- und B-seitigen Drücke durch Messung zu ermitteln.
- Mit niedriger Geschwindigkeit wird anschließend der obere (in Richtung steigender Istposition gelegene) Block angefahren. Dabei wird erneut für **AdcValue** der Wert ermittelt und festgehalten. Auch jetzt sind die Drücke festzustellen.
- Die einzutragenden Parameter errechnen sich dann wie folgt:

$$\text{ScaleFactor} := (\text{Druck}_{\text{MAX}} - \text{Druck}_{\text{MIN}}) / (\text{AdcValue}_{\text{MAX}} - \text{AdcValue}_{\text{MIN}});$$

$$\text{ScaleOffset} := \text{Druck}_{\text{MIN}} - \text{ScaleFactor} * \text{AdcValue}_{\text{MIN}}$$

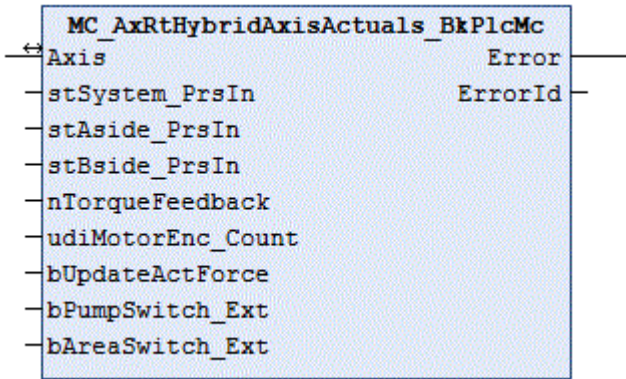
## Inbetriebnahme-Variante B

Ersatzweise kann eine Inbetriebnahme der Erfassung ohne Ansteuerung der Achse erfolgen. Dabei wird jedoch eine weitaus geringere Genauigkeit erreicht.

- Zunächst ist die Achse druckfrei zu machen. Dazu ist der Kompressor still zu setzen und der Druckspeicher zu entlasten.
- Es ist sicher zu stellen, dass die Achse keinen Druck aufbaut. Dazu ist eine von außen mit Kräften (Gravitation usw.) beaufschlagte Achse mechanisch abzustützen. Das Ventil ist mehrfach manuell oder elektrisch in beide Richtungen zu öffnen.
- Jetzt wird für **AdcValue** der Wert ermittelt und festgehalten. Es sollte sowohl auf der A- als auch auf der B-Seite der Tankdruck und somit der Umgebungsdruck anstehen. Ist dies aus irgendwelchen Gründen nicht der Fall ist der A-seitige Druck durch Messung zu ermitteln. Die so gefundenen Werte sind in den oben genannten Gleichungen als **MIN** Werte einzusetzen.
- Aus den Datenblattangaben der Drucksensoren ist der Druck für die obere Grenze des elektrischen Signals (10 Volt, 20 mA) zu entnehmen. Als **AdcValue** ist der obere Grenzwert für die gewandelte elektrische Größe anzunehmen. Diese Werte sind als **MAX** Werte in die oben genannten Gleichungen einzusetzen.
- Die einzutragenden Parameter errechnen sich dann wie oben beschrieben.



4.4.3.6 MC\_AxRtHybridAxisActuals\_BkPlcMc (ab V3.0.44)



Der Funktionsbaustein übernimmt die Ermittlung der wesentlichen Istwerte einer servoelektrisch-hydraulischen Hybrid-Achse.

**Eingänge**

```
VAR_INPUT
  stSystem_PrsIn:   ST_TcPlcInputAnalog [▶ 148];
  stAside_PrsIn:   ST_TcPlcInputAnalog [▶ 148];
  stBside_PrsIn:   ST_TcPlcInputAnalog [▶ 148];

  nTorqueFeedback: INT;
  udiMotorEnc_Count: UDINT;

  bUpdateActForce: ST_BOOL;

  bPumpSwitch_Ext:  BOOL;
  bAreaSwitch_Ext:  BOOL;
```

Name	Typ	Beschreibung
stSystem_PrsIn	ST_TcPlcInputAnalog [▶ 148]	Wenn am vorgespannten Tank eine Druck-Sensorik vorhanden ist, werden hier die Input-Variablen der Klemme übergeben.
stAside_PrsIn	ST_TcPlcInputAnalog [▶ 148]	Wenn an der positiv wirkenden Fläche des Zylinders eine Druck-Sensorik vorhanden ist, werden hier die Input-Variablen der Klemme übergeben.
stBside_PrsIn	ST_TcPlcInputAnalog [▶ 148]	Wenn an der negativ wirkenden Fläche des Zylinders eine Druck-Sensorik vorhanden ist, werden hier die Input-Variablen der Klemme übergeben.
nTorqueFeedback	INT	Hier ist die Drehmoment-Rückmeldung des Antriebs zu übergeben.
udiMotorEnc_Count	UDINT	Hier ist der Zählwert des Motor-Encoders zu übergeben.
bUpdateActForce	ST_BOOL	Bei diesem Signal errechnet der Baustein die aktuelle Istkraft der Achse und aktualisiert sie in stAxRtData.fActForce.
bPumpSwitch_Ext	BOOL	Dieses Signal teilt dem Baustein mit, dass eine Pumpenumschaltung der Achse aktiv ist.
bAreaSwitch_Ext	BOOL	Dieses Signal teilt dem Baustein mit, dass eine Flächenumschaltung der Achse aktiv ist.

**Ein-/Ausgänge**

```
VAR_IN_OUT
Axis:  AXIS_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	AXIS_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>AxisRef BkPlcMc</code> [► 84] zu übergeben.

## Ausgänge

```
VAR_OUTPUT
  Error:    BOOL;
  ErrorId:  BOOL;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorId	BOOL	Hier wird eine codierte Fehlerursache bereitgestellt.

## Verhalten des Bausteins

Wenn die Achse als ‚hybrid‘ gekennzeichnet ist führt der Baustein folgende Schritte aus.

- Wenn für die Achse ein zulässiger Encodertyp für den Motor-Encoder angegeben ist ermittelt der Baustein aus **udiMotorEnc\_Count** den aktuellen Istwinkel. Andernfalls wird die Achse in einen Stöorzustand versetzt und der Winkel auf 0.0° gesetzt.
- Mit der eingestellten Anzahl der Pumpen-Kavitäten wird der Modulo-Istwinkel aktualisiert.
- Die Pumpen-Istgeschwindigkeit wird ermittelt.
- Das Pumpendrehmoment wird ermittelt.
- Wenn für den Systemdruck ein Skalierungsdruck eingestellt ist und die Verbindungsüberwachung in **stSystem\_Prsln** kein Problem zeigt wird der Eingangswert in einen Druck umgerechnet und in `stAxRtData.fSupplyPressure` der Achse aktualisiert.
- Wenn für den Druck auf der positiv wirkenden Seite ein Skalierungsdruck eingestellt ist und die Verbindungsüberwachung in **stAside\_Prsln** kein Problem zeigt wird der Eingangswert in einen Druck umgerechnet und in `stAxRtData.fActPressureA` der Achse aktualisiert.
- Wenn für den Druck auf der negativ wirkenden Seite ein Skalierungsdruck eingestellt ist und die Verbindungsüberwachung in **stBside\_Prsln** kein Problem zeigt wird der Eingangswert in einen Druck umgerechnet und in `stAxRtData.fActPressureB` der Achse aktualisiert.
- Wenn in **bUpdateActForce** ein TRUE übergeben wird aktualisiert der Baustein `stAxRtData.fActForce` der Achse. Dabei verwendet er die aktuell wirksamen Flächen.
- Wenn eine Flanke an einem der Umschaltsignale (**bAreaSwitch\_Ext**, **bPumpSwitch\_Ext**) erkannt wird initiiert der Baustein eine Rampe für die Änderung der Vorschubkonstante und der maximalen Geschwindigkeit.

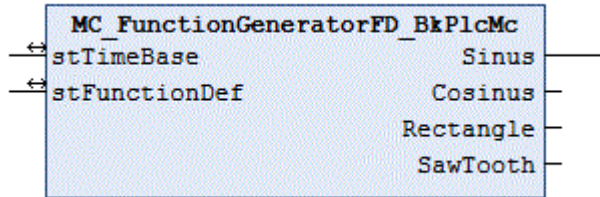
Unabhängig davon ob die Achse als ‚hybrid‘ gekennzeichnet ist ruft dieser Baustein für die Achse eine lokale Instanz von `MC_AxRtEncoder BkPlcMc()` [► 183] auf.



Wenn bei einer servoelektrisch-hydraulischen Achse kein Baustein dieses Typs aufgerufen wird werden Umschaltungen nicht korrekt behandelt. Damit könnte es zu unerwartetem Verhalten der Achse kommen. In diesem Fall wird die Achse in den Fehlerzustand versetzt und eine Meldung ausgegeben.

## 4.4.4 FunctionGenerator

### 4.4.4.1 MC\_FunctionGeneratorFD\_BkPlcMc (ab V3.0.31)



Der Funktionsbaustein berechnet die Signale eines Funktionsgenerators.

#### Ein-/Ausgänge

```
VAR_INOUT
  stTimeBase:    ST_FunctionGeneratorTB_BkPlcMc;
  stFunctionDef: ST_FunctionGeneratorFD_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
stTimeBase	ST_FunctionGeneratorTB_BkPlcMc	<b>stTimeBase:</b> Eine Struktur mit den Parametern der Zeitbasis dieses Funktionsgenerators.
stFunctionDef	ST_FunctionGeneratorFD_BkPlcMc	<b>stFunctionDef:</b> Eine Struktur mit den Definitionen der Ausgangssignale eines Funktionsgenerators.

#### Ausgänge

```
VAR_OUTPUT
  Sinus:      LREAL;
  Cosinus:    LREAL;
  Rectangle:  LREAL;
  SawTooth:   LREAL;
END_VAR
```

Name	Typ	Beschreibung
Sinus	LREAL	Die Ausgangssignale des Funktionsgenerators.
Cosinus	LREAL	
Rectangle	LREAL	
SawTooth	LREAL	

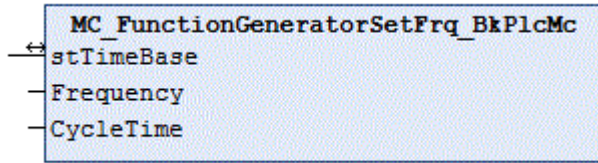
#### Verhalten des Bausteins

Aus **stTimeBase.CurrentRatio** und den Parametern in [stFunctionDef](#) [► 117] werden die Ausgangssignale ermittelt.

Die Zeitbasis in **stTimeBase** ist mit einem [MC\\_FunctionGeneratorTB\\_BkPlcMc](#) [► 212]() Baustein zu aktualisieren.

Zur Veränderung der Arbeitsfrequenz sollte ein [MC\\_FunctionGeneratorSetFrg\\_BkPlcMc](#) [► 212]() Baustein verwendet werden.

### 4.4.4.2 MC\_FunctionGeneratorSetFrq\_BkPlcMc (ab V3.0.31)



Der Funktionsbaustein aktualisiert die Arbeitsfrequenz einer Zeitbasis für einen oder mehrere Funktionsgeneratoren [► 211].

#### Eingänge

```
VAR_INPUT
    Frequency:    LREAL;
    CycleTime:    LREAL;
END_VAR
```

Name	Typ	Beschreibung
Frequency	LREAL	Die zu verwendende Arbeitsfrequenz.
CycleTime	LREAL	Die Zykluszeit der aufrufenden Task.

#### Ein-/Ausgänge

```
VAR_INOUT
    stTimeBase:    ST_FunctionGeneratorTB_BkPlcMc;
END_VAR
```

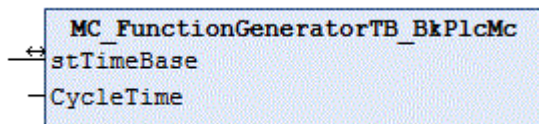
Name	Typ	Beschreibung
stTimeBase	ST_FunctionGeneratorTB_BkPlcMc	Eine Struktur mit den Parametern der Zeitbasis eines oder mehrerer Funktionsgeneratoren [► 118].

#### Verhalten des Bausteins

Der Baustein setzt **stTimeBase.Frequency** auf den übergebenen Wert. Dabei wird **stTimeBase.CurrentTime** bei Bedarf angepasst.

Der Baustein verhindert mit Hilfe von **stTimeBase.Freeze** eine Kollision mit MC\_FunctionGeneratorTB\_BkPlcMc [► 212]() Bausteinen. Somit kann er auch aus einer anderen Task aufgerufen werden.

### 4.4.4.3 MC\_FunctionGeneratorTB\_BkPlcMc (ab V3.0.31)



Der Funktionsbaustein aktualisiert eine Zeitbasis für einen oder mehrere Funktionsgeneratoren [► 211].

#### Eingänge

```
VAR_INPUT
    CycleTime:    LREAL;
END_VAR
```

Name	Typ	Beschreibung
CycleTime	LREAL	Die Zykluszeit der aufrufenden Task.

 Ein-/Ausgänge

```
VAR_INOUT
  stTimeBase:    ST_FunctionGeneratorTB_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
stTimeBase	ST_FunctionGeneratorTB_BkPlcMc	Eine Struktur mit den Parametern der Zeitbasis eines oder mehrerer Funktionsgeneratoren [► 118].

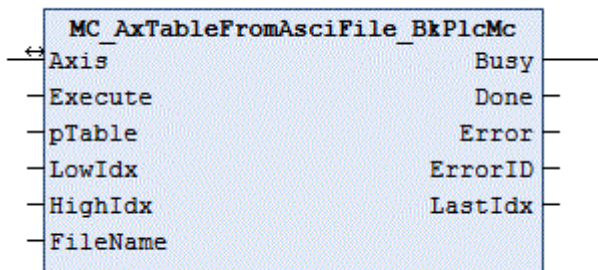
**Verhalten des Bausteins**

Wenn **stTimeBase Freeze** nicht gesetzt ist wird **stTimeBase.CurrentTime** mit **CycleTime** aktualisiert und **stTimeBase.CurrentRatio** ermittelt. Dabei wird **stTimeBase.Frequency** berücksichtigt.

Zur Veränderung der Arbeitsfrequenz sollte ein MC\_FunctionGeneratorSetFrq\_BkPlcMc [► 212]() Baustein verwendet werden.

## 4.4.5 TableFunctions

### 4.4.5.1 MC\_AxTableFromAsciiFile\_BkPlcMc (ab V3.0)



Der Funktionsbaustein liest den Inhalt einer Tabelle aus einer Textdatei.

 Eingänge

```
VAR_INPUT
  Execute:    BOOL:=FALSE;
  pTable:    POINTER TO LREAL:=0;
  LowIdx:    INT:=0;
  HighIdx:    INT:=0;
  FileName:    STRING(255) := '';
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Lesevorgang.
pTable	POINTER TO LREAL	Hier ist die Adresse eines ARRAY[nFirstIdx..nLastIdx,1..2] zu übergeben.
LowIdx	INT	Hier ist der untere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
HighIdx	INT	Hier ist der obere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
FileName	STRING	Hier kann ein Dateiname vorgegeben werden.

 Ein-/Ausgänge

```
VAR_INOUT
  Axis:    Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc zu übergeben.

## Ausgänge

```
VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
  LastIdx:   INT:=0;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Referenzfahrt signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
LastIdx	INT	Hier wird der Index der letzten durch den Lesevorgang definierte Tabellenzeile signalisiert.

## Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein die übergebenen Parameter. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **LowIdx** negativ ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **pTable=0** ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **LowIdx** und **HighIdx** eine Tabelle mit weniger als fünf Zeilen beschreiben wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.

Sind diese Überprüfungen ohne Probleme durchgeführt wird der Lesevorgang gestartet. Für die Dauer des Vorgangs ist **Busy** auf TRUE. Dabei kann es zu einigen weiteren Problemen kommen, die durch verschiedene Fehlercodes signalisiert werden. Ein erfolgreiches Lesen der Datei wird mit **Done** signalisiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Vorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende des Vorgangs (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

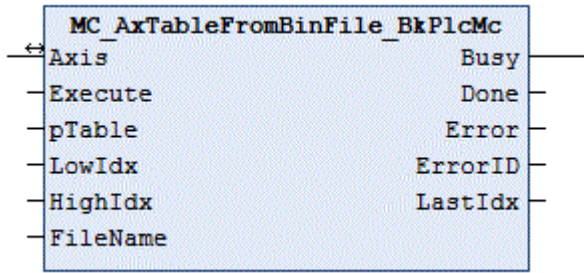
Wird ein **FileName** vorgegeben muss dieser vollständig (falls nötig inklusive Angabe des Laufwerks und des Pfades, immer inklusive Dateityp) sein, da er vom Baustein ohne jede weitere Veränderung oder Ergänzung benutzt wird.

Wird kein **FileName** vorgegeben verwendet der Baustein den Pfad und den Dateinamen, der durch den [MC\\_AxUtiStandardInit\\_BkPlcMc \[► 238\]](#) Baustein festgelegt wurde. Zur Unterscheidung von der Parameterdatei mit Dateityp DAT wird hier der Typ TXT verwendet.



Der Inhalt der Datei ist mit einem ASCII-Editor les- oder änderbar. Veränderungen des Inhalts können ein korrektes Lesen oder die vorgesehene Verwendung unmöglich machen oder die Wirkung der Tabelle auf kaum nachvollziehbare Weise verändern. Manuelle Veränderungen sollten darum wenn überhaupt mit der gebotenen Vorsicht und nur von sachkundigen Personen vorgenommen werden.

4.4.5.2 MC\_AxTableFromBinFile\_BkPlcMc (ab V3.0)



Der Funktionsbaustein liest den Inhalt einer Tabelle aus einer binären Datei.

Eingänge

```
VAR_INPUT
  Execute:    BOOL:=FALSE;
  pTable:    POINTER TO LREAL:=0;
  LowIdx:    INT:=0;
  HighIdx:   INT:=0;
  FileName:  STRING(255) := '';
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Lesevorgang.
pTable	POINTER TO LREAL	Hier ist die Adresse eines ARRAY[nFirstIdx..nLastIdx,1..2] zu übergeben.
LowIdx	INT	Hier ist der untere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
HighIdx	INT	Hier ist der obere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
FileName	STRING	Hier kann ein Dateiname vorgegeben werden.

Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

Ausgänge

```
VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
  LastIdx:   INT:=0;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Referenzfahrt signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

Name	Typ	Beschreibung
LastIdx	INT	Hier wird der Index der letzten durch den Lesevorgang definierte Tabellenzeile signalisiert.

### Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein die übergebenen Parameter. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **LowIdx** negativ ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **pTable=0** ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **LowIdx** und **HighIdx** eine Tabelle mit weniger als fünf Zeilen beschreiben wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.

Sind diese Überprüfungen ohne Probleme durchgeführt wird der Lesevorgang gestartet. Für die Dauer des Vorgangs ist **Busy** auf TRUE. Dabei kann es zu einigen weiteren Problemen kommen, die durch verschiedene Fehlercodes signalisiert werden. Ein erfolgreiches Lesen der Datei wird mit **Done** signalisiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Vorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende des Vorgangs (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

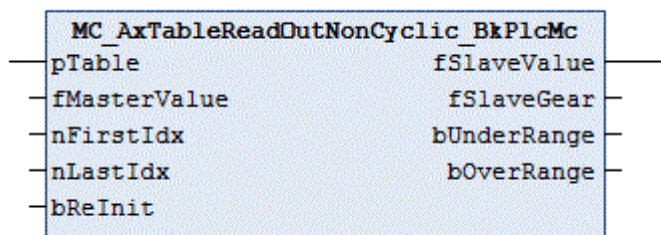
Wird ein **FileName** vorgegeben muss dieser vollständig (falls nötig inklusive Angabe des Laufwerks und des Pfades, immer inklusive Dateityp) sein, da er vom Baustein ohne jede weitere Veränderung oder Ergänzung benutzt wird.

Wird kein **FileName** vorgegeben verwendet der Baustein den Pfad und den Dateinamen, der durch den [MC\\_AxUtiStandardInit\\_BkPlcMc \[► 238\]](#) Baustein festgelegt wurde. Zur Unterscheidung von der Parameterdatei mit Dateityp DAT wird hier der Typ TBL verwendet.



Der Inhalt der Datei ist nicht mit einem ASCII-Editor les- oder änderbar.

### 4.4.5.3 MC\_AxTableReadOutNonCyclic\_BkPlcMc (ab V3.0)



Der Funktionsbaustein ermittelt mit Hilfe einer Tabelle die einem Master-Wert zugeordneten Slave-Werte.



Dieser Baustein ist eine Komponente von Kurvenscheiben oder ähnlichen nichtlinearen Kopplungen. Er wird in der Regel nicht direkt von einer Applikation aufgerufen.

#### Eingänge

```
VAR_INPUT
  pTable:      POINTER TO LREAL:=0;
  fMasterValue: LREAL:=0.0;
  nFirstIdx:   UDINT:=1;
  nLastIdx:    UDINT:=1;
  bReInit:     BOOL:=FALSE;
END_VAR
```



Name	Typ	Beschreibung
pTable	POINTER TO LREAL	Hier ist die Adresse eines ARRAY[nFirstIdx..nLastIdx,1..2] zu übergeben. <b>Eine nicht zutreffende Angabe an dieser Stelle führt zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>
fMasterValue	LREAL	Hier ist der Wert des Masters zu übergeben, für den die zugehörigen Slave-
nFirstIdx	UDINT	Hier ist der untere Index des ARRAY zu übergeben, dessen Adresse als pTable übergeben wird. <b>Eine nicht zutreffende Angabe an dieser Stelle führt zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>
nLastIdx	UDINT	Hier ist der obere Index des ARRAY zu übergeben, dessen Adresse als pTable übergeben wird. <b>Eine nicht zutreffende Angabe an dieser Stelle führt zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>
bReInit	BOOL	Dieser Eingang signalisiert dem Baustein, dass die Suchprozedur am Anfang der Tabelle aufgesetzt werden soll.

 **Ausgänge**

```
VAR_OUTPUT
  fSlaveValue:    LREAL:=0.0;
  fSlaveGear:    LREAL:=0.0;
  bUnderRange:   BOOL;
  bOverRange:    BOOL;
END_VAR
```

Name	Typ	Beschreibung
fSlaveValue	LREAL	Hier wird der zum fMasterValue gehörende Slave-Wert ausgegeben.
fSlaveGear	LREAL	Hier wird die lokale Steigung der Slave-Werte an der vom Master-Wert festgelegten Stelle der Tabelle ausgegeben.
bUnderRange	BOOL	Dieser Ausgang wird TRUE, Wenn der Master-Wert das untere Ende der Tabelle berührt oder unterschreitet.
bOverRange	BOOL	Dieser Ausgang wird TRUE, Wenn der Master-Wert das obere Ende der Tabelle berührt oder überschreitet.

**Verhalten des Bausteins**

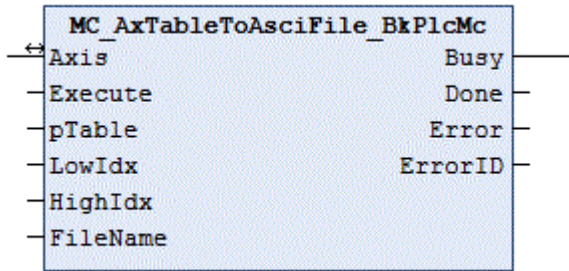
Der Baustein sucht innerhalb der übergebenen Tabelle nach einem Master-Wertepaar, das den übergebenen **fMasterValue** trifft oder umfasst. Innerhalb des gefundenen Intervalls wird eine lineare Zwischeninterpolation gerechnet. Das Ergebnis wird als **fSlaveValue** ausgegeben. Die bei dieser Berechnung ermittelte lokale Steigung wird als **fSlaveGear** ausgegeben.

Liegt der **fMasterValue** unterhalb des von der Tabelle beschriebenen Wertebereichs wird **bUnderRange** signalisiert. Als **fSlaveValue** wird der Wert ausgegeben, der dem untersten Tabellenpunkt zugeordnet ist. Als **fSlaveGear** wird 0.0 zurückgegeben.

Liegt der **fMasterValue** oberhalb des von der Tabelle beschriebenen Wertebereichs wird **bOverRange** signalisiert. Als **fSlaveValue** wird der Wert ausgegeben, der dem obersten Tabellenpunkt zugeordnet ist. Als **fSlaveGear** wird 0.0 zurückgegeben.

Der Rückgabewert **fSlaveGear** stellt das Verhältnis der ersten Ableitungen von **fMasterValue** und **fSlaveValue** dar. Stellt **fMasterValue** eine Position oder eine virtuelle Zeit dar ergibt die Multiplikation von Master-Fortschritts-geschwindigkeit und **fSlaveGear** die Slave-Sollgeschwindigkeit. Dies kann für die Erzeugung einer Vorsteuergeschwindigkeit ausgenutzt werden. Dazu ist vorzugsweise ein [MC\\_AxRtSetExtGenValues\\_BkPlcMc](#) [▶ 236] Baustein zu verwenden.

### 4.4.5.4 MC\_AxTableToAsciiFile\_BkPlcMc (ab V3.0)



Der Funktionsbaustein schreibt den Inhalt einer Tabelle in eine Textdatei.

#### Eingänge

```
VAR_INPUT
  Execute:    BOOL:=FALSE;
  pTable:     POINTER TO LREAL:=0;
  LowIdx:     INT:=0;
  HighIdx:    INT:=0;
  FileName:   STRING(255) := '';
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.
pTable	POINTER TO LREAL	Hier ist die Adresse eines ARRAY[nFirstIdx..nLastIdx,1..2] zu übergeben.
LowIdx	INT	Hier ist der untere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
HighIdx	INT	Hier ist der obere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
FileName	STRING	Hier kann ein Dateiname vorgegeben werden.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Referenzfahrt signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein die übergebenen Parameter. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **LowIdx** negativ ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **pTable=0** ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **LowIdx** und **HighIdx** eine Tabelle mit weniger als fünf Zeilen beschreiben wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.

Sind diese Überprüfungen ohne Probleme durchgeführt wird der Schreibvorgang gestartet. Für die Dauer des Vorgangs ist **Busy** auf TRUE. Dabei kann es zu einigen weiteren Problemen kommen, die durch verschiedene Fehlercodes signalisiert werden. Ein erfolgreiches Schreiben der Datei wird mit **Done** signalisiert.

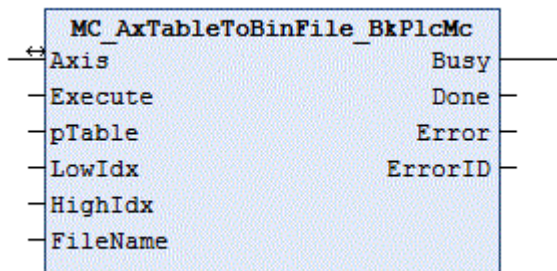
Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Vorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende des Vorgangs (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

Wird ein **FileName** vorgegeben muss dieser vollständig (falls nötig inklusive Angabe des Laufwerks und des Pfades, immer inklusive Dateityp) sein, da er vom Baustein ohne jede weitere Veränderung oder Ergänzung benutzt wird.

Wird kein **FileName** vorgegeben verwendet der Baustein den Pfad und den Dateinamen, der durch den MC\_AxUtiStandardInit\_BkPlcMc [▶ 238] Baustein festgelegt wurde. Zur Unterscheidung von der Parameterdatei mit Dateityp DAT wird hier der Typ TXT verwendet.

**i** Der Inhalt der Datei ist mit einem ASCII-Editor les- oder änderbar. Veränderungen des Inhalts können ein korrektes Lesen oder die vorgesehene Verwendung unmöglich machen oder die Wirkung der Tabelle auf kaum nachvollziehbare Weise verändern. Manuelle Veränderungen sollten darum wenn überhaupt mit der gebotenen Vorsicht und nur von sachkundigen Personen vorgenommen werden.

**4.4.5.5 MC\_AxTableToBinFile\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein schreibt den Inhalt einer Tabelle in eine binäre Datei.

**Eingänge**

```
VAR_INPUT
  Execute:      BOOL:=FALSE;
  pTable:      POINTER TO LREAL:=0;
  LowIdx:      INT:=0;
  HighIdx:     INT:=0;
  FileName:    STRING(255) := '';
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.
pTable	POINTER TO LREAL	Hier ist die Adresse eines ARRAY[nFirstIdx..nLastIdx,1..2] zu übergeben.

Name	Typ	Beschreibung
LowIdx	INT	Hier ist der untere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
HighIdx	INT	Hier ist der obere Index des ARRAY zu übergeben, dessen Adresse als <b>pTable</b> übergeben wird.
FileName	STRING	Hier kann ein Dateiname vorgegeben werden.

### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

### Ausgänge

```
VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  Error:      BOOL;
  ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Abarbeitung der Referenzfahrt signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

## Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein die übergebenen Parameter. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn **LowIdx** negativ ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **pTable=0** ist wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.
- Wenn **LowIdx** und **HighIdx** eine Tabelle mit weniger als fünf Zeilen beschreiben wird mit **Error** und **ErrorID=dwTcHydErrCdTblEntryCount** reagiert.

Sind diese Überprüfungen ohne Probleme durchgeführt wird der Schreibvorgang gestartet. Für die Dauer des Vorgangs ist **Busy** auf TRUE. Dabei kann es zu einigen weiteren Problemen kommen, die durch verschiedene Fehlercodes signalisiert werden. Ein erfolgreiches Schreiben der Datei wird mit **Done** signalisiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Vorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende des Vorgangs (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

Wird ein **FileName** vorgegeben muss dieser vollständig (falls nötig inklusive Angabe des Laufwerks und des Pfades, immer inklusive Dateityp) sein, da er vom Baustein ohne jede weitere Veränderung oder Ergänzung benutzt wird.

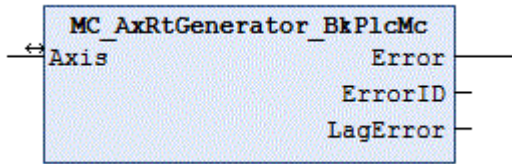
Wird kein **FileName** vorgegeben verwendet der Baustein den Pfad und den Dateinamen, der durch den MC\_AxUtiStandardInit\_BkPlcMc [► 238] Baustein festgelegt wurde. Zur Unterscheidung von der Parameterdatei mit Dateityp DAT wird hier der Typ TBL verwendet.



Der Inhalt der Datei ist nicht mit einem ASCII-Editor les- oder änderbar.

## 4.4.6 Generatoren

### 4.4.6.1 MC\_AxRtGenerator\_BkPlcMc



Der Funktionsbaustein übernimmt die Aufgabe des Stellwertgenerators. Dazu wird in Abhängigkeit des in **Axis.ST\_TcHydAxParam** [► 120] als nProfileType eingestellten Wertes ein profilspezifischer Baustein aufgerufen.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Error:      BOOL;
  ErrorID:    UDINT;
  LagError:   BOOL;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
LagError	BOOL	Wenn der Schleppabstand die eingestellten Grenzen verletzt, wird dies hier signalisiert. Dieses Signal steht auch bei nicht aktivierter Schleppabstandsüberwachung zur Verfügung.

#### Verhalten des Bausteins

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn einer der Pointer nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** oder **dwTcHydErrCdPtrMcPlc** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird die Stellwertgenerierung der Achse entsprechend nProfileType in **Axis.ST\_TcHydAxParam** [► 120] durch Aufruf eines entsprechenden Bausteins durchgeführt.

Der Ausgang **LagError** signalisiert, ob der aktuelle Schleppabstand der Achse die eingestellten Grenzen überschreitet. Nur wenn bMaxLagEna in **Axis.ST\_TcHydAxParam** [► 120] gesetzt ist wird dann die Achse in einen Fehlerzustand versetzt.

Derzeit stehen die folgenden Generatoren zur Verfügung:

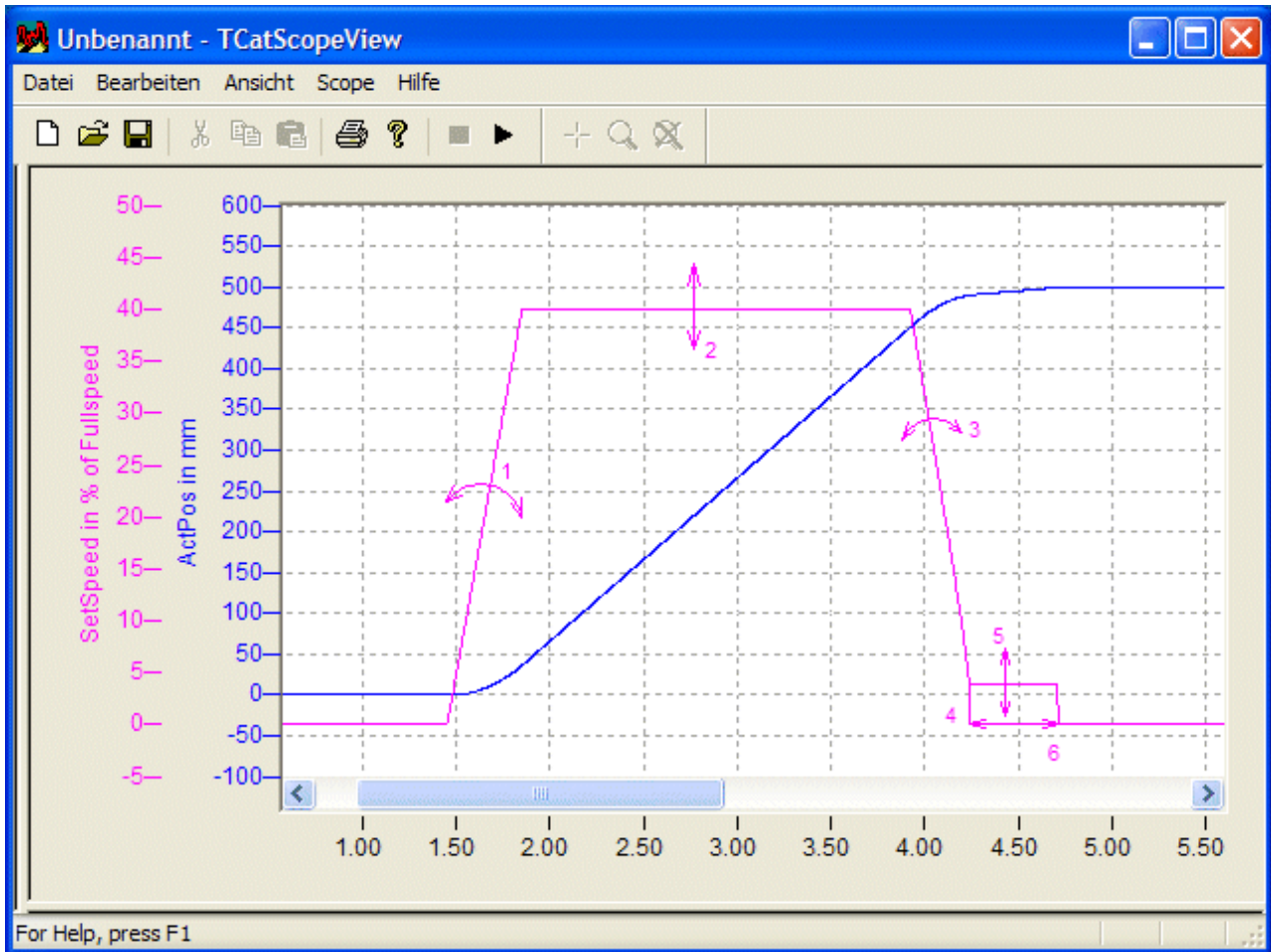
nProfileType	Beschreibung
iTcMc_ProfileCtrlBased [► 223]	<b>Standardprofil:</b> Einstufige zeitgesteuerte Beschleunigung, weggesteuerte (Wurzelbildner) Bremsrampe, Zielanfahrt mit Schleichgeschwindigkeit, wählbares Verhalten im Ruhezustand.

nProfileType	Beschreibung
	<p>Das Starten einer fahrenden Achse (neues Ziel, neue Geschwindigkeit usw.) ist außer im Störzustand oder einem Zustand mit abhängiger Stellwertgenerierung immer möglich.</p> <p><b>Info:</b> Es kann zu einem Überfahren des neuen Ziels auch dann kommen, wenn sich die Achse zum Zeitpunkt des Startens noch vor der Zielposition befindet.</p> <p><b>Info:</b> Der Baustein kann so parametrieren werden, dass er unter bestimmten durch seine Parameter definierten Bedingungen automatisch starten und in einen aktiven Fahrzustand übergehen wird.</p> <p><b>Info:</b> Dieser Generatortyp kann optional auch rein zeitgesteuert mit ständig geschlossenem Lageregler arbeiten.</p>
iTcMc_ProfileJerkBased	<p><b>Standardprofil:</b> Ein- oder zweistufige zeitgesteuerte Beschleunigung durch optionale Ruckbegrenzung, weggesteuerte (Wurzelbildner) Bremsrampe, Zielfahrt mit Ruckbegrenzung, wählbares Verhalten im Ruhezustand.</p> <p>Das Starten einer fahrenden Achse (neues Ziel, neue Geschwindigkeit usw.) ist außer im Störzustand oder einem Zustand mit abhängiger Stellwertgenerierung immer möglich.</p> <p><b>Info:</b> Es kann zu einem Überfahren des neuen Ziels auch dann kommen, wenn sich die Achse zum Zeitpunkt des Startens noch vor der Zielposition befindet.</p> <p><b>Info:</b> Der Baustein kann so parametrieren werden, dass er unter bestimmten durch seine Parameter definierten Bedingungen automatisch starten und in einen aktiven Fahrzustand übergehen wird.</p> <p><b>Info:</b> Dieser Generatortyp kann optional auch rein zeitgesteuert mit ständig geschlossenem Lageregler arbeiten.</p> <p><b>Info:</b> Einige Funktionen werden von diesem Generatortyp nicht oder unvollständig unterstützt.</p>
iTcMc_ProfileTimePosCtrl	<p><b>Info:</b> Nur aus Kompatibilitätsgründen vorhanden, wird in Kürze nicht mehr unterstützt werden.</p> <p><b>Sonderprofil:</b> Zweistufige (zunächst zeitgesteuert, dann weggesteuert mit Wurzelbildner) Beschleunigung, weggesteuerte (Wurzelbildner) Bremsrampe, Zielfahrt mit Schleichgeschwindigkeit, wählbares Verhalten im Ruhezustand.</p> <p>Das Starten einer fahrenden Achse (neues Ziel, neue Geschwindigkeit usw.) ist nicht möglich.</p>
iTcMc_ProfileCosine	<p><b>Info:</b> Nur aus Kompatibilitätsgründen vorhanden, wird in Kürze nicht mehr unterstützt werden.</p> <p><b>Sonderprofil:</b> Zweistufige (zunächst zeitgesteuert, dann weggesteuert mit Cosinusbildner) Beschleunigung, weggesteuerte (Cosinusbildner) Bremsrampe, Zielfahrt mit Schleichgeschwindigkeit, wählbares Verhalten im Ruhezustand.</p> <p>Das Starten einer fahrenden Achse (neues Ziel, neue Geschwindigkeit usw.) ist nicht möglich.</p>
iTcMc_ProfileTimeRamp [▶ 225]	<p><b>Sonderprofil:</b> Einstufige zeitgesteuerte Beschleunigung, zeitgesteuerte Bremsrampe, Zielfahrt mit Schleichgeschwindigkeit, eingeschränkt wählbares Verhalten im Ruhezustand. Der Generator arbeitet mit Positionsnocken anstelle eines Encoders.</p> <p>Das Starten einer fahrenden Achse (neues Ziel, neue Geschwindigkeit usw.) ist außer im Störzustand möglich.</p> <p><b>Info:</b> Dieser Generatortyp ist für Achsen vorgesehen, die an Stelle eines Encoders lediglich über digitale Nocken verfügen.</p>

Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollte zur Vereinfachung ein Baustein des Typs `MC_AxStandardBody_BkPlcMc` [▶ 237] verwendet werden.

iTcMc\_ProfileCtrlBased

Es wird ein Profil mit einer zeitgesteuerten Beschleunigungsphase, einer weggesteuerten Bremsphase nach dem Prinzip des Wurzelbildners und eine Zielannäherung mit Schleichgeschwindigkeit erzeugt.



Die Pfeile am Stellwertprofil deuten die Gestaltungsmöglichkeiten durch Parameter des Bewegungsauftrags bzw. der Achse an. Zunächst wird mit einer zeitgesteuerten Rampenfunktionen "1" auf die geforderten Fahrgeschwindigkeit "2" beschleunigt. Dieser Stellwert wird aufrechterhalten, bis ein beim Start vorberechneter Wegpunkt erreicht wird. Ab hier wird mit einer weggesteuerten Rampe "3" so vom Fahrstellwert auf den Schleichstellwert "5" heruntergebremst, dass dieser Stellwert in einer festgelegten Entfernung "4" zum Ziel erreicht wird. Dieser Stellwert wird aufrechterhalten, bis die Entfernung sich dem Ziel bis auf eine Reststrecke "6" genähert hat. Jetzt wird in das Ruheverhalten umgeschaltet.

**Aktive Parameter im Fahrprofil**

**Startrampe "1"**: Wirksam wird der kleinste von folgenden Werten: **fMaxAcc** und **fAcc** in **Axis.ST TcHydAxParam** [▶ 120], **Acceleration** des zum Achsstart verwendeten Bausteins (Beispiel: [MC MoveAbsolute BkPlcMc](#) [▶ 72]).

**Fahrphase "2"**: Wirksam wird der kleinste von folgenden Werten: **fRefVelo** und **fMaxVelo** in **Axis.ST TcHydAxParam** [▶ 120], **Velocity** des zum Achsstart verwendeten Bausteins (Beispiel: [MC MoveAbsolute BkPlcMc](#) [▶ 72]).

**Bremsrampe "3"**: Wirksam wird der kleinste von folgenden Werten: **fMaxDec** und **fDec** in **Axis.ST TcHydAxParam** [▶ 120], **Deceleration** des zum Achsstart verwendeten Bausteins (Beispiel: [MC MoveAbsolute BkPlcMc](#) [▶ 72]).

**Schleichphase "4", "5"**: Wirksam werden die Werte **fCreepSpeed** und **fCreepDistance** in **Axis.ST TcHydAxParam** [▶ 120].

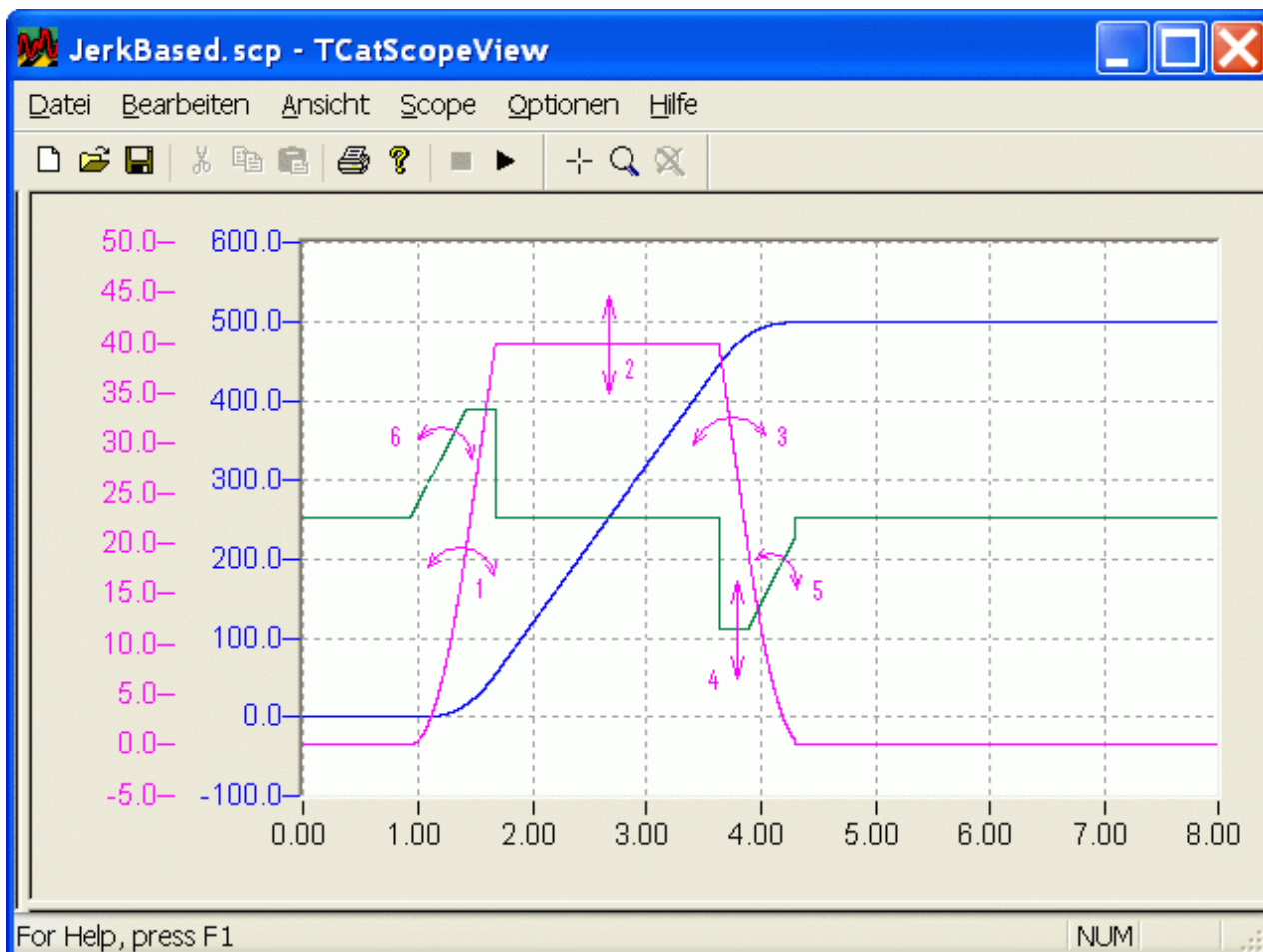
**Übergang im Ziel "6"**: Wirksam wird **fBrakeDistance** und/oder **fBrakeDeadTime** in **Axis.ST\_TcHydAxParam** [► 120].

### Automatisches Starten der Achse

Wenn die Abweichung zwischen der Istposition und der aktuell geltenden Zielposition den Wert in **Axis.ST\_TcHydAxParam** [► 120].**fReposDistance** überschreitet wird ein automatischer Start ausgelöst.

### iTcMc\_ProfileJerkBased

Es wird ein Profil mit einer zeitgesteuerten Beschleunigungsphase (mit optionaler Ruckbegrenzung), einer weggesteuerten Bremsrampe nach dem Prinzip des Wurzelbildners und einer Zielfahrt mit Ruckbegrenzung erzeugt.



Die Pfeile am Stellwertprofil deuten die Gestaltungsmöglichkeiten durch Parameter des Bewegungsauftrags bzw. der Achse an. Zunächst wird mit einer zeitgesteuerten Rampenfunktionen "1" auf die geforderten Fahrgeschwindigkeit "2" beschleunigt. Dabei kann die optionale Ruckbegrenzung "6" wirksam werden. Die Fahrgeschwindigkeit wird aufrechterhalten, bis ein beim Start vorberechneter Wegpunkt erreicht wird. Ab hier wird mit einer weggesteuerten Rampe "3" heruntergebremst bis die Entfernung sich dem Ziel bis auf die Reststrecke genähert hat. Dabei wird die Verzögerung "4" mit einem Begrenzten Ruck "5" zum Ziel hin abgebaut. Jetzt wird in das Ruheverhalten umgeschaltet.

### Aktive Parameter im Fahrprofil

**Startrampe "1"**: Wirksam wird der kleinste von folgenden Werten: **fMaxAcc** und **fAcc** in **Axis.ST\_TcHydAxParam** [► 120], **Acceleration** des zum Achsstart verwendeten Bausteins (Beispiel: **MC\_MoveAbsolute BkPlcMc** [► 72]).



**Fahrphase "2"**: Wirksam wird der kleinste von folgenden Werten: **fRefVelo** und **fMaxVelo** in **Axis.ST\_TcHydAxParam** [▶ 120], **Velocity** des zum Achsstart verwendeten Bausteins (Beispiel: **MC\_MoveAbsolute BkPlcMc** [▶ 72]).

**Bremsrampe "3", "4"**: Wirksam wird der kleinste von folgenden Werten: **fMaxDec** und **fDec** in **Axis.ST\_TcHydAxParam** [▶ 120], **Deceleration** des zum Achsstart verwendeten Bausteins (Beispiel: **MC\_MoveAbsolute BkPlcMc** [▶ 72]).

**Übergang im Ziel "5"**: Wirksam werden **fMaxJerk** in **Axis.ST\_TcHydAxParam** [▶ 120] und **fJerk** des zum Achsstart verwendeten Bausteins (Beispiel: **MC\_MoveAbsolute BkPlcMc** [▶ 72]) sowie **fBrakeDistance** und/oder **fBrakeDeadTime** in **Axis.ST\_TcHydAxParam** [▶ 120].

**iTcMc\_ProfileTimePosCtrl**

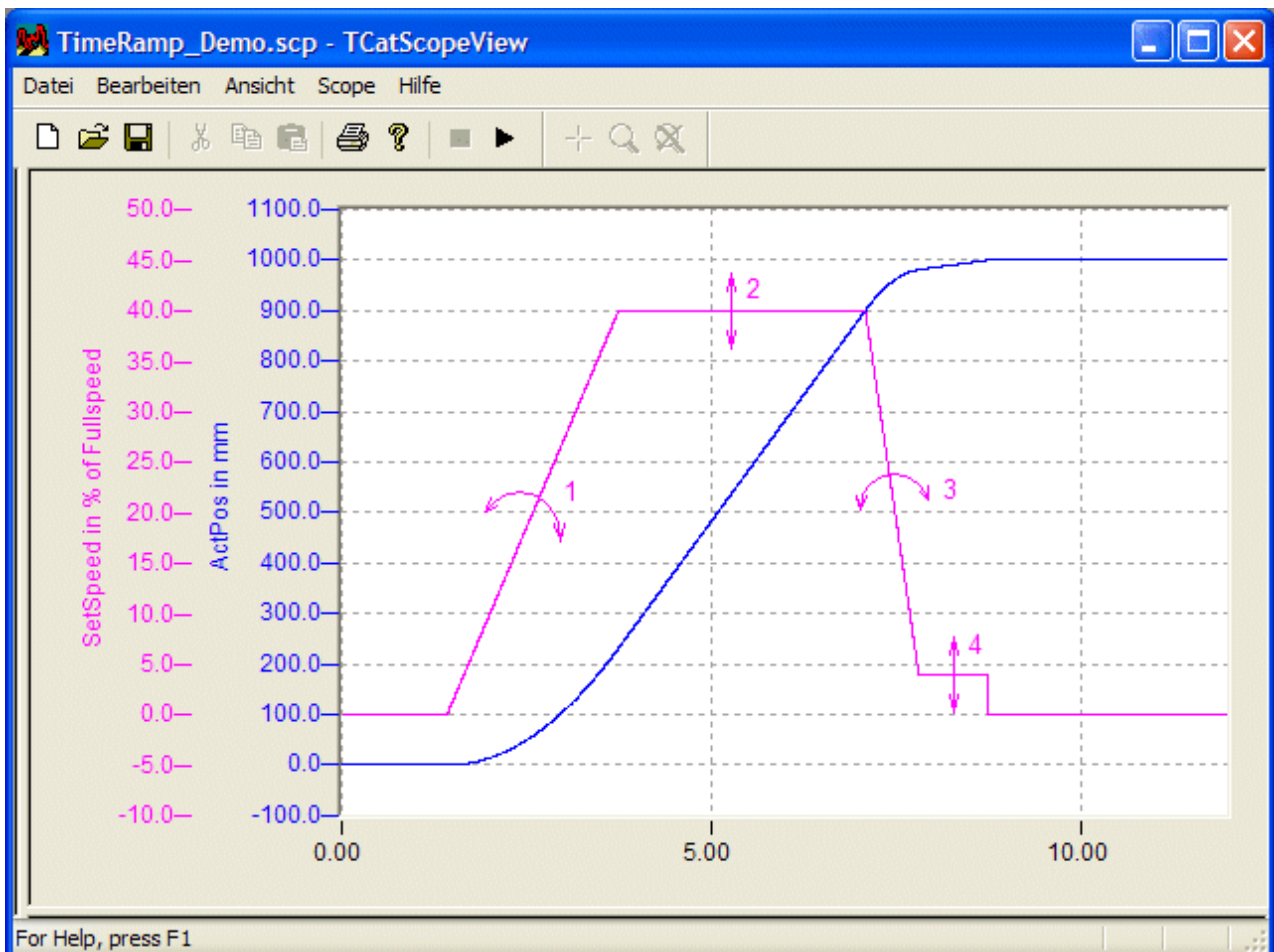
**i** Nur aus Kompatibilitätsgründen vorhanden, wird in Kürze nicht mehr unterstützt werden. Er sollte bei neuen Projekten nicht eingesetzt und bei der Überarbeitung von bestehenden Projekten möglichst ersetzt werden.

**iTcMc\_ProfileCosine**

**i** Nur aus Kompatibilitätsgründen vorhanden, wird in Kürze nicht mehr unterstützt werden. Er sollte bei neuen Projekten nicht eingesetzt und bei der Überarbeitung von bestehenden Projekten möglichst ersetzt werden.

**iTcMc\_ProfileTimeRamp**

Es wird ein Profil mit einer zeitgesteuerten Beschleunigungsphase, einer zeitgesteuerten Bremsphase und eine Zielannäherung mit Schleichgeschwindigkeit erzeugt.



Die Pfeile am Stellwertprofil deuten die Gestaltungsmöglichkeiten durch Parameter des Bewegungsauftrags bzw. der Achse an. Zunächst wird mit einer zeitgesteuerten Rampenfunktionen "1" auf die geforderten Fahrgeschwindigkeit "2" beschleunigt. Dieser Stellwert wird aufrechterhalten, bis der richtungsspezifische Zielfensternocken erkannt wird. Ab hier wird mit einer zeitgesteuerten Rampe "3" vom Fahrstellwert auf den Schleichstellwert "5" heruntergebremst. Dieser Stellwert wird aufrechterhalten, bis der richtungsspezifische Zielnocken erkannt wird. Jetzt wird in das Ruheverhalten umgeschaltet.

### Aktive Parameter im Fahrprofil

**Startrampe "1"**: Wirksam wird **fStartRamp** in **Axis.ST\_TcHydAxParam** [► 120].

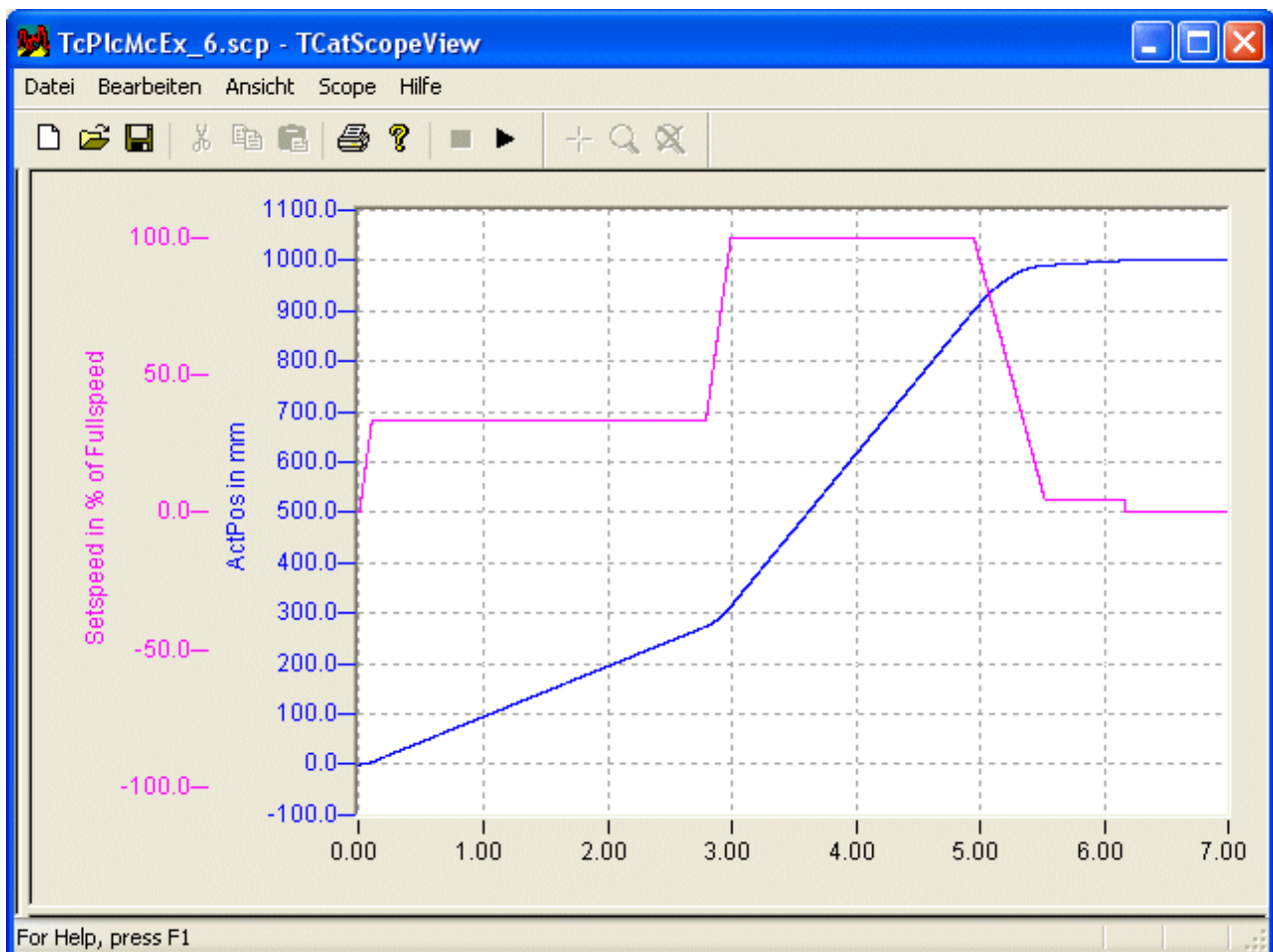
**Fahrphase "2"**: Wirksam wird der kleinste von folgenden Werten: **fRefVelo** und **fMaxVelo** in **Axis.ST\_TcHydAxParam** [► 120], **Velocity** des zum Achsstart verwendeten Bausteins (Beispiel: **MC\_MoveAbsolute** **BkPlcMc** [► 72]).

**Bremsrampe "3"**: Wirksam wird **fStopRamp** in **Axis.ST\_TcHydAxParam** [► 120].

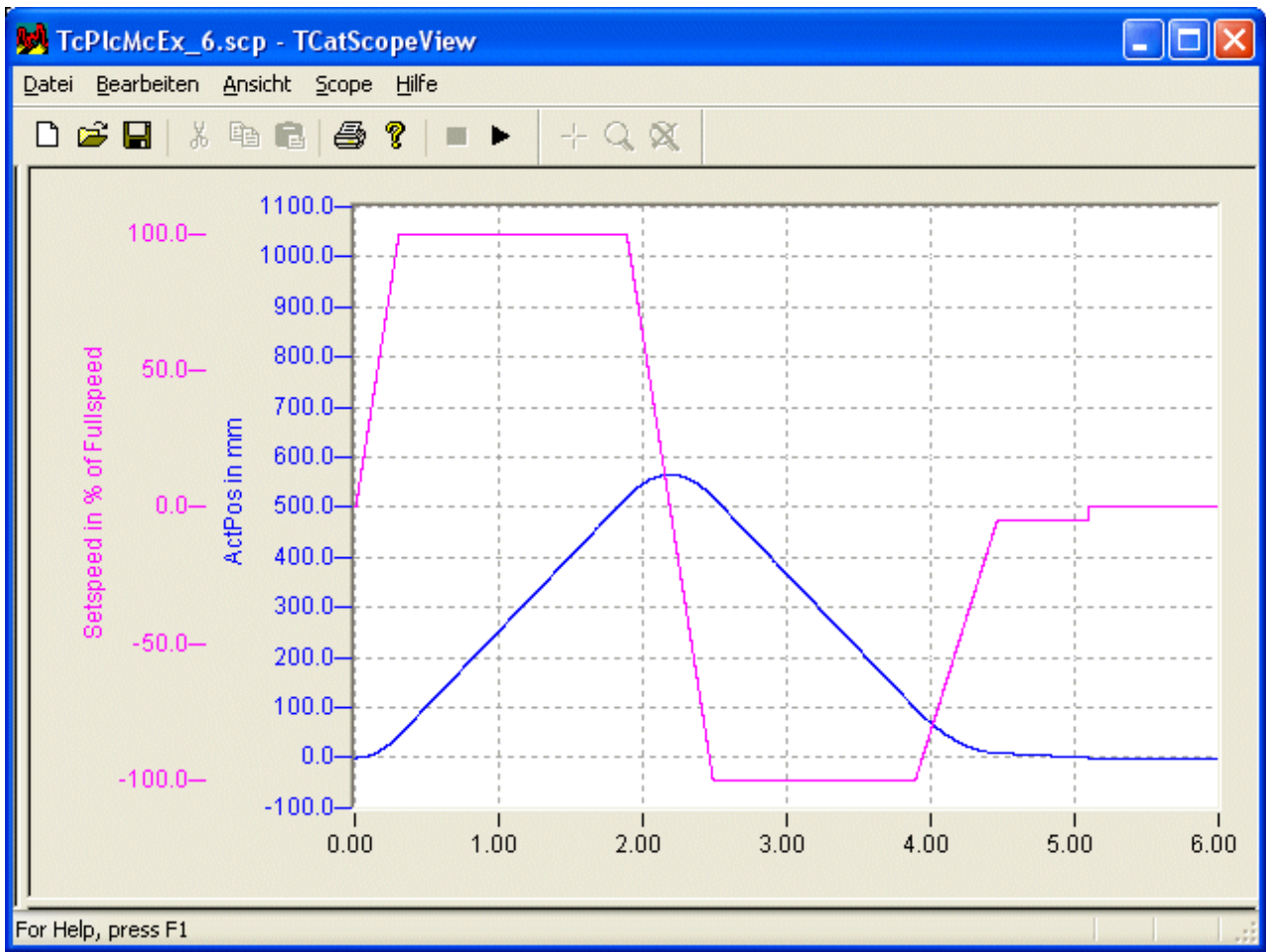
**Schleichphase "4"**: Wirksam wird **fCreepSpeed** in **Axis.ST\_TcHydAxParam** [► 120].

### Verhalten des Bausteins beim Durchstarten während einer Bewegung

Wird während einer aktiven Bewegung ein weiterer Startbefehl gegeben ist sind zwei Fälle zu unterscheiden.



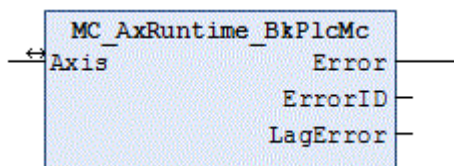
Dieses Profil entsteht beim Nachstarten in gleicher Richtung mit anderer (hier mit höherer) Geschwindigkeit.



Dieses Profil entsteht beim Nachstarten in entgegengesetzter Richtung, hier mit gleicher Geschwindigkeit.

Dieser Profiltyp kann sinnvoll nur in Kombination mit dem Encodertyp [iTcMc EncoderDigCam \[▶ 191\]](#) verwendet werden. Siehe auch Sonderfall digitale Positionsnocken.

#### 4.4.6.2 MC\_AxRuntime\_BkPlcMc (ab V3.0)



Der Baustein integriert je einen Baustein vom Typ [MC\\_AxRtGenerator\\_BkPlcMc\(\) \[▶ 221\]](#) und vom Typ [MC\\_AxRtController\\_BkPlcMc\(\) \[▶ 229\]](#). Die Ausgänge des Generators werden weitergegeben.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [▶ 84]</a> zu übergeben.

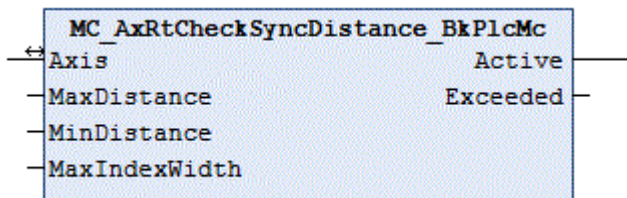
**Ausgänge**

```
VAR_OUTPUT
  Error:      BOOL;
  ErrorID:    UDINT;
  LagError:   BOOL;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
LagError	BOOL	Wenn der Schleppabstand die eingestellten Grenzen verletzt, wird dies hier signalisiert. Dieses Signal steht auch bei nicht aktivierter Schleppabstandsüberwachung zur Verfügung.

## 4.4.7 Runtime

### 4.4.7.1 MC\_AxRtCheckSyncDistance\_BkPlcMc (ab V3.0)



Der Funktionsbaustein überprüft, ob bei einer Referenzfahrt nach dem Verlassen des Nockens eine unzulässige Strecke zurückgelegt wird.

**Eingänge**

```
VAR_INPUT
  MaxDistance:  LREAL;
  MinDistance:  LREAL;
  MaxIndexWidth: LREAL;
END_VAR
```

Name	Typ	Beschreibung
MaxDistance	LREAL	[mm] Hier ist die maximal zulässige Strecke vorzugeben, die vom Referenznocken bis zum Erreichen des Nullimpulses zurückgelegt werden darf.
MinDistance	LREAL	[mm] Hier ist die minimal zulässige Strecke vorzugeben, die vom Referenznocken bis zum Erreichen des Nullimpulses zurückgelegt werden muss.
MaxIndexWidth	LREAL	[mm] Hier ist die minimal zulässige Strecke vorzugeben, die zum Verlassen des Referenznockens zurückgelegt werden muss. (ab V3.0.20)

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Active:          BOOL;
  Exceeded:       BOOL;
END_VAR
```

Name	Typ	Beschreibung
Active	BOOL	Hier wird signalisiert, dass die Achse den Nocken verlassen hat und den Nullimpuls des Gebers erwartet.
Exceeded	BOOL	Hier wird signalisiert, dass die Achse nach dem Verlassen des Nockens einen Weg von mehr als MaxDistance zurückgelegt hat ohne dass der Nullimpuls des Gebers erkannt wurde.

**Verhalten des Bausteins**

Der Baustein erkennt den Teil der Referenzfahrt, in dem die Achse den Nullimpuls des Gebers sucht und überwacht die dabei zurückgelegte Strecke. Dabei können zwei Probleme erkannt werden:

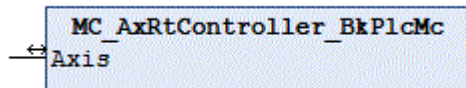
- Die Achse legt **MaxIndexWidth** zurück, ohne dass die fallenden Flanken des Referenznockens erkannt werden.
- Die Achse legt **MaxDistance** zurück, ohne dass ein Nullimpuls erkannt wird.
- Der Nullimpuls wird erkannt, bevor die Achse **MinDistance** zurückgelegt hat.

Erkannte Probleme werden mit **Exceeded** signalisiert. Soll dies zu einem Achsfehler führen muss von der Applikation ein solcher Zustandswechsel vorgegeben werden. Hierzu sollte ein `MC_AxRtGoErrorState_BkPlcMc` [[▶ 233](#)] Baustein und ein codierter `Error Code` [[▶ 321](#)] verwendet werden.



Die Überwachung auf MinDistance und MaxDistance kann unterdrückt werden, indem der jeweilige Parameter auf 0.0 gestellt wird.

**4.4.7.2 MC\_AxRtController\_BkPlcMc**



Dieser Baustein enthält den Standard-Lageregler der Achse.

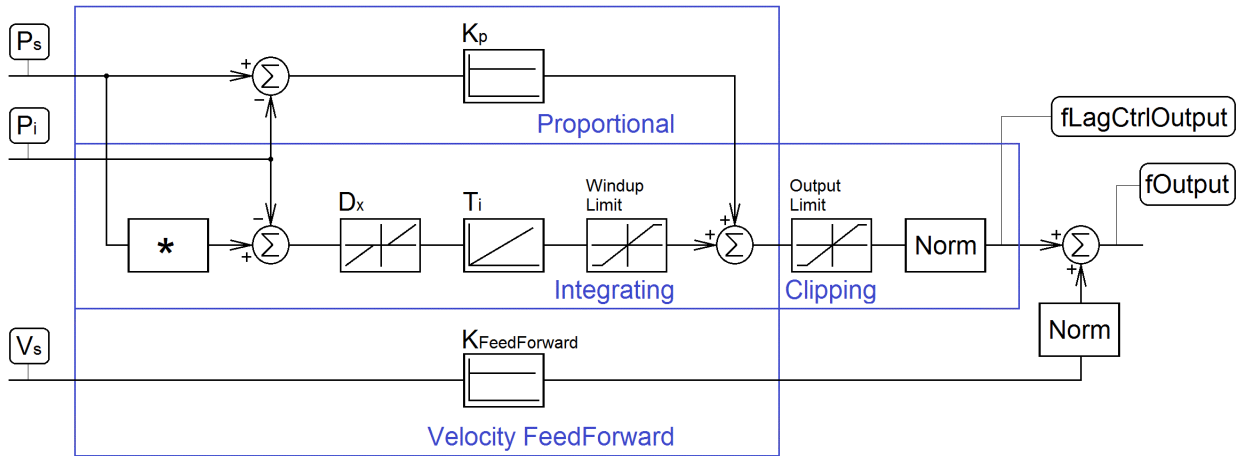
Bei Bedarf kann an Stelle dieses Bausteins ein Baustein vom erheblich komplexeren Typ `MC_AxRtPosPiControllerEx_BkPlcMc` [[▶ 169](#)] verwendet werden.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

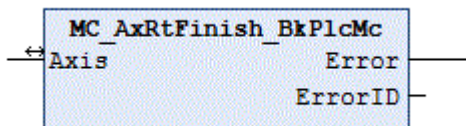
Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist eine Variable vom Typ <code>Axis Ref BkPlcMc</code> [ <a href="#">▶ 84</a> ] zu übergeben.

**Struktur des Reglers**



**i** Der Parameter  $T_i$  wird von diesem Regler als  $K_i$  verwendet. Ein Wert von 0.0 deaktiviert den I-Anteil. Steigende Werte erzeugen stärker werde Reaktionen des I-Anteils.

**4.4.7.3 MC\_AxRtFinish\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein sorgt für die Anpassung des erzeugten Stellwerts an die Besonderheiten der Achse. Soll eine Kennlinienlinearisierung durchgeführt werden ist ein MC\_AxRtFinishLinear\_BkPlcMc [▶ 231] Baustein zu verwenden.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [▶ 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Error:      BOOL;
  ErrorID:    UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

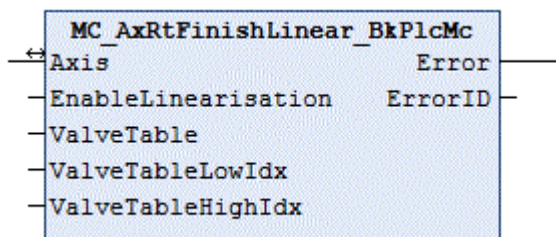
- Wenn einer der Pointer nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** oder **dwTcHydErrCdPtrMcPlc** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnte wird die Stellwertanpassung für die Achse entsprechend der Werte in **Axis.ST\_TcHydAxParam** [► 120] durchgeführt.

- Der Vorschubstellwert und die Lageregelreaktion werden zum Ausgabestellwert zusammengefasst.
- Die Flächenkompensation wird berücksichtigt.
- Ein Kennlinienknick wird kompensiert.
- Die Überdeckungskompensation, der Klemmstellwert und die Offsetkompensation werden eingerechnet.

Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollte zur Vereinfachung ein Baustein des Typs MC\_AxStandardBody\_BkPlcMc [► 237] verwendet werden.

#### 4.4.7.4 MC\_AxRtFinishLinear\_BkPlcMc (ab V3.0.16)



Der Funktionsbaustein sorgt für die Anpassung des erzeugten Stellwerts an die Besonderheiten der Achse unter Berücksichtigung einer Kennlinie.

#### Eingänge

```
VAR_INPUT
  EnableLinearisation:   BOOL;
  ValveTable:           POINTER TO LREAL:=0;
  ValveTableLowIdx:    INT:=0;
  ValveTableHighIdx:   INT:=0;
END_VAR
```

Name	Typ	Beschreibung
EnableLinearisation	BOOL	Ein TRUE an diesem Eingang aktiviert die Linearisierung.
ValveTable	POINTER TO LREAL	Die Adresse der Linearisierungstabelle ist hier zu übergeben. Dabei sollte es sich möglichst um den ValveCharacteristicTable einer <u>ST_TcMcAutoldent</u> [► 118] handeln, die mit der Achse verbunden ist. Wird hier ein Null-Pointer übergeben wird die Linearisierungstabelle und die Begrenzungs-Indizes der mit der Achse verbundenen <u>ST_TcMcAutoldent</u> Struktur verwendet. Ist eine solche Struktur nicht vorhanden zeigt der Baustein das Verhalten eines <u>MC_AxRtFinish()</u> Bausteins.
ValveTableLowIdx	INT	Der Index des ersten Punktes in der Linearisierungstabelle.
ValveTableHighIdx	INT	Der Index des letzten Punktes in der Linearisierungstabelle. Dabei sollte es sich möglichst um den ValveCharacteristicTblCount einer <u>ST_TcMcAutoldent</u> [► 118] handeln, die mit der Achse verbunden ist.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

## VAR\_ Ausgänge

```

UTPUT
  Error:          BOOL;
  ErrorID:       UDINT;
END_VAR

```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

## Verhalten des Bausteins

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt werden:

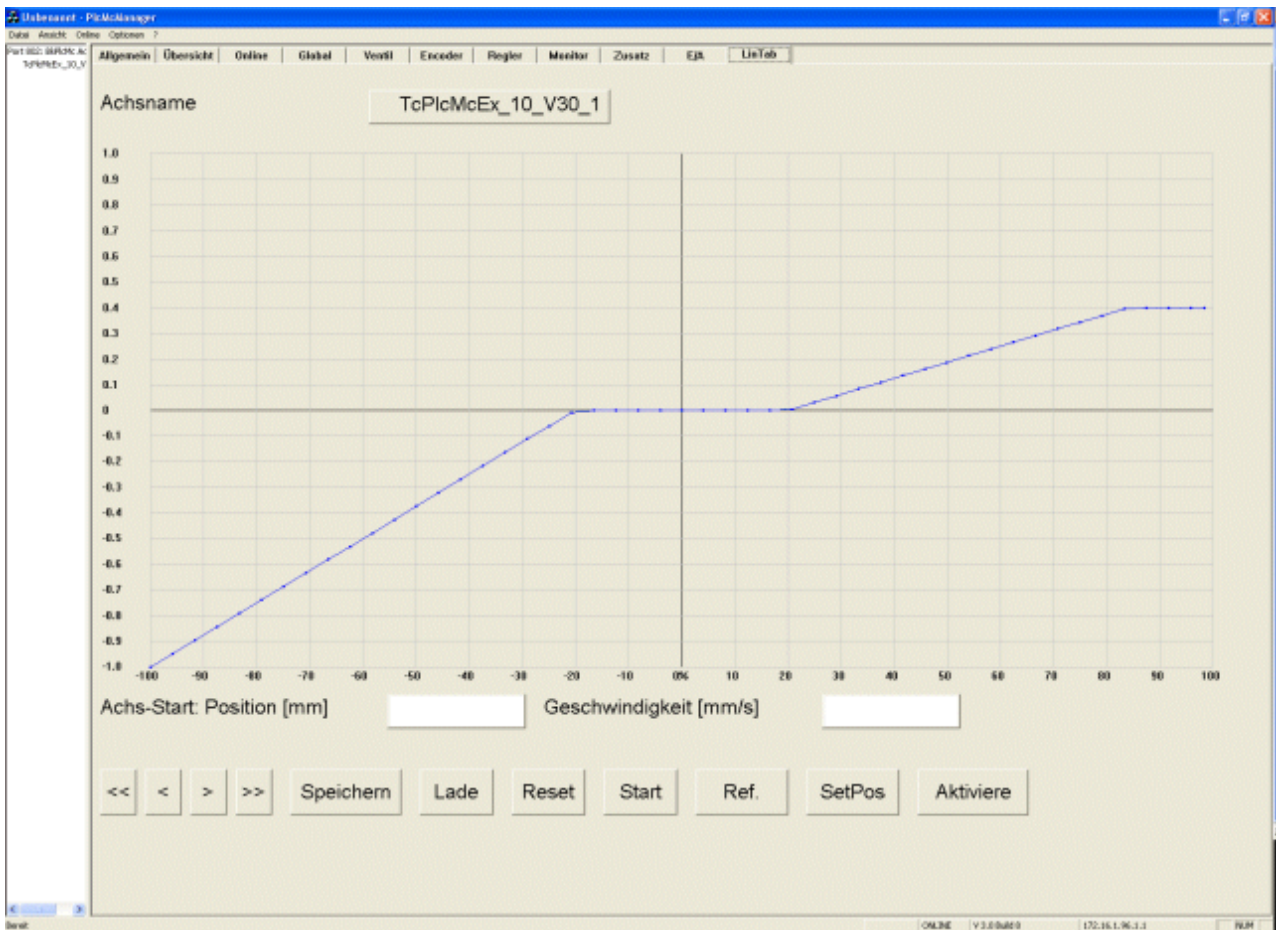
- **EnableLinearisation** ist FALSE.
- Es ist kein **ValveTable** verfügbar.
- **ValveTableLowIdx** ist kleiner als 0.
- **ValveTableHighIdx** ist kleiner oder gleich **ValveTableLowIdx** .

In diesen Fällen wird intern ein MC\_AxRtFinish\_BkPlcMc [► 230] Baustein aufgerufen und dessen Ausgänge durchgereicht. Andernfalls wird die Tabellenlinearisierung für die Achse durchgeführt. Dabei sind folgende Besonderheiten zu beachten:

- Der Parameter für die Kompensation der Richtungsabhängigkeit (Flächenverhältnis, Gravitation usw.) der Achsgeschwindigkeit ist unwirksam. Diese Kompensation ist in der Tabelle zu berücksichtigen.
- Die Parameter für die Kompensation eines Kennlinienknicks sind unwirksam. Diese Kompensation ist in der Tabelle zu berücksichtigen.
- Der Parameter für die Überdeckungskompensation ist unwirksam. Diese Kompensation ist in der Tabelle zu berücksichtigen.
- Eine Pressdruckausgabe oder eine Offsetkompensation können nicht durch eine Linearisierung realisiert werden. Die entsprechenden Parameter sind aktiv.

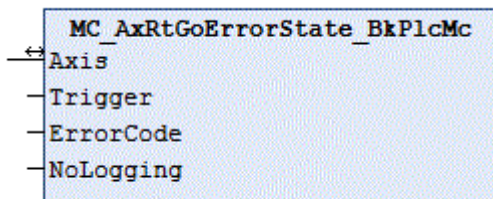
**Beispiel:** Anzeige einer Linearisierung im PlcMcManager:





Ein Programmbeispiel finden Sie in der [SampleList](#) [▶ 353] der [Knowledge Base](#) [▶ 302]. Hier wird auch die automatische Ermittlung einer Kennlinie mit einem [MC\\_AxUtiAutoIdent\\_BkPlcMc](#) [▶ 254] Baustein demonstriert.

#### 4.4.7.5 MC\_AxRtGoErrorState\_BkPlcMc (ab V3.0)



(not recommended) Der Funktionsbaustein versetzt die Achse in einen Störzustand.

#### 🔌 Eingänge

```
VAR_INPUT
    Trigger:      BOOL;
    ErrorID:      UDINT;
    NoLogging:    BOOL;
END_VAR
```

Name	Typ	Beschreibung
Trigger	BOOL	Eine steigende Flanke an diesem Eingang versetzt die Achse in einen Störzustand.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
NoLogging	BOOL	Ein TRUE an diesem Eingang unterdrückt die Ausgabe einer Meldung.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

**Verhalten des Bausteins**

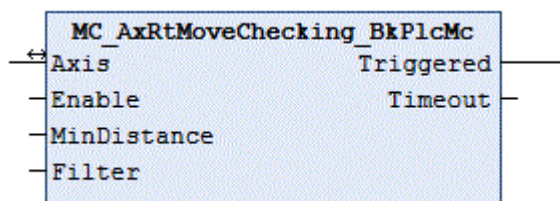
Mit einer steigenden Flanke am Eingang **Trigger** wird die Achse in einen Störzustand versetzt.

Voraussetzungen:

- Der Wert am Eingang **ErrorCode** ist ungleich 0.
- Die Achse befindet sich nicht bereits in einem Störzustand.

**i** Wenn NoLogging auf FALSE steht (Default-Zustand) wird beim Übergang der Achse in den Störzustand eine Meldung erzeugt, die Angaben über die betroffene Achse und den ErrorCode enthält. Diese Default-Meldung sollte unbedingt durch eine aussagefähige Meldung der Applikation ersetzt werden. In diesem Fall ist die Default-Meldung mit NoLogging auf TRUE zu unterdrücken.

**4.4.7.6 MC\_AxRtMoveChecking\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein überwacht die Reaktion einer Achse.

 **Eingänge**

```
VAR_INPUT
  Enable:      BOOL;
  MinDistance: LREAL;
  Filter:      LREAL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein TRUE an diesem Eingang aktiviert die Überwachung.
MinDistance	LREAL	[mm] Hier ist die geforderte Mindeststrecke zu übergeben.
Filter	LREAL	[s] Hier ist die Messzeit anzugeben.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

**Ausgänge**

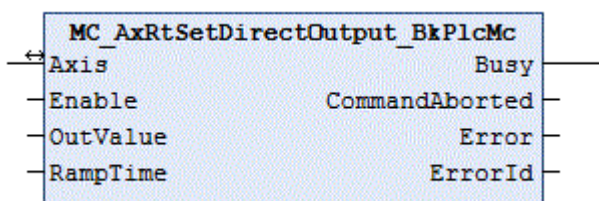
```
VAR_OUTPUT
    Triggered:    BOOL;
    Timeout:      BOOL;
END_VAR
```

Name	Typ	Beschreibung
Triggered	BOOL	Dieser Ausgang signalisiert, dass die Achse in den Fehlerzustand versetzt wurde.
Timeout	BOOL	Dieser Ausgang signalisiert, dass die Überwachung angesprochen hat.

**Verhalten des Bausteins**

Der Baustein überprüft kontinuierlich, ob die Achse innerhalb von **Filter** mindestens einen Weg von **MinDistance** in der zur geforderten Bewegung passenden Richtung zurückgelegt hat. Ist dies nicht der Fall wird **Timeout** signalisiert. Ist **Enable** auf TRUE wird in diesem Fall die Achse in den Stöorzustand **dwTcHydErrCdNotMoving = 16#435D = 17245** versetzt. Dies wird durch **Triggered** signalisiert.

**4.4.7.7 MC\_AxRtSetDirectOutput\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein gibt unabhängig von einer Profilgenerierung einen Stellwert aus.

**Eingänge**

```
VAR_INPUT
    Enable:        BOOL;
    OutValue:      LREAL;
    RampTime:     LREAL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein <b>TRUE</b> an diesem Eingang aktiviert die Ausgabe.
OutValue	LREAL	Hier ist der auszugebende Stellwert zu übergeben.
RampTime	LREAL	[s] Hier ist die Zeit anzugeben in der der Stellwert die Vollaussteuerung erreichen würde.

**Ein-/Ausgänge**

```
VAR_INOUT
    Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
    Busy:          BOOL;
    CommandAborted:  BOOL;
    Error:         BOOL;
    ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird ein Abbruch der Funktion signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

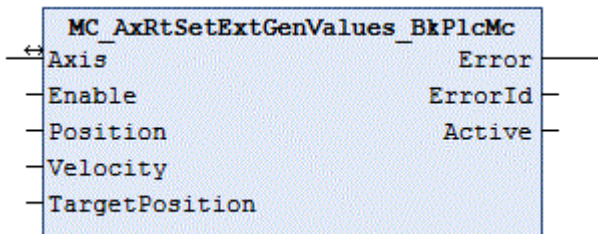
**Verhalten des Bausteins**

Eine steigende Flanke an **Enable** aktiviert die Funktion. Die Achse wird in die Zustände McState\_Continuousmotion [▶ 96] und iTcHydStateExtGenerated [▶ 88] versetzt und **Busy** wird **TRUE**. Der Stellwert der Achse wird mit **OutValue** aktualisiert. Dabei wird die Änderungsrate durch **RampTime** festgelegt.

Wird **Enable** auf **FALSE** gesetzt wird der Stellwert unter Verwendung von **RampTime** auf 0.0 gebracht und die Funktion beendet. Erst dann wird **Busy** **FALSE**.

Wenn die Kontrolle über die Achse durch einen anderen Baustein übernommen wird während der **MC\_AxRtSetDirectOutput\_BkPlcMc** Baustein aktiv ist stellt es seine Funktion ein und signalisiert **CommandAborted**.

**4.4.7.8 MC\_AxRtSetExtGenValues\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein versorgt eine Achse mit Führungsgrößen, die nicht aus dem achseigenen Generator stammen.

**Eingänge**

```
VAR_INPUT
  Enable:          BOOL;
  Position:        LREAL:=0.0;
  Velocity:        LREAL:=0.0;
  TargetPosition:  LREAL:=0.0;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein TRUE an diesem Eingang aktiviert die Übernahme der bereitgestellten Führungsgrößen.
Position	LREAL	[mm] Zyklisch zu übergebender Wert für die Sollposition.
Velocity	LREAL	[mm/s] Zyklisch zu übergebender Wert für die Sollgeschwindigkeit.
TargetPosition	LREAL	[mm] Zyklisch zu übergebender Wert für die Zielposition der aktuellen Bewegung.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [▶ 84] zu übergeben.

**Ausgänge**

```
OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface. Erkennt er eine steigende Flanke an **Execute** versetzt er die Achse in den Zustand **McState\_Synchronizedmotion** und **iTcHydStateExtGenerated**.

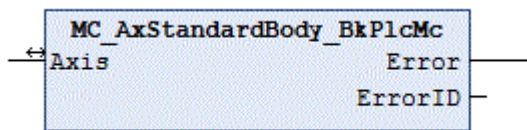
Wenn Execute TRUE ist werden die Werte von **Position**, **Velocity** und **TargetPosition** in die Laufzeitvariablen der Achse eingetragen. Dabei wird soweit möglich das Verhalten des Generatorbaustein bei einer vergleichbaren Bewegung nachgebildet.

Wird eine fallende Flanke an Execute erkannt versetzt der Baustein die Achse in den Zustand **McState\_Standstill**. Ist die Achse zu diesem Zeitpunkt nicht im Stillstand wird sie über die fStopRamp eingestellte zeitgesteuerte Rampe angehalten.



Der Generatorbaustein der Achse ist nach wie vor zyklisch aufzurufen. Er nimmt die Lageregelung vor und aktualisiert weitere interne Variablen.

**4.4.7.9 MC\_AxStandardBody\_BkPlcMc (V3.0)**



Der Funktionsbaustein ruft je einen Baustein der Typen MC\_AxRtEncoder\_BkPlcMc [▶ 183], MC\_AxRuntime\_BkPlcMc [▶ 221], MC\_AxRtFinish\_BkPlcMc [▶ 230] und MC\_AxRtDrive\_BkPlcMc [▶ 172] auf.

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [▶ 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

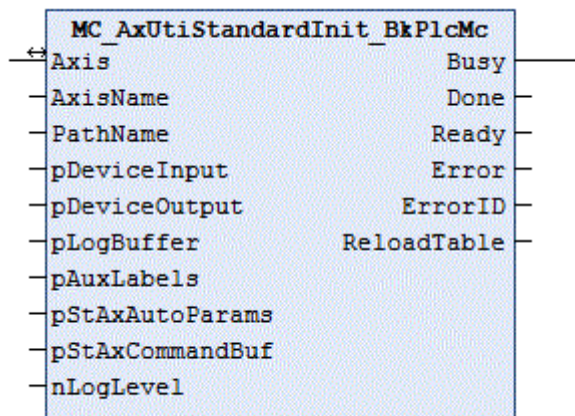
**Verhalten des Bausteins**

Die üblichen Bestandteile der Achse werden in Abhängigkeit der Werte in [ST\\_TcHydAxParam](#) [▶ 120] aufgerufen. Sollte einer der aufgerufenen Bausteine einen **Error** zurückmelden wird dieser mit seinem **ErrorID** an den Ausgängen dieses Bausteins wiedergegeben.



Bei mehreren Problemen gilt eine Priorisierung entsprechend der Aufruffreihenfolge (Encoder, Generator, Finish, Drive).

**4.4.7.10 MC\_AxUtiStandardInit\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein übernimmt die Initialisierung und Überwachung der Bestandteile einer Achse.

**Eingänge**

```
VAR_INPUT
  AxisName:          STRING(255);
  PathName:          STRING(255);
  pDeviceInput:      POINTER TO ST_TcPlcDeviceInput:=0;
  pDeviceOutput:     POINTER TO ST_TcPlcDeviceOutput:=0;
  pLogBuffer:        POINTER TO ST_TcPlcMcLogBuffer:=0;
  pStAxAuxLabels:    POINTER TO ST_TcMcAuxDataLabels:=0;
  pStAxAutoParams:  POINTER TO ST_TcMcAutoIdent;
  pStAxCommandBuf:  POINTER TO ST_TcPlcCmdBuffer_BkPlcMc:=0; (* ab/from V3.0.8 *)
  nLogLevel:         DINT:=0;
END_VAR
```

Name	Typ	Beschreibung
AxisName	STRING	Hier ist der textuelle Name der Achse (Beispiel: 'Achse_1') zu übergeben.
PathName	STRING	Hier ist der textuelle Pfadname (Beispiel: 'C:\TwinCAT\Projekt\') zu übergeben, unter dem die Parameter der Achse zu speichern sind.
pDeviceInput	POINTER	Hier ist die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcDeviceInput</a> [▶ 138] zu übergeben.
pDeviceOutput	POINTER	Hier ist die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcDeviceOutput</a> [▶ 140] zu übergeben.
pLogBuffer	POINTER	Hier kann die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcMcLogBuffer</a> [▶ 142] übergeben werden.
pStAxAuxLabels	POINTER	Hier kann die Adresse einer <a href="#">ST_TcMcAuxDataLabels</a> [▶ 137] Struktur mit Beschriftungstexten der kundenspezifischen Achsparameter übergeben werden.
pStAxAutoParams	POINTER	Hier kann die Adresse einer Variablen vom Typ <a href="#">ST_TcMcAutoIdent</a> [▶ 118] übergeben werden.

Name	Typ	Beschreibung
pStAxCommandBuf	POINTER	Ab V3.0.8 ist bei diversen Bausteinen der durch die PLCopen definierte Eingang <b>BufferMode</b> vorhanden. Die damit steuerbare Funktionalität wird derzeit vorbereitet. In diesem Zusammenhang ist dieser Befehls-Puffer ergänzt worden.  <b>Der Eingang pStAxCommandBuf darf derzeit nicht oder nur mit dem Wert 0 versorgt werden.</b>
nLogLevel	DINT	Hier ist ein <u>codierter Wert</u> [► 328] zu übergeben, der den Schwellwert für die Aufzeichnung von Meldungen festlegt.

 **Ein-/Ausgänge**

```
VAR_OUTPUT
  Busy:      BOOL;
  Done:      BOOL;
  Ready:     BOOL;
  Error:     BOOL;
  ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	
Done	BOOL	
Ready	BOOL	
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

 **Ausgänge**

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [► 84] zu übergeben.

**Verhalten des Bausteins**

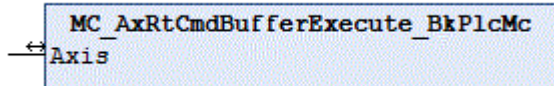
Bei jedem Aufruf untersucht der Baustein das übergebene Achsinterface und die übergebenen Pointer. Wird eine Veränderung erkannt signalisiert der Baustein in der übergebenen Axis Ref BkPlcMc [► 84] Struktur, dass die Achse reinitialisiert werden muss. Der von diesem Baustein verwendete MC AxParamLoad BkPlcMc [► 270] Baustein wird nun automatisch die Parameter der Achse aus der Datei laden. Ist **pAuxLabels** versorgt werden anschließend mit einem MC AxParamAuxLabelsLoad BkPlcMc [► 269] Baustein die Beschriftungstexte der kundenspezifischen Achsparameter geladen.

**i** Die als AxisName und PathName übergebenen Strings dürfen keine Leer- oder Sonderzeichen enthalten, die sie für die Erzeugung eines Dateinamens ungeeignet machen. Der Dateiname wird durch aneinanderhängen der übergebenen Strings und Anfügen der Extension '.dat' erzeugt. Der Dateiname für die Beschriftungstexte der kundenspezifischen Achsparameter wird auf die gleiche Weise, jedoch mit der Extension '.txt' erzeugt.

**i** Die Parameter pDeviceInput und pDeviceOutput sind bei allen Achsen zu versorgen, die für die Positionserfassung eine E/A-Hardware verwenden. Bei Verwendung von virtuellen Achsen sind diese Parameter nicht oder mit 0 zu belegen.

**i** Der Eingang **pStAxCommandBuf** darf derzeit nicht oder nur mit dem Wert 0 versorgt werden.

#### 4.4.7.11 MC\_AxRtCmdBufferExecute\_BkPlcMc



##### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc</a> [► 84] zu übergeben.

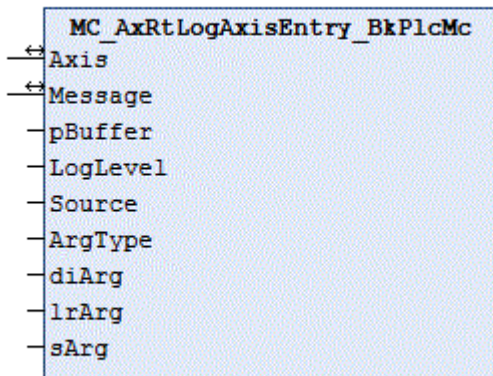
##### Verhalten des Bausteins

Wenn die Achse über einen MC\_AxUtiStandardInit()-Baustein mit einem Kommando-Puffer ausgerüstet ist werden Positionier-Befehle wie MC\_MoveAbsolute\_BkPlcMc in diesem Puffer eingetragen.

Wenn als Sollwert-Generator iTcMc\_ProfileCtrlBased eingestellt ist muss ein Baustein dieses Typs zyklisch aufgerufen werden damit diese Kommandos an die Achse weitergegeben und aktiv abgearbeitet werden.

### 4.4.8 Message Logging

#### 4.4.8.1 MC\_AxRtLogAxisEntry\_BkPlcMc (ab V3.0)



Der Funktionsbaustein trägt eine achsbezogene Meldung in den LogBuffer der Library ein. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der [Knowledge Base](#) [► 302].



Alle achsbezogenen Bausteine der Bibliothek verwenden diesen Baustein zur Meldungsabgabe.

##### Eingänge

```
VAR_INPUT
  pBuffer:      POINTER TO ST_TcPlcMcLogBuffer;
  LogLevel:    DWORD:=0;
  Source:      DWORD:=0;
  Message:    STRING(255);
  ArgType:    INT:=0;
  diArg:      DINT:=0;
```



```

lrArg:      LREAL:=0;
sArg:      STRING(255);
END_VAR
    
```

Name	Typ	Beschreibung
pBuffer	POINTER	Hier ist die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcMcLogBuffer [► 142]</a> zu übergeben.
LogLevel	DWORD	Eine kodierte Angabe des Meldungstyps. Es ist ein <a href="#">Logger Levels [► 328]</a> Wert aus den <a href="#">Global Constants [► 319]</a> zu verwenden.
Source	DWORD	Eine kodierte Angabe der Meldungsquelle. Es ist ein <a href="#">Logger Sources [► 328]</a> Wert aus den <a href="#">Global Constants [► 319]</a> zu verwenden.
Message	STRING	Der Text der Meldung.
ArgType	INT	Der Typ des optionalen Argumentes.
diArg	DINT	Der Wert des optionalen Arguments, wenn es vom Typ DINT ist.
lrArg	LREAL	Der Wert des optionalen Arguments, wenn es vom Typ LREAL ist.
sArg	STRING	Der Wert des optionalen Arguments, wenn es vom Typ STRING ist.

 **Ein-/Ausgänge**

```

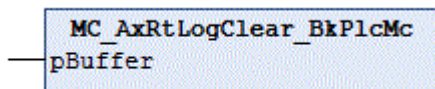
VAR_INOUT
  Axis:      POINTER TO Axis_Ref_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Axis	POINTER	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [► 84]</a> zu übergeben

**Verhalten des Bausteins**

Der Baustein unterscheidet sich von [MC\\_AxRtLogEntry\\_BkPlcMc \[► 242\]](#) nur dadurch, dass er der Meldung den Achsnamen voranstellt.

**4.4.8.2 MC\_AxRtLogClear\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein löscht einen LogBuffer der Library. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der [Knowledge Base \[► 302\]](#).

 **Ein-/Ausgänge**

```

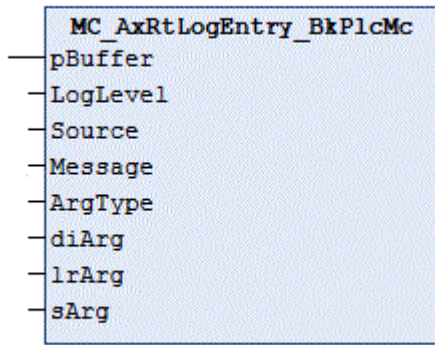
VAR_INOUT
  pBuffer:      POINTER TO ST_TcPlcMcLogBuffer;
END_VAR
    
```

Name	Typ	Beschreibung
pBuffer	POINTER	<b>pBuffer:</b> Hier ist die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcMcLogBuffer [► 142]</a> zu übergeben.

**Verhalten des Bausteins**

Alle Einträge im LogBuffer werden gelöscht und initialisiert.

### 4.4.8.3 MC\_AxRtLogEntry\_BkPlcMc (ab V3.0)



Der Funktionsbaustein trägt eine Meldung in den LogBuffer der Library ein. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der [Knowledge Base](#) [▶ 302].

#### Eingänge

```

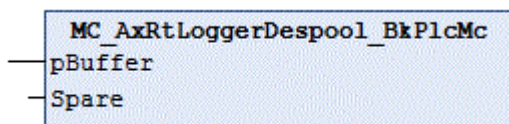
VAR_INPUT
  pBuffer:      POINTER TO ST_TcPlcMcLogBuffer;
  LogLevel:     DWORD:=0;
  Source:       DWORD:=0;
  Message:      STRING(255);
  ArgType:      INT:=0;
  diArg:        DINT:=0;
  lrArg:        LREAL:=0;
  sArg:         STRING(255);
END_VAR
    
```

Name	Typ	Beschreibung
pBuffer	POINTER	Hier ist die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcMcLogBuffer</a> [▶ 142] zu übergeben.
LogLevel	DWORD	Eine kodierte Angabe des Meldungstyps. Es ist ein <a href="#">Logger Levels</a> [▶ 328] Wert aus den <a href="#">Global Constants</a> [▶ 319] zu verwenden.
Source	DWORD	Eine kodierte Angabe der Meldungsquelle. Es ist ein <a href="#">Logger Sources</a> [▶ 328] Wert aus den <a href="#">Global Constants</a> [▶ 319] zu verwenden.
Message	STRING	Der Text der Meldung.
ArgType	INT	Der Typ des optionalen Argumentes.
diArg	DINT	Der Wert des optionalen Arguments, wenn es vom Typ DINT ist.
lrArg	LREAL	Der Wert des optionalen Arguments, wenn es vom Typ LREAL ist.
sArg	STRING	Der Wert des optionalen Arguments, wenn es vom Typ STRING ist.

#### Verhalten des Bausteins

Wenn pBuffer korrekt versorgt wurde und auf einen [ST\\_TcPlcMcLogBuffer](#) [▶ 142] zeigt, der noch mindestens eine Meldung aufnehmen kann werden die übergebenen Meldungsdaten mit Informationen zur lokalen Zeit komplettiert und in den Meldungspuffer eingetragen.

### 4.4.8.4 MC\_AxRtLoggerDespool\_BkPlcMc (ab V3.0)



Der Funktionsbaustein stellt sicher, dass der LogBuffer der Library nicht überläuft. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der [Knowledge Base \[▶ 302\]](#).

 **Eingänge**

```
VAR_INPUT
  Spare:      INT;
END_VAR
```

Name	Typ	Beschreibung
Spare	INT	Die sicher zu stellende Anzahl von freien Meldungen im LogBuffer.

 **Ein-/Ausgänge**

```
VAR_INOUT
  pBuffer:    POINTER TO ST_TcPlcMcLogBuffer;
END_VAR
```

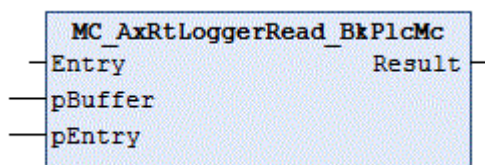
Name	Typ	Beschreibung
pBuffer	POINTER	Hier ist die Adresse einer Variablen vom Typ ST_TcPlcMcLogBuffer [ <a href="#">▶ 142</a> ] zu übergeben.

**Verhalten des Bausteins**

Bei jedem Aufruf entfernt der Baustein eine Meldung aus dem LogBuffer wenn die Anzahl der freien Meldungen kleiner ist als die in **Spare** übergebene Minimalanzahl. Soll die gesamte Historie in die Ereignisanzeige von Windows übernommen werden ist ein [MC\\_AxRtLoggerSpool\\_BkPlcMc \[▶ 244\]](#) Baustein zu verwenden.

**i** Der Einsatz dieses Bausteins ist immer dann sinnvoll, wenn ein Massenspeicher mit eingeschränkter Beschreibbarkeit (typischerweise FLASH DISK) verwendet wird. Es wird zumindest eine eingeschränkte Historie von 10 bis 15 Meldungen ermöglicht.

**4.4.8.5 MC\_AxRtLoggerRead\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein liest eine Meldung aus dem LogBuffer der Library. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der [Knowledge Base \[▶ 302\]](#).

**i** Dieser Baustein wird via ADS von Diagnose-Tools genutzt. Ein direkter Aufruf durch die SPS-Applikation macht in der Regel keinen Sinn.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Entry:      INT:=0;
  pBuffer:    POINTER TO ST_TcPlcMcLogBuffer;
  pEntry:     POINTER TO ST_TcPlcMcLogEntry;
END_VAR
```

Name	Typ	Beschreibung
Entry	INT	Die Nummer der zu lesenden Meldung,

Name	Typ	Beschreibung
pBuffer	POINTER	Hier ist die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcMcLogBuffer [▶ 142]</a> zu übergeben.
pEntry	POINTER	Hier ist als Ziel die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcMcLogEntry [▶ 142]</a> zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
    Result:      DWORD:=0;
END_VAR
```

Name	Typ	Beschreibung
Result	DWORD	Hier wird ein eine codierte Fehlerursache bereitgestellt.

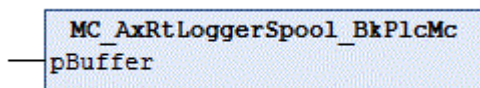
**Verhalten des Bausteins**

Der Baustein überprüft bei jedem Aufruf die übergebenen Eingangswerte. Dabei können zwei Probleme erkannt werden:

- Ist **Entry** nicht im zulässigen Wertebereich 1..20 gibt der Baustein `dwTcHydAdsErrInvalidIdxOffset` als **Result** zurück.
- Sind **pBuffer** oder **pEntry** nicht definiert gibt der Baustein `dwTcHydAdsErrNotReady` als **Result** zurück.

Ist bei der Überprüfung kein Problem festgestellt worden kopiert der Baustein die von **Entry** selektierte Meldung aus dem LogBuffer **pBuffer** in die mit **pEntry** adressierte Meldungsstruktur. Dabei wird Entry als relative Altersangabe verstanden: Mit Entry:=1 wird die zuletzt eingetragene Meldung ausgewählt, mit Entry:=2 die nächst ältere usw. Steht die geforderte Meldung nicht zur Verfügung wird eine leere Meldung bereitgestellt.

**4.4.8.6 MC\_AxRtLoggerSpool\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein übernimmt die Übertragung von Meldungen aus dem LogBuffer der Library in die Ereignisanzeige von Windows. Näheres über das Anlegen eines LogBuffers finden Sie unter FAQ #10 in der Knowledge Base [\[▶ 302\]](#).

 **Ein-/Ausgänge**

```
VAR_INOUT
    pBuffer:      POINTER TO ST_TcPlcMcLogBuffer;
END_VAR
```

Name	Typ	Beschreibung
pBuffer	POINTER	Hier ist die Adresse einer Variablen vom Typ <a href="#">ST_TcPlcMcLogBuffer [▶ 142]</a> zu übergeben. Beschreibung

**Verhalten des Bausteins**

Bei jedem Aufruf entfernt der Baustein eine Meldung aus dem LogBuffer und überträgt sie in die Ereignisanzeige von Windows.

Wenn der eingesetzte Rechner einen Massenspeicher mit eingeschränkter Schreibbarkeit verwendet (typischerweise FLASH DISK) ist es nicht sinnvoll, die Ereignisanzeige von Windows zu nutzen. Um trotzdem eine Historie zu erzeugen kann ein [MC\\_AxRtLoggerDespool\\_BkPlcMc \[▶ 242\]](#) Baustein verwendet werden.

## 4.4.9 Utilitys

### 4.4.9.1 MC\_AxParamDelayedSave\_BkPlcMc



Der Funktionsbaustein löst ein zeitverzögertes Schreiben der Parameter-Datei nach einer Änderung eines Parameters aus.

#### Eingänge

```
VAR_INPUT
    Delay:          LREAL:=0.0;
END_VAR
```

Name	Typ	Beschreibung
Delay	LREAL	[s] Verzögerung beim Auslösen der Parametersicherung.

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    Error:         BOOL;
    ErrorID:       UDINT;
END_VAR
```

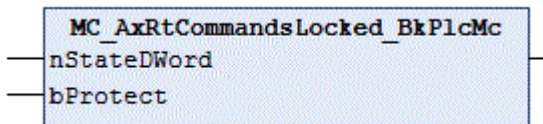
Name	Typ	Beschreibung
Error	BOOL	Dieser Ausgang signalisiert aufgetretene Probleme.
ErrorID	UDINT	Im Fehlerfall wird hier eine codierte Information über die Art des Problems gemeldet.

#### Verhalten des Bausteins

Wenn einer der unten aufgeführten Bausteine einen Parameter erkennbar verändert, wird eine Zeit zurückgesetzt. In jedem Zyklus, in dem dies nicht der Fall ist, wird diese Zeit um die Zykluszeit erhöht. Erreicht sie den Parameter Delay wird das Schreiben der Parameter-Datei ausgelöst.

- MC\_WriteBoolParameter\_BkPlcMc
- MC\_WriteParameter\_BkPlcMc
- MC\_AxUtiAutoident\_BkPlcMc
- MC\_LinTableImportFromAsciiFile\_BkPlcMc
- MC\_LinTableImportFromBinFile\_BkPlcMc

### 4.4.9.2 MC\_AxRtCommandsLocked\_BkPlcMc : DWORD



Die Funktion erleichtert das Setzen und Löschen einer Schutzfunktion im Status-Doppelwort einer Achse.

#### Eingänge

```
VAR_INPUT
    nStateDWord:  DWORD:=0;
    bProtect:     BOOL:=FALSE;
END_VAR
```

Name	Typ	Beschreibung
nStateDWord	DWORD	Der aktuelle Zustand des Status-Doppelworts.
bProtect	BOOL	Der gewünschte Zustand der Schutzfunktion.

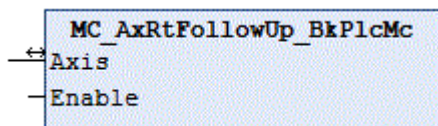
#### Verhalten der Funktion

Die Funktion blendet abhängig von **bProtect** das Status-Bit der Schutzfunktion in das übergebene Status-Doppelwort ein.

**i** Das Ergebnis der Funktion muss von der Applikation auf das Status-Doppelwort der Achse zugewiesen werden.

Es steht ein [Beispiel](#) [▶ 357] zur Verfügung.

### 4.4.9.3 MC\_AxRtFollowUp\_BkPlcMc



Der Funktionsbaustein aktualisiert die Offsetkompensation.

#### Eingänge

```
VAR_INPUT
    Enable:  BOOL;
END_VAR
```

Name	Typ	Beschreibung
Enable	BOOL	Ein TRUE an diesem Eingang aktiviert den Baustein.

#### Ein-/Ausgänge

```
VAR_IN_OUT
    Axis:  AXIS_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	AXIS_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis Ref BkPlcMc [▶ 84] zu übergeben.

**Verhalten des Bausteins**

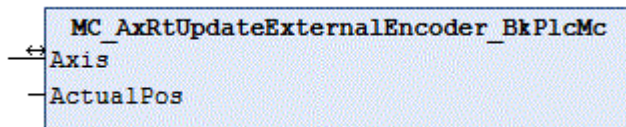
Wenn Enable = TRUE ist wird die Istposition auf alle Ziel- und Sollpositionen kopiert. Sollgeschwindigkeit, Schleppabstand und Lageregler-Ausgabe werden auf 0.0 gesetzt. Bei einer fallenden Flanke an Enable wird die Achse abhängig von Parametern und Freigaben an der aktuellen Istposition die Lageregelung reaktivieren.

Der Baustein sollte nicht für ein Achse aktiviert werden, die eine aktive Bewegung ausführt oder geregelt werden muss.

Da die Lageregelung inaktiv ist kann die Achse driften.

Soll die Achse durch äußere Wirkungen bewegt werden, sind die dafür benötigten Ölwege durch die Applikation zu öffnen.

**4.4.9.4 MC\_AxRtUpdateExternalEncoder\_BkPlcMc**



Der Funktionsbaustein aktualisiert die Istposition einer Achse mit einem durch die Applikation ermittelten Wert.

**Eingänge**

```
VAR_INPUT
    ActualPos: LREAL;
END_VAR
```

Name	Typ	Beschreibung
ActualPos	LREAL	Der neue Wert für die Istposition.

**Ein-/Ausgänge**

```
VAR_IN_OUT
    Axis: AXIS_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	AXIS_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [► 84] zu übergeben.

**Verhalten des Bausteins**

ActualPos wird als Istposition in die Laufzeitdaten der Achse eingetragen. Als Istgeschwindigkeit wird die dadurch erzeugte Änderung bezogen auf die Zykluszeit der Achse verwendet. In stAxRtData wird bEncoderResponse gesetzt, um eine erfolgte Aktualisierung zu markieren.

**HINWEIS**

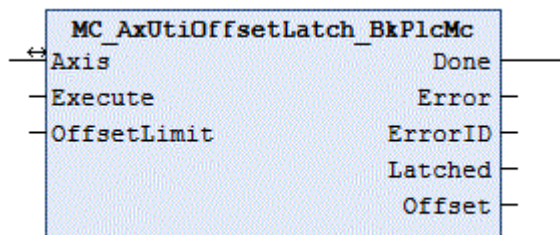
Wenn dieser Baustein genutzt wird darf kein MC\_AxRtEncoder\_BkPlcMc() oder MC\_AxStandardBody\_BkPlcMc() Baustein aufgerufen werden.

**HINWEIS**

Die Aktualisierung der Istposition muss zwingend in jedem Zyklus erfolgen.

**i** Der Baustein nimmt keine Filterung vor.

### 4.4.9.5 MC\_AxUtiOffsetLatch\_BkPlcMc (ab V3.0.40)



Der Funktionsbaustein aktualisiert die Offsetkompensation.

#### Eingänge

```
VAR_INPUT
    Execute:      BOOL;
    OffsetLimit:  LREAL:=0.5;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke löst die Aktualisierung aus.
OffsetLimit	LREAL	[V] Der maximal zulässige Wertebereich der Offsetkompensation.

#### Ein-/Ausgänge

```
VAR_IN_OUT
    Axis:      AXIS_REF_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	AXIS_REF_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    Done:      BOOL;
    Error:     BOOL;
    ErrorId:   UDINT;
    Latched:   BOOL;
    Offset:    LREAL;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Aktualisierung signalisiert.
Error	BOOL	Dieser Ausgang signalisiert aufgetretene Probleme.
ErrorId	UDINT	Im Fehlerfall wird hier eine codierte Information über die Art des Problems gemeldet.
Latched	BOOL	Dieser Ausgang signalisiert die erfolgreiche Durchführung der Aktualisierung.
Offset	LREAL	[V] Dieser Ausgang meldet den Offsetwert. Er wird nur bei <b>Done</b> als neue Kompensation übernommen.

#### Verhalten des Bausteins

Bei einer steigenden Flanke an **Execute** wird **Offset** mit der momentanen Ausgabe des Lagereglers aktualisiert.

Bevor dieser Wert als Kompensation übernommen wird, überprüft der Baustein mehrere Fehlermöglichkeiten:

- Die Achse muss eine Reglerfreigabe haben und darf nicht in einem aktiven Fahrzustand oder Fehlerzustand sein (Axis.stAxRtData.iCurrentStep=iTcHydStatIdle). (Fehler-Code dwTcHydAdsErrBusy)



- Die festgestellte Reglerausgabe darf nicht außerhalb von  $\pm$ OffsetLimit liegen. (Fehler-Code dwTcHydAdsErrIllegalValue)

Ist einer der Fehler aufgetreten, wird **Error** gemeldet und **ErrorId** mit dem genannten Fehler-Code belegt. In diesem Fall bleibt die Kompensation unverändert.

Andernfalls wird **Offset** als neuer Kompensationswert übernommen. Da ab diesem Zeitpunkt der vorher vom Lageregler bereitgestellte Anteil des Ausgabewerts von der Kompensation übernommen wird, muss die Ausgabe des Reglers aufgehoben werden. Hierzu werden einmalig folgende Schritte durchgeführt:

- Soll- und momentane Zielposition werden mit der Istposition aktualisiert.
- Der Positionsfehler (Schleppabstand) wird zu Null gesetzt.
- Die Lagereglerausgabe wird zu Null gesetzt.
- Der I-Anteil des Default-Lagereglers wird abgelöscht.
- Sollte ein anderer als der Default-Lageregler genutzt werden ist dessen I-Anteil durch die Applikation zu löschen.

Bei einer fallenden Flanke an **Execute** werden alle Ausgänge in den Ruhezustand gesetzt.

### 4.4.9.6 Filters

#### 4.4.9.6.1 MC\_AxUtiAverageDerivative\_BkPlcMc (ab V3.0)



Der Funktionsbaustein ermittelt die Ableitung eines Wertes durch numerische Differentiation über mehr als einen Zyklus.

#### Eingänge

```
VAR_INPUT
    Input:      LREAL:=0.0;
    MinIdx:     DINT:=0;
    MaxIdx:     DINT:=0;
    pBuffer:    POINTER TO LREAL:=0;
END_VAR
```

Name	Typ	Beschreibung
Input	LREAL	Der Roh-Wert der zu filternden Größe.
MinIdx	DINT	Der Index des ersten zu verwendenden Elements des Filterpuffers.
MaxIdx	DINT	Der Index des letzten zu verwendenden Elements des Filterpuffers.
pBuffer	POINTER	Die Adresse des ersten Elements des Filterpuffers.

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Output:      LREAL:=0.0;
  Error:       BOOL:=FALSE;
  ErrorID:     UDINT:=0;
END_VAR
```

Name	Typ	Beschreibung
Output	LREAL	Der gefilterte Wert.
Error	BOOL	Dieser Ausgang signalisiert Probleme mit den übergebenen Parametern.
ErrorID	UDINT	Im Fehlerfall wird hier eine kodierte Information über die Art des Problems gemeldet.

**Verhalten des Bausteins**

Bei jedem Aufruf überprüft der Baustein die Adresse des Filterpuffers **pBuffer** und die Indices der zu verwendenden Elemente **MinIdx** und **MaxIdx**. Sind die übergebenen Werte erkennbar unsinnig wird mit **Error** und einer kodierten Information in ErrorID reagiert. Andernfalls wird bei jedem Aufruf **Input** in den Filterpuffer eingetragen und der Mittelwert der Änderung über den im Puffer verfügbaren Wertevorrat gebildet und als **Output** zurückgegeben.

- i** Der Wertevorrat für die Mittelwertbildung umfasst (MaxIdx - MinIdx + 1) Werte. Die Messzeit ergibt sich durch Multiplikation dieser Anzahl mit der Zykluszeit.

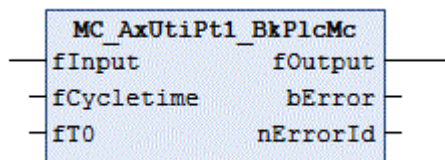
---

- i** Das Prinzip der gleitenden Mittelwertbildung führt zu einer Verzögerung in Höhe der halben Messzeit. Wird die gefilterte Größe in einem Regelkreis verwendet kann die dadurch verursachte frequenzabhängige Phasenverschiebung Einschränkungen bei der Parameterwahl verursachen.

---

- i** Der Baustein hat keine Möglichkeit, die übergebenen Werte von pBuffer, MinIdx und MaxIdx vollständig zu überprüfen. Es ist besonders darauf zu achten, dass diese Werte gefahrlos verwendet werden können. Andernfalls kann es zu nicht vorhersagbarem Verhalten (Überschreiben von Speicher) oder zum Abbrechen des PLC-Betriebs kommen.

**4.4.9.6.2 MC\_AxUtiPT1\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein berechnet einen Tiefpass 1. Ordnung.

**Eingänge**

```
VAR_INPUT
  fInput:      LREAL:=0.0;
  fCycletime:  LREAL:=0.001;
  fT0:         LREAL:=1.0;
END_VAR
```

Name	Typ	Beschreibung
fInput	LREAL	Der Roh-Wert der zu filternden Größe.
fCycletime	LREAL	[s] Die Zykluszeit der aufrufenden PLC-Task.
fT0	LREAL	[s] Die Filterzeitkonstante.

**Ausgänge**

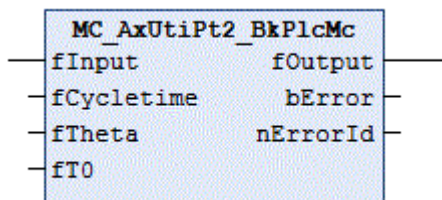
```
VAR_OUTPUT
  fOutput:    LREAL;
  bError:     BOOL;
  nErrorId:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
fOutput	LREAL	Der gefilterte Wert.
bError	BOOL	Dieser Ausgang signalisiert Probleme mit den übergebenen Parametern.
nErrorId	UDINT	Im Fehlerfall wird hier eine codierte Information über die Art des Problems gemeldet.

**Verhalten des Bausteins**

Bei jedem Aufruf überprüft der Baustein die übergebenen Parameter. Sollte ein unzulässiger Wert erkannt werden reagiert der Baustein mit **bError** und einem entsprechenden Wert in **nErrorId**. Andernfalls werden die internen Variablen mit **fInput** aktualisiert und der gefilterte Wert als **fOutput** zurückgegeben.

**4.4.9.6.3 MC\_AxUtiPT2\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein berechnet einen Tiefpass 2. Ordnung.

**Eingänge**

```
VAR_INPUT
  fInput:      LREAL:=0.0;
  fCycletime:  LREAL:=0.001;
  fTheta:      LREAL:=1.0;
  fT0:         LREAL:=1.0;
END_VAR
```

Name	Typ	Beschreibung
fInput	LREAL	Der Roh-Wert der zu filternden Größe.
fCycletime	LREAL	[s] Die Zykluszeit der aufrufenden PLC-Task.
fTheta	LREAL	Die Dämpfung.
fT0	LREAL	Die Filterzeitkonstante.

**Ausgänge**

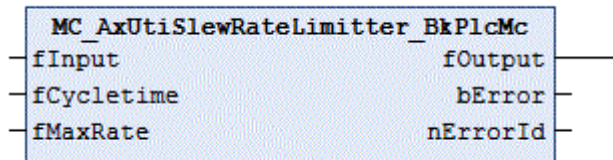
```
VAR_OUTPUT
  fOutput:    LREAL;
  bError:     BOOL;
  nErrorId:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
fOutput	LREAL	Der gefilterte Wert.
bError	BOOL	Dieser Ausgang signalisiert Probleme mit den übergebenen Parametern.
nErrorId	UDINT	Im Fehlerfall wird hier eine codierte Information über die Art des Problems gemeldet.

### Verhalten des Bausteins

Bei jedem Aufruf überprüft der Baustein die übergebenen Parameter. Sollte ein unzulässiger Wert erkannt werden reagiert der Baustein mit **bError** und einem entsprechenden Wert in **nErrorId**. Andernfalls werden die internen Variablen mit **fInput** aktualisiert und der gefilterte Wert als **fOutput** zurückgegeben.

#### 4.4.9.6.4 MC\_AxUtiSlewRateLimiter\_BkPlcMc (ab V3.0)



Der Funktionsbaustein erzeugt eine anstiegsbegrenzte Rampe.

#### Eingänge

```
VAR_INPUT
    fInput:      LREAL:=0.0;
    fCycletime:  DINT:=0;
    fMaxRate:    DINT:=0;
END_VAR
```

Name	Typ	Beschreibung
fInput	LREAL	Der Roh-Wert der zu filternden Größe.
fCycletime	DINT	[s] Die Zykluszeit der aufrufenden PLC Task in Sekunden.
fMaxRate	DINT	Der Betrag der am Ausgang maximal zulässigen Änderungsrate als Änderung pro Sekunde.

#### Ausgänge

```
VAR_OUTPUT
    fOutput:      LREAL:=0.0;
    bError:       BOOL:=FALSE;
    nErrorId:     UDINT:=0;
END_VAR
```

Name	Typ	Beschreibung
fOutput	LREAL	[1/s] Der gefilterte Wert.
bError	BOOL	Dieser Ausgang signalisiert Probleme mit den übergebenen Parametern.
nErrorId	UDINT	Im Fehlerfall wird hier eine codierte Fehlerinformation ausgegeben.

### Verhalten des Bausteins

Bei jedem Aufruf überprüft der Baustein die übergebenen Werte für **fCycletime** und **fMaxRate**. Sind die Werte nicht korrekt wird mit **bError** und einer codierten Information in **nErrorId** reagiert. Andernfalls wird bei jedem Aufruf die Differenz zwischen **Input** und **Output** ermittelt. Ist der Betrag dieser Differenz kleiner oder gleich **fMaxRate \* fCycletime** wird der Wert von **Input** unmittelbar als **fOutput** übernommen. Andernfalls wird **fOutput** um **fMaxRate \* fCycletime** geändert. Dabei wird das Vorzeichen automatisch gewählt.



Der Wert für fCycletime muss  $\geq 0.001$  sein. Für fMaxRate sind negative Werte nicht zulässig.

In der Regel wird es sich bei **Input** um einen Wert handeln, der im Takt der Achsbausteine ermittelt und gefiltert wird. Hier kann für **fCycletime** [Axis\\_Ref\\_BkPlcMc \[► 84\].ST\\_TcHydAxParam \[► 120\].fCycletime](#) verwendet werden.

4.4.9.6.5 MC\_AxUtiSlidingAverage\_BkPlcMc (ab V3.0)



Der Funktionsbaustein ermittelt einen gleitenden Mittelwert.

**Eingänge**

```
VAR_INPUT
  Input:      LREAL:=0.0;
  MinIdx:     DINT:=0;
  MaxIdx:     DINT:=0;
  pBuffer:    POINTER TO LREAL:=0;
END_VAR
```

Name	Typ	Beschreibung
Input	LREAL	Der Roh-Wert der zu filternden Größe.
MinIdx	DINT	Der Index des ersten zu verwendenden Elements des Filterpuffers.
MaxIdx	DINT	Der Index des letzten zu verwendenden Elements des Filterpuffers.
pBuffer	POINTER	Die Adresse des ersten Elements des Filterpuffers.

**Ausgänge**

```
VAR_OUTPUT
  Output:     LREAL:=0.0;
END_VAR
```

Name	Typ	Beschreibung
Output	LREAL	Output: Der gefilterte Wert.

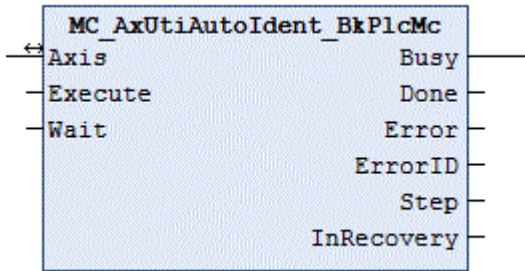
**Verhalten des Bausteins**

Bei jedem Aufruf überprüft der Baustein die Adresse des Filterpuffers **pBuffer** und die Indices der zu verwendenden Elemente **MinIdx** und **MaxIdx**. Sind die übergebenen Werte erkennbar unsinnig wird **Input** als **Output** ausgegeben. Andernfalls wird bei jedem Aufruf **Input** in den Filterpuffer eingetragen und der Mittelwert über den im Puffer verfügbaren Wertevorrat gebildet und als **Output** zurückgegeben.

- i** Der Wertevorrat für die Mittelwertbildung umfasst (**MaxIdx - MinIdx + 1**) Werte. Die Filterzeit ergibt sich durch Multiplikation dieser Anzahl mit der Zykluszeit.
- i** Das Prinzip der gleitenden Mittelwertbildung führt zu einer Verzögerung in Höhe der halben Filterzeit. Wird die gefilterte Größe in einem Regelkreis verwendet kann die dadurch verursachte frequenzabhängige Phasenverschiebung Einschränkungen bei der Parameterwahl verursachen.
- i** Der Baustein hat keine Möglichkeit, die übergebenen Werte von pBuffer, MinIdx und MaxIdx vollständig zu überprüfen. Es ist besonders darauf zu achten, dass diese Werte gefahrlos verwendet werden können. Andernfalls kann es zu nicht vorhersagbarem Verhalten (Überschreiben von Speicher) oder zum Abbrechen des PLC-Betriebs kommen.

### 4.4.9.7 Identification

#### 4.4.9.7.1 MC\_AxUtiAutoIdent\_BkPlcMc (ab V3.0.28)



Der Funktionsbaustein ermittelt automatisch eine Reihe von Parametern der Achse.

#### Eingänge

```
VAR_INPUT
    Execute: BOOL;
    Wait: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang löst die Identifikation aus.
Wait	BOOL	(Ab TC2 V3.0.44 / TC3 V3.3.1.22) Wenn dieser Eingang auf TRUE gesetzt wird verlässt das interne Schrittschaltwerk die Erholungszeit nicht. Das Hochrampen des Ausgabewerts bei noch nicht geschalteten Ventilen kann so verhindert werden.

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis: Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    Busy: BOOL;
    Done: BOOL;
    Error: BOOL;
    ErrorID: UDINT;
    Step: INT;
    InRecovery: BOOL;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung der Identifikation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
Step	INT	Der aktuelle Schritt des internen Schrittschaltwerks.
InRecovery	BOOL	(Ab TC2 V3.0.44 / TC3 V3.3.1.22) Hier wird signalisiert, dass eine Erholungszeit (festgelegt durch ValveCharacteristicRecovery) abläuft.

## Verhalten des Bausteins

Der Baustein überprüft, ob der Pointer [Axis\\_Ref\\_BkPlcMc \[▶ 84\]](#).pStAxAutoParams initialisiert wurde. Ist dies der Fall führt er bei einer steigenden Flanke am **Execute** Eingang eine Reihe von Initialisierungen aus und beginnt dann mit der Parameter-Identifikation. Die einzelnen Schritte der Identifikation werden durch die Werte in [ST\\_TcMcAutolIdent \[▶ 118\]](#) festgelegt.

**EnableEndOfTravel:** Wenn dieser boolesche Parameter gesetzt ist werden die mechanischen Fahrwegsgrenzen automatisch ermittelt. Hierzu wird zunächst sichergestellt, dass die Achse frei beweglich oder am positiven Block ist. Jetzt wird die Achse mit einer negativen Steuerspannung gefahren bis sie den Block erreicht hat. Anschließend wird die Achse mit positiver Steuerspannung betrieben bis der positive Block erkannt ist. Die Steuerspannung wird auf **EndOfTravel\_NegativLimit** und **EndOfTravel\_PositivLimit** begrenzt. Sollte die positive Fahrwegsgrenze kleiner sein als die negative werden die Werte getauscht und `Axis.stAxParams.bDrive_Reversed` invertiert.

**EnableOverlap, EnableZeroAdjust:** Wenn einer dieser booleschen Parameter gesetzt ist wird die Überdeckung bzw. die Offsetspannung des Ventils ermittelt.



Diese Operation wird durch `EndOfTravel_Negativ` und `EndOfTravel_Positiv` beeinflusst.

---

Zunächst wird die Achse in ein Positionsfenster bewegt, dass sich in der Mitte zwischen **EndOfTravel\_Positiv** und **EndOfTravel\_Negativ** befindet. Die Breite des Fensters ist 80% des durch diese Parameter festgelegten Bereichs. Dabei wird bei Bedarf die Ausgabepolarität des Drives invertiert. Jetzt wird die Ausgabespannung ermittelt, bei der die Achse beginnt, sich in positiver Richtung zu bewegen. Anschließend wird die entsprechende negative Spannung ermittelt. Durch Verrechnung dieser Parameter wird sowohl die Überdeckung als auch die Offsetspannung ermittelt. Die Art des Eintrags in die Achsparameter wird durch **EnableOverlap** und **EnableZeroAdjust** gesteuert.

**EnableArreaRatio:** Wenn dieser boolesche Parameter gesetzt ist wird das richtungsabhängige Geschwindigkeitsverhältnis ermittelt. Dazu wird die Achse zunächst in ein Positionsfenster bewegt, dass sich in der Mitte zwischen `pStAxAutoParams.EndOfTravel_Positiv` und `pStAxAutoParams.EndOfTravel_Negativ` befindet. Die Breite des Fensters ist 80% des durch diese Parameter festgelegten Bereichs. Anschließend wird in positiver und negativer Richtung für eine Sekunde mit einer Steuerspannung von 1Volt gefahren. Die dabei ermittelten Geschwindigkeiten werden zur Ermittlung des Geschwindigkeitsverhältnisses dividiert.

**EndOfTravel\_Negativ:** [mm] Wenn die Ermittlung der Fahrwegsgrenzen aktiviert ist wird dieser Wert vom Baustein ermittelt. Ist sie deaktiviert muss hier die Vorgabe durch die Applikation gemacht werden.



Dieser Parameter beeinflusst die Ermittlung von Offsetspannung und Flächenverhältnis.

---

**EndOfTravel\_Positiv:** [mm] Wenn die Ermittlung der Fahrwegsgrenzen aktiviert ist wird dieser Wert vom Baustein ermittelt. Ist sie deaktiviert muss hier die Vorgabe durch die Applikation gemacht werden.



Dieser Parameter beeinflusst die Ermittlung von Offsetspannung und Flächenverhältnis.

---

**EndOfIncrements\_Negativ:** [1] Wenn die Ermittlung der Fahrwegsgrenzen aktiviert ist wird dieser Wert vom Baustein ermittelt. Er entspricht dann **EndOfTravel\_Negativ**, ist aber der Encoder-Rohwert in Inkrementen.

**EndOfIncrements\_Positiv:** [1] Wenn die Ermittlung der Fahrwegsgrenzen aktiviert ist wird dieser Wert vom Baustein ermittelt. Er entspricht dann **EndOfTravel\_Positiv**, ist aber der Encoder-Rohwert in Inkrementen.

**EndOfTravel\_NegativLimit:** [V] Dieser Parameter begrenzt negative Ausgangsspannungen.

**EndOfTravel\_PositivLimit:** [V] Dieser Parameter begrenzt positive Ausgangsspannungen.

**EndOfTravel\_PositivDone:** Dieses Signal wird vom Baustein gesetzt, wenn die Ermittlung der Fahrwegsgrenzen deaktiviert ist oder die positive Fahrwegsgrenze ermittelt wurde.

**EndOfTravel\_NegativDone:** Dieses Signal wird vom Baustein gesetzt, wenn die Ermittlung der Fahrwegsgrenzen deaktiviert ist oder die negative Fahrwegsgrenze ermittelt wurde.

**EndOfVelocity\_NegativLimit:** [mm/s] Dieser Parameter begrenzt negative Geschwindigkeiten. Ist diese Geschwindigkeit bei der Vermessung erreicht oder überschritten wird die aktuelle Messung zu Ende geführt, aber keine weitere Messung in dieser Richtung vorgenommen.

**EndOfVelocity\_PositivLimit:** [mm/s] Dieser Parameter begrenzt positive Geschwindigkeiten. Ist diese Geschwindigkeit bei der Vermessung erreicht oder überschritten wird die aktuelle Messung zu Ende geführt, aber keine weitere Messung in dieser Richtung vorgenommen.

**DecelerationFactor:** [1] Nach dem Messhub wird die Achse für den nächsten Messhub zum Ende des Messwegs bewegt. Dabei werden die mit diesem Faktor gewichteten regulären Achsparameter **fMaxAcc** und **fMaxDec** verwendet.

**EnableValveCharacteristic:** Wenn dieser boolesche Parameter gesetzt ist wird die Geschwindigkeitskennlinie automatisch ermittelt.

**ValveCharacteristicTable:** Dieses ARRAY[1..2,1..100] enthält die Wertepaare der Linearisierungstabelle. Dabei ist ValveCharacteristicTable[nnn,1] der normierte Geschwindigkeitswert und ValveCharacteristicTable[nnn,2] der normierte Ausgabewert. Innerhalb der Tabelle weisen die Wertepaare mit steigendem Index steigende Werte für Geschwindigkeitswert und Ausgabewert. Das erste Wertepaar beschreibt somit den schnellsten negativen und das letzte aktive Wertepaar den schnellsten positiven Punkt. Bei der automatischen Ermittlung wird die Steuerspannung auf **EndOfTravel\_NegativLimit** und **EndOfTravel\_PositivLimit** und die Geschwindigkeit auf **EndOfVelocity\_NegativLimit** und **EndOfVelocity\_PositivLimit** begrenzt. Die weiteren Punkte der Tabelle werden aus den letzten beiden Messpunkten durch Extrapolation ermittelt.

**ValveCharacteristicType:** Hier kann die Identifikation an spezielle Ventil-Varianten oder besondere Bedingungen der Achse angepasst werden. Siehe hierzu auch [E\\_TcMcValveType](#) [► 111].

**ValveCharacteristicTblCount:** Dieser Parameter legt die Anzahl der zu ermittelnden Wertepaare in **ValveCharacteristicTable** fest. Der Wert muss ungerade sein und zwischen 3 und 99 (einschließlich) liegen.

**ValveCharacteristicLowEnd:** [mm] Die untere Endposition des für die Kennlinienermittlung zugelassenen Bereichs.

**ValveCharacteristicHighEnd:** [mm] Die obere Endposition des für die Kennlinienermittlung zugelassenen Bereichs.

**ValveCharacteristicRamp:** [s] Dieser Parameter legt die Rampe beim Aufbau der Messspannung für die Kennlinienermittlung fest. In der angegebenen Zeit wird die Spannung auf 10 Volt erhöht. Da die tatsächlichen Spannungen in der Regel geringer sind wird für den Aufbau eine entsprechend kleinere Zeit benötigt. Die Hinweise am Ende dieses Dokuments sind zu beachten.

**ValveCharacteristicSettling:** [s] Nachdem der Stellwert auf den Testpegel der Messung gerammt wurde kann der Start der Messung durch diesen Parameter verzögert werden. Die Hinweise am Ende dieses Dokuments sind zu beachten.

**ValveCharacteristicRecovery:** [s] Dieser Parameter legt eine Verweilzeit fest, die vor der Messfahrt eingehalten wird. Dadurch erhält die Versorgung die Möglichkeit, einen von der vorherigen Messfahrt verursachten Druckabfall abzubauen.



Die Achse wird während dieser Zeit nicht geregelt.

---

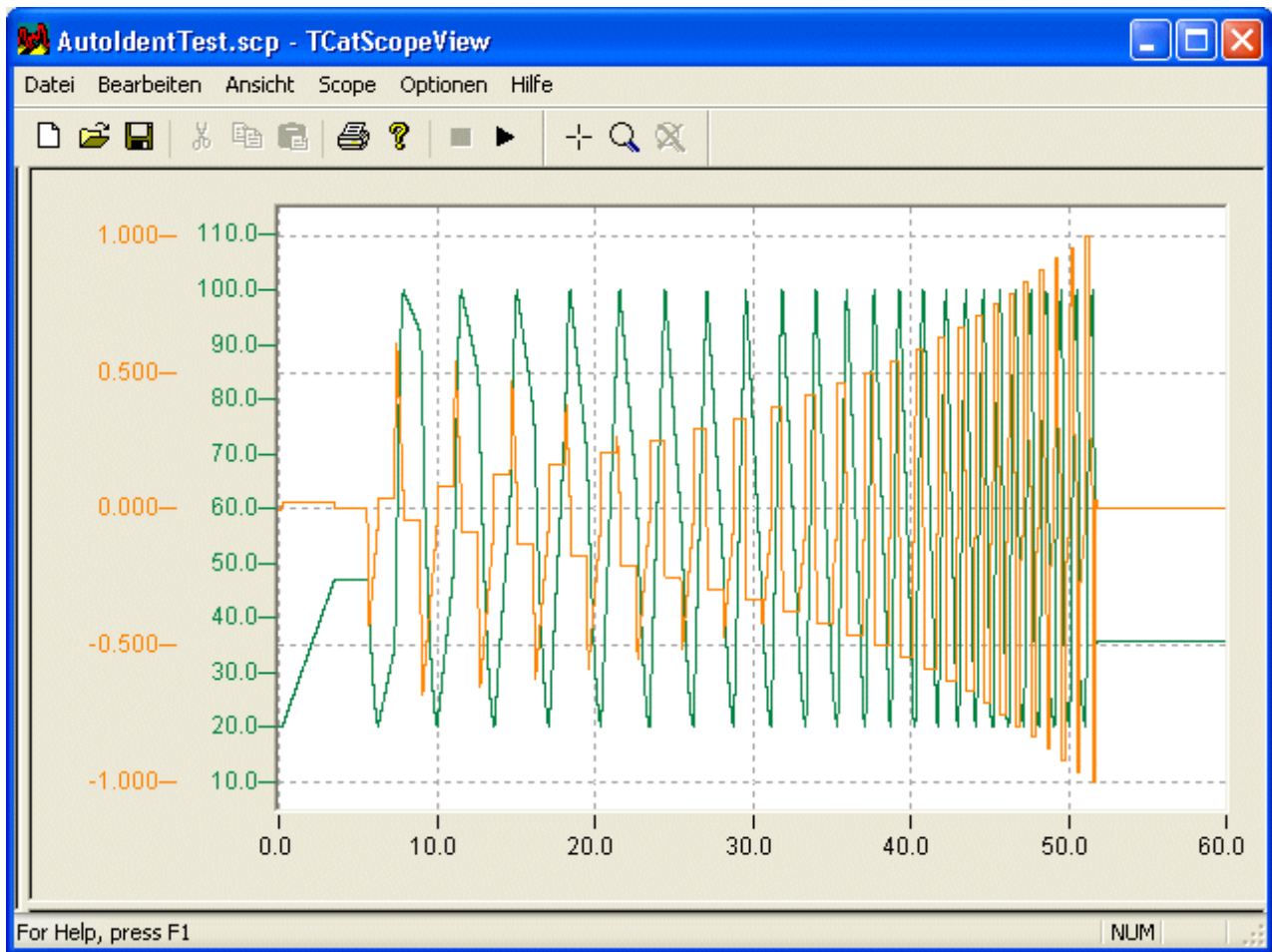
Ab TC2 V3.0.44 / TC3 V3.3.1.22: Der Ablauf der Verweilzeit wird am Ausgang **InRecovery** signalisiert.

**ValveCharacteristicMinCycle:** [mm] Die Messfahrt ist nur gültig wenn der Aufbau der Messspannung abgeschlossen ist, bevor die Achse sich der Mitte der durch **ValveCharacteristicHighEnd** und **ValveCharacteristicLowEnd** festgelegten Messstrecke auf weniger als der Hälfte dieses Wertes genähert hat. Andernfalls ist die effektive Messstrecke (ohne Rampen) kleiner als diese Strecke und diese Messung und alle weiteren in dieser Richtung werden durch einen unter Verwendung der Bezugsgeschwindigkeit der Achse errechneten Wert ersetzt.

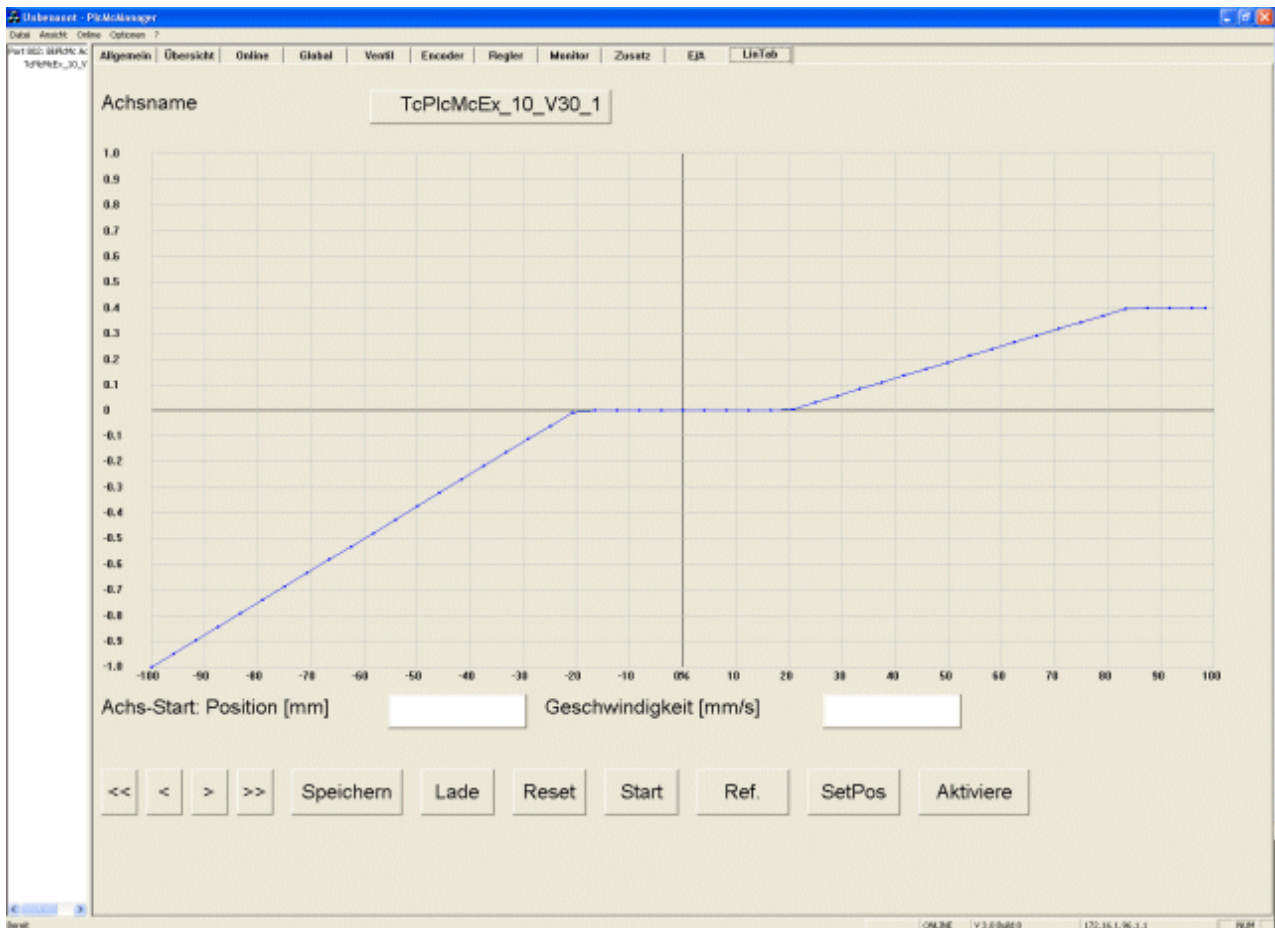


**Valve\_LinLimitP, Valve\_LinLimitM:** [mm/s] Die niedrigste Geschwindigkeit für die Benutzung der Linearisierungstabelle. Für niedrigere Geschwindigkeiten wird die Kennlinie durch eine Gerade ersetzt, die den Nullpunkt mit dem Punkt für die hier angegebene Geschwindigkeit verbindet.

**Beispiel:** Darstellung einer Kennlinienermittlung im TwinCAT ScopeView:



**Beispiel:** Anzeige einer Linearisierung im PlcMcManager:



Die so ermittelte Kennlinie kann mit einem `MC_AxRtFinishLinear_BkPlcMc` Baustein für die Linearisierung zur Laufzeit genutzt werden.



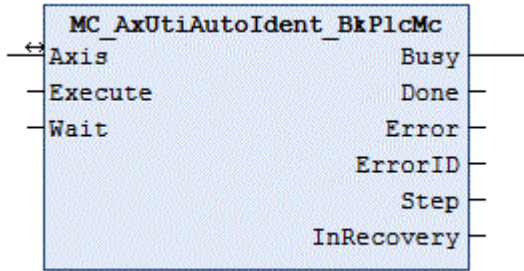
Die Kennlinie wird in der Parameter-Datei der Achse gespeichert und beim Start des Systems automatisch gelesen.

Unabhängig davon kann die Linearisierungstabelle mit einem `MC_LinTableImportFromAsciiFile_BkPlcMc` [▶ 261] oder `MC_LinTableImportFromBinFile_BkPlcMc` [▶ 262] Baustein aus einer Text- bzw. Binär-Datei [▶ 317] importiert oder mit einem `MC_LinTableExportToAsciiFile_BkPlcMc` [▶ 259] oder `MC_LinTableExportToBinFile_BkPlcMc` [▶ 260] Baustein exportiert werden.



Sollte während der Vermessung bei einer Test-Ausgabe eine geringere Geschwindigkeit als beim vorherigen Messpunkt in gleicher Richtung erkannt werden erfolgt unabhängig vom eingestellten Logger-Limit eine Warnung. Der Messpunkt wird automatisch korrigiert um fallende Kennlinienbereiche zu vermeiden. Diese Korrektur hat keinen Einfluss auf die Gültigkeit der Kennlinie. Allerdings sollte überprüft werden, ob die Werte in `ValveCharacteristicRamp` und die `ValveCharacteristicSettling` für diese Achse geeignet sind.

### 4.4.9.7.2 MC\_LinTableExportToAsciiFile\_BkPlcMc



Der Funktionsbaustein exportiert eine Linearisierungstabelle in eine Datei im ASCII Format.

#### Eingänge

```
VAR_INPUT
    Execute:    BOOL:=FALSE;
    FileName:   STRING(255):='';
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke löst den Export aus.
FileName	STRING	

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis Ref BkPlcMc [► 84]</a> zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    Busy:      BOOL;
    Done:      BOOL;
    Error:     BOOL;
    ErrorID:   UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung der Identifikation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

#### Verhalten des Bausteins

Der Baustein erzeugt eine Datei im von [MC\\_LinTableImportFromAsciiFile\\_BkPlcMc \[► 261\]](#) erwarteten Format.

### 4.4.9.7.3 MC\_LinTableExportToBinFile\_BkPlcMc



Der Funktionsbaustein exportiert eine Linearisierungstabelle in eine Datei im Binär-Format.

#### Eingänge

```
VAR_INPUT
    Execute:      BOOL:=FALSE;
    FileName:     STRING(255):='';
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke löst den Export aus.
FileName	STRING	

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:         Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [► 84]</a> zu übergeben.

#### Ausgänge

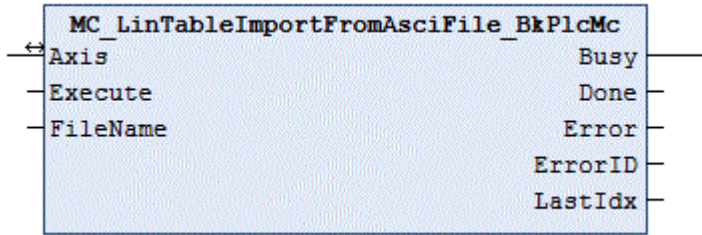
```
VAR_OUTPUT
    Busy:         BOOL;
    Done:         BOOL;
    Error:        BOOL;
    ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung der Identifikation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

#### Verhalten des Bausteins

Der Baustein erzeugt eine Datei im von [MC\\_LinTableImportFromBinFile\\_BkPlcMc \[► 262\]](#) erwarteten Format.

### 4.4.9.7.4 MC\_LinTableImportFromAsciiFile\_BkPlcMc



Der Funktionsbaustein importiert eine Linearisierungstabelle aus einer Datei im ASCII Format.

#### Eingänge

```

VAR_INPUT
  Execute:      BOOL:=FALSE;
  FileName:    STRING(255) := '';
END_VAR
  
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke löst den Import aus.
FileName	STRING	

#### Ein-/Ausgänge

```

VAR_INOUT
  Axis:        Axis_Ref_BkPlcMc;
END_VAR
  
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis Ref BkPlcMc [► 84]</a> zu übergeben.

#### Ausgänge

```

VAR_OUTPUT
  Busy:        BOOL;
  Done:        BOOL;
  Error:        BOOL;
  ErrorID:     UDINT;
  LastIdx:     INT:=0;
END_VAR
  
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung der Identifikation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
LastIdx	INT	

#### Verhalten des Bausteins

Der Baustein liest den Inhalt der Datei und interpretiert ihn zeilenweise nach den folgenden Regeln:

- Führende Leerzeichen (Blanks, Tabs) werden ignoriert.
- Die Ziffern 0..9 werden akzeptiert und als Stellen vor dem Komma für den ersten Wert verwendet.
- Ein Punkt oder Komma wird als Trennung zwischen Stellen vor und nach dem Komma interpretiert.
- Die Ziffern 0..9 werden akzeptiert und als Stellen nach dem Komma für den ersten Wert verwendet.
- Leerzeichen (Blanks, Tabs) werden als Trennung zwischen dem ersten und dem zweiten Wert interpretiert.

- Die Ziffern 0..9 werden akzeptiert und als Stellen vor dem Komma für den zweiten Wert verwendet.
- Ein Punkt oder Komma wird als Trennung zwischen Stellen vor und nach dem Komma interpretiert.
- Die Ziffern 0..9 werden akzeptiert und als Stellen nach dem Komma für den zweiten Wert verwendet.
- Werden unerwartete Zeichen erkannt oder fehlen erwartete Elemente wird der Import mit einem Fehler abgebrochen.
- Jedes Zahlenpaar wird als Punkt in die Linearisierungstabelle der Achse eingetragen. Dabei wird LastIdx hochgezählt. Nach erfolgreichem Import steht hier die Anzahl der gelesenen Tabellenpunkte bereit.

**Manipulation der Datei**

Das ASCII Format macht es leicht, eine solche Datei mit einem einfachen Editor zu manipulieren. Dies ist möglich, aber nicht empfehlenswert. Eine Abweichung vom erwarteten Aufbau der Datei macht es unmöglich, sie zu importieren. Aber selbst bei Beachtung der Formatierung kann unbeabsichtigt eine Linearisierungstabelle erzeugt werden, die eine korrekte Funktion der Anlage unmöglich macht. Darüber hinaus kann es zu Gefahren für Personen und Schäden an Produkten oder der Anlage kommen.

**4.4.9.7.5 MC\_LinTableImportFromBinFile\_BkPlcMc**



Der Funktionsbaustein importiert eine Linearisierungstabelle aus einer Datei im Binär-Format.

**Eingänge**

```
VAR_INPUT
  Execute:      BOOL:=FALSE;
  FileName:     STRING(255):='';
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke löst den Import aus.
FileName	STRING	

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:         Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Busy:         BOOL;
  Done:         BOOL;
  Error:        BOOL;
  ErrorID:      UDINT;
  LastIdx:      INT:=0;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird die erfolgreiche Durchführung der Identifikation signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.
LastIdx	INT	

**Verhalten des Bausteins**

Der Baustein liest den Inhalt der Datei und interpretiert in zeilenweise nach folgenden Regeln:

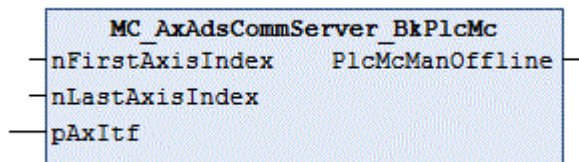
- Je zwei Werte werden als Zahlenpaar betrachtet.
- Jedes Zahlenpaar wird als Punkt in die Linearisierungstabelle der Achse eingetragen. Dabei wird LastIdx hochgezählt. Nach erfolgreichem Import steht hier die Anzahl der gelesenen Tabellenpunkte bereit.

**Manipulation der Datei**

Das Binär-Format macht es praktisch unmöglich, eine solche Datei mit einem einfachen Editor zu manipulieren. Mit einem entsprechenden Programm ist dies möglich, aber absolut nicht empfehlenswert. Der erwartete Aufbau der Datei ist kaum ersichtlich und es ist praktisch unmöglich, die Formatierung einzuhalten. Aber selbst bei Beachtung der Formatierung wird es kaum gelingen, eine brauchbare Linearisierungstabelle zu erzeugen. Die korrekte Funktion der Anlage wird nicht möglich sein. Darüber hinaus kann es zu Gefahren für Personen und Schäden an Produkten oder der Anlage kommen.

## 4.5 Parameter

### 4.5.1 MC\_AxAdsCommServer\_BkPlcMc (ab V3.0)



Der Funktionsbaustein stattet die Applikation mit den Fähigkeiten eines ADS-Servers aus. Er ruft bei Bedarf Bausteine vom Typ [MC\\_AxAdsReadDecoder\\_BkPlcMc](#) [▶ 266] und [MC\\_AxAdsWriteDecoder\\_BkPlcMc](#) [▶ 268] auf. Eine Auflistung der nutzbaren [ADS-Codes](#) [▶ 327] finden Sie in der Knowledge Base.

**Eingänge**

```

VAR_INPUT
  nFirstAxisIndex:   INT;
  nLastAxisIndex:   INT;
END_VAR
    
```

Name	Typ	Beschreibung
nFirstAxisIndex	INT	Hier ist die Dimensionierung des <a href="#">Axis_Ref_BkPlcMc</a> [▶ 84] Arrays anzugeben. <b>Eine nicht zutreffende Angabe an dieser Stelle schließt einen Teil der Achsen von der Kommunikation aus oder führt zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>
nLastAxisIndex	INT	

**Ein-/Ausgänge**

```

VAR_INOUT
  pAxItf:           POINTER TO Axis_Ref_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
pAxItf	POINTER	Hier ist die Adresse einer Variablen oder eines Arrays von Variablen vom Typ <u>Axis_Ref BkPlcMc</u> [▶ 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
    PlcMcManOffline:    BOOL;
END_VAR
```

Name	Typ	Beschreibung
PlcMcManOffline	BOOL	Dieser Parameter entspricht dem gleichnamigen Parameter des Bausteins <u>MC_AxAdsCommServer_BkPlcMc</u> [▶ 263].

**Verhalten des Bausteins**

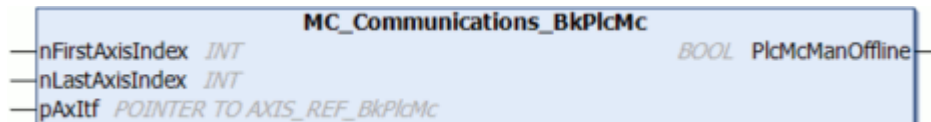
Durch zyklischen Aufruf dieses Bausteins in der PLC-Applikation erhält diese den Charakter eines ADS-Servers und beantwortet ADS-Read- und ADS-Write-Zugriffe wie jeder andere ADS Server. Dies schließt die Dekodierung einer IdxGroup/IdxOffset-Adressierung ein. Dazu werden bei Bedarf Bausteine vom Typ MC\_AxAdsReadDecoder\_BkPlcMc [▶ 266] und MC\_AxAdsWriteDecoder\_BkPlcMc [▶ 268] aufgerufen.



Ist die PLC-Applikation bereits ein ADS-Server darf dieser Baustein nicht verwendet werden.

In diesem Fall sind die Bausteine vom Typ MC\_AxAdsReadDecoder\_BkPlcMc [▶ 266] und MC\_AxAdsWriteDecoder\_BkPlcMc [▶ 268] aus dem vorhandenen ADS Server-Baustein der Applikation aufzurufen.

### 4.5.2 MC\_Communications\_BkPlcMc



Der Funktionsbaustein ruft intern die Funktionsbausteine MC\_AxAdsCommServer\_BkPlcMc [▶ 263] und MC\_AxRtLoggerSpool\_BkPlcMc [▶ 244] auf. Außerdem stellt er einen Meldungspuffer zur Verfügung.

**Eingänge**

```
VAR_INPUT
    nFirstAxisIndex:    INT;
    nLastAxisIndex:    INT;
    pAxItf:             POINTER TO Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
nFirstAxisIndex	INT	Diese Parameter entsprechen den gleichnamigen Parametern des Bausteins <u>MC_AxAdsCommServer_BkPlcMc</u> .
nLastAxisIndex	INT	
pAxItf	POINTER	Hier ist die Adresse des <u>Axis_Ref BkPlcMc</u> [▶ 84] Arrays zu übergeben. <b>Eine nicht zutreffende Angabe an dieser Stelle führt unausweichlich zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>

**Ausgänge**

```
VAR_OUTPUT
    PlcMcManOffline:    BOOL;
END_VAR
```



Name	Typ	Beschreibung
PlcMcManOffline	BOOL	Dieser Parameter entspricht dem gleichnamigen Parameter des Bausteins MC_AxAdsCommServer_BkPlcMc.

-----

**nFirstAxisIdx, nLastAxisIdx:** Hier ist die Dimensionierung des [Axis\\_Ref\\_BkPlcMc \[▶ 84\]](#) Arrays anzugeben.

**HINWEIS**

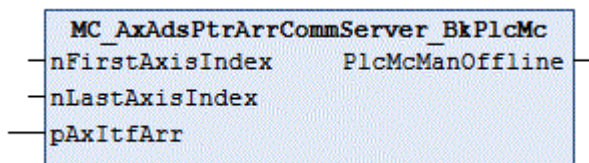
**Absturz der PLC-Applikation**

Eine nicht zutreffende Angabe an dieser Stelle schließt einen Teil der Achsen von der Kommunikation aus oder führt zum **Absturz der PLC-Applikation** durch Auslösung von schweren Laufzeitfehlern (**Page Fault Exception**)

**Verhalten des Bausteins**

Durch zyklischen Aufruf dieses Bausteins in der PLC-Applikation werden die übergebenen Achsen an einen internen Meldungspuffer angebunden. Der beim Aufruf von [MC\\_AxUtiStandardInit\\_BkPlcMc \[▶ 238\]](#) referenzierte Meldungspuffer wird ignoriert, wenn der Baustein [MC\\_Communications\\_BkPlcMc](#) aufgerufen wird. Die Meldungen aus dem internen Meldungspuffer werden zyklisch in die Ereignisanzeige von Windows übertragen, indem intern eine Instanz des Bausteins [MC\\_AxRtLoggerSpool\\_BkPlcMc \[▶ 244\]](#) aufgerufen wird. Außerdem erhält die PLC-Applikation den Charakter eines ADS-Servers, da intern eine Instanz des Bausteins [MC\\_AxAdsCommServer\\_BkPlcMc \[▶ 263\]](#) aufgerufen wird.

### 4.5.3 MC\_AxAdsPtrArrCommServer\_BkPlcMc



Der Funktionsbaustein stattet die Applikation mit den Fähigkeiten eines ADS-Servers aus. Er ruft bei Bedarf Bausteine vom Typ [MC\\_AxAdsReadDecoder\\_BkPlcMc \[▶ 266\]](#) und [MC\\_AxAdsWriteDecoder\\_BkPlcMc \[▶ 268\]](#) auf. Eine Auflistung der nutzbaren [ADS-Codes \[▶ 327\]](#) finden Sie in der Knowledge Base.



Für die meisten Applikationen ist die Nutzung eines [MC\\_AxAdsCommServer\\_BkPlcMc](#) ausreichend und vorzuziehen.

([MC\\_AxAdsCommServer\\_BkPlcMc \[▶ 263\]](#))

**Eingänge**

```
VAR_INPUT
  nFirstAxisIndex: INT;
  nLastAxisIndex: INT;
END_VAR
```

Name	Typ	Beschreibung
nFirstAxisIndex	INT	Hier ist die Dimensionierung des <a href="#">Axis_Ref_BkPlcMc [▶ 84]</a> Arrays anzugeben.
nLastAxisIndex	INT	<b>Eine nicht zutreffende Angabe an dieser Stelle schließt einen Teil der Achsen von der Kommunikation aus oder führt zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>

 **Ein-/Ausgänge**

```
VAR_INOUT
  pAxItfArr:      POINTER TO DWORD;
END_VAR
```

Name	Typ	Beschreibung
pAxItfArr	POINTER	Hier ist die Adresse einer Variablen vom Typ ARRAY [ncnstFirstAxId..ncnstLastAxId] OF POINTER TO <u>Axis_Ref_BkPlcMc</u> [▶ 84] zu übergeben.  <b>Eine nicht zutreffende Angabe an dieser Stelle führt unausweichlich zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>

 **Ausgänge**

```
VAR_OUTPUT
  PlcMcManOffline:  BOOL;
END_VAR
```

Name	Typ	Beschreibung
PlcMcManOffline	BOOL	Dieser Parameter entspricht dem gleichnamigen Parameter des Bausteins <u>MC_AxAdsCommServer_BkPlcMc</u> .

**Verhalten des Bausteins**

Durch zyklischen Aufruf dieses Bausteins in der PLC-Applikation erhält diese den Charakter eines ADS-Servers und beantwortet ADS-Read- und ADS-Write-Zugriffe wie jeder andere ADS Server. Dies schließt die Dekodierung einer IdxGroup/IdxOffset-Adressierung ein. Dazu werden bei Bedarf Bausteine vom Typ MC\_AxAdsReadDecoder\_BkPlcMc [▶ 266] und MC\_AxAdsWriteDecoder\_BkPlcMc [▶ 268] aufgerufen.

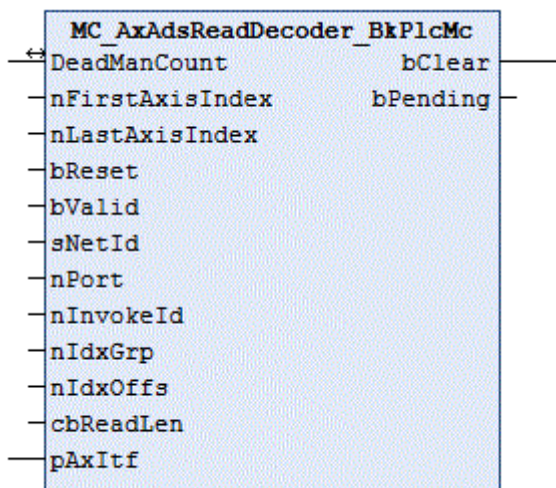


Ist die PLC-Applikation bereits ein ADS-Server darf dieser Baustein nicht verwendet werden.


In diesem Fall sind die Bausteine vom Typ MC\_AxAdsReadDecoder\_BkPlcMc [▶ 266] und MC\_AxAdsWriteDecoder\_BkPlcMc [▶ 268] aus dem vorhandenen ADS Server-Baustein der Applikation aufzurufen.

Es steht ein Programm-Beispiel [▶ 303] #16 zur Verfügung.

**4.5.4 MC\_AxAdsReadDecoder\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein dekodiert ADS-Read-Zugriffe. Eine Auflistung der nutzbaren [ADS-Codes](#) [► 327] finden Sie in der Knowledge Base.

 **Eingänge**

```
VAR_INPUT
  nFirstAxisIndex: INT;
  nLastAxisIndex: INT;
  bReset: BOOL;
  bValid: BOOL;
  sNetId: STRING(80);
  nPort: UINT;
  nInvokeId: UDINT;
  nIdxGroup: UDINT;
  nIdxOffs: UDINT;
  cbReadLen: UDINT;
  pAxItf: POINTER TO Axis_Ref_BkPlcMc:=0;
END_VAR
```

Name	Typ	Beschreibung
nFirstAxisIndex	INT	Hier ist die Dimensionierung des <a href="#">Axis_Ref_BkPlcMc</a> [► 84] Arrays anzugeben. <b>Eine nicht zutreffende Angabe an dieser Stelle schließt einen Teil der Achsen von der Kommunikation aus oder führt zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>
nLastAxisIndex	INT	
bReset	BOOL	Diese Signale dienen zur Koordinierung des Dekoders mit dem ADS-Server.
bValid	BOOL	
sNetId	STRING	Diese Werte werden für die Erzeugung des ADS Response benötigt. Sie werden vom ADS-Indication-Baustein eines ADS-Servers geliefert.
nPort	UINT	
nInvokeld	UDINT	
nIdxGroup	UDINT	Diese Werte werden für die Dekodierung des Zugriffs benötigt. Sie werden vom ADS-Indication-Baustein eines ADS-Servers geliefert.
nIdxOffs	UDINT	
cbReadLen	UDINT	
pAxItf	POINTER	Hier ist die Adresse einer Variablen oder eines Arrays von Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc</a> [► 84] zu übergeben.

 **Ein-/Ausgänge**

```
VAR_INOUT
  DeadManCount: UDINT;
END_VAR
```

Name	Typ	Beschreibung
DeadManCount	UDINT	

 **Ausgänge**

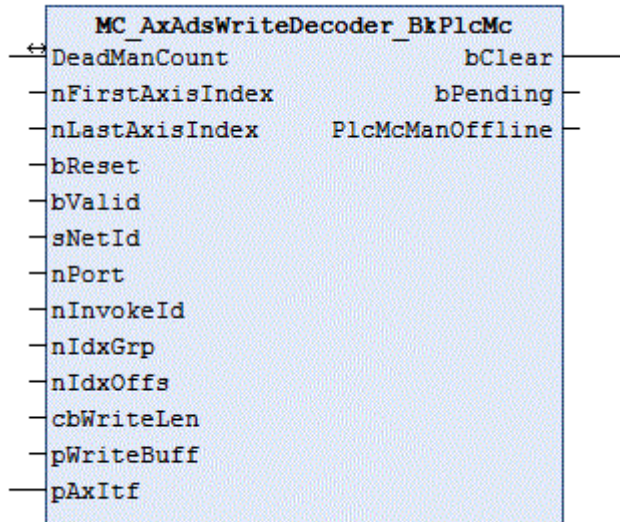
```
VAR_OUTPUT
  bClear: BOOL;
  bPending: BOOL;
END_VAR
```

Name	Typ	Beschreibung
bClear	BOOL	Hier wird signalisiert, dass ein mit bValid signalisierter ADS-Zugriff quittiert werden soll.
bPending	BOOL	Hier wird signalisiert, dass ein mit bValid signalisierter ADS-Zugriff bearbeitet wird.

**Verhalten des Bausteins**

Signalisiert der Baustein bei anstehendem bValid-Signal weder bClear noch bPending hat er die Kombination aus nIdxGroup und nIdxOffs nicht dekodiert und es wurde kein Response erzeugt. In diesem Fall muss der ADS-Server (soweit vorhanden) andere Decoder aufrufen oder einen Response mit entsprechendem Errorcode erzeugen.

**4.5.5 MC\_AxAdsWriteDecoder\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein dekodiert ADS-Write-Zugriffe. Eine Auflistung der nutzbaren [ADS-Codes](#) [▶ 327] finden Sie in der Knowledge Base.

**Eingänge**

```

VAR_INPUT
  nFirstAxisIndex: INT;
  nLastAxisIndex: INT;
  bReset: BOOL;
  bValid: BOOL;
  sNetId: STRING(80);
  nPort: UINT;
  nInvokeId: UDINT;
  nIdxGroup: UDINT;
  nIdxOffs: UDINT;
  cbWriteLen: UDINT;
  pWriteBuff: DWORD;
  pAxItf: POINTER TO Axis_Ref_BkPlcMc:=0;
END_VAR
    
```

Name	Typ	Beschreibung
nFirstAxisIndex	INT	Hier ist die Dimensionierung des <a href="#">Axis_Ref_BkPlcMc</a> [▶ 84] Arrays anzugeben.  <b>Eine nicht zutreffende Angabe an dieser Stelle schließt einen Teil der Achsen von der Kommunikation aus oder führt zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception).</b>
nLastAxisIndex	INT	
bReset	BOOL	Diese Signale dienen zur Koordinierung des Decoders mit dem ADS-Server.
bValid	BOOL	
sNetId	STRING	Diese Werte werden für die Erzeugung des ADS Response benötigt. Sie werden vom ADS-Indication-Baustein eines ADS-Servers geliefert.
nPort	UINT	
nInvokeld	UDINT	
nIdxGroup	UDINT	Diese Werte werden für die Dekodierung des Zugriffs benötigt. Sie werden vom ADS-Indication-Baustein eines ADS-Servers geliefert.
nIdxOffs	UDINT	

Name	Typ	Beschreibung
cbWriteLen	UDINT	
pWriteBuff	DWORD	
pAxIxf	POINTER	Hier ist die Adresse einer Variablen oder eines Arrays von Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

 **Ein-/Ausgänge**

```
VAR_INOUT
  DeadManCount:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
DeadManCount	UDINT	

 **Ausgänge**

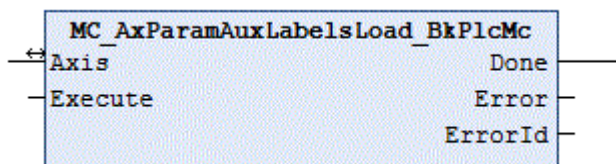
```
VAR_OUTPUT
  bClear:             BOOL;
  bPending:           BOOL;
  PlcMcManOffline:   BOOL;
END_VAR
```

Name	Typ	Beschreibung
bClear	BOOL	Hier wird signalisiert, dass ein mit bValid signalisierter ADS-Zugriff quittiert werden soll.
bPending	BOOL	Hier wird signalisiert, dass ein mit bValid signalisierter ADS-Zugriff bearbeitet wird.
PlcMcManOffline	BOOL	

**Verhalten des Bausteins**

Signalisiert der Baustein bei anstehendem bValid-Signal weder bClear noch bPending hat er die Kombination aus nIdxGroup und nIdxOffs nicht dekodiert und es wurde kein Response erzeugt. In diesem Fall muss der ADS-Server (soweit vorhanden) andere Decoder aufrufen oder einen Response mit entsprechendem Errorcode erzeugen.

**4.5.6 MC\_AxParamAuxLabelsLoad\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein lädt die Beschriftungstexte für die kundenspezifischen Achsparameter aus einer Datei. Diese Texte können mit einem einfachen Texteditor wie Microsoft Notepad erzeugt werden.



Die Datei muss strikt entsprechend der unten angegebenen Regeln aufgebaut sein. Andernfalls kann es zu erheblichen Problemen bis hin zum Systemabsturz kommen.

Dieser Baustein wird in der Regel nicht direkt durch die Applikation aufgerufen. Soweit möglich sollte ein Baustein des Typs `MC_AxUtiStandardInit_BkPlcMc` [► 238] verwendet werden, der einen Baustein des Typs `MC_AxParamAuxLabelsLoad_BkPlcMc` verwendet.

**Eingänge**

```
VAR_INPUT
    Execute:          BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Ladevorgang.

**Ein-/Ausgänge**

```
VAR_INOUT
    Axis:             Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
    Done:             BOOL;
    Error:             BOOL;
    ErrorID:           UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird das erfolgreiche Laden der Parameter signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

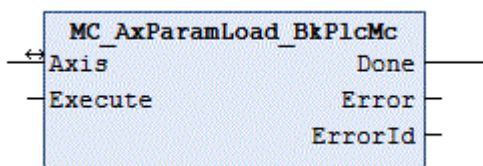
- Wenn einer der Pointer nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** oder **dwTcHydErrCdPtrMcPlc** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird der Ladevorgang initiiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Ladevorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

**i** Die Anzahl der Zeilen in der Datei muss mit der in den globalen Konstanten der Bibliothek als `iTcHydfCustDataMaxIdx` (derzeit: 20) festgelegten Anzahl übereinstimmen. Jede der Zeilen darf maximal 20 Zeichen (inklusive Leerzeichen, ohne Zeilenwechsel) enthalten.

**4.5.7 MC\_AiParamLoad\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein lädt die Parameter einer Achse aus einer Datei. Für die kompatible Erzeugung der Parameterdatei ist ein Baustein vom Typ `MC_AiParamSave_BkPlcMc` [► 271] zu verwenden.

Dieser Baustein wird in der Regel nicht direkt durch die Applikation aufgerufen. Soweit möglich sollte ein Baustein des Typs MC\_AxUtilStandardInit\_BkPlcMc [► 238] verwendet werden, der einen Baustein des Typs **MC\_AiParamLoad\_BkPlcMc** verwendet.

**Eingänge**

```
VAR_INPUT
    Execute:          BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Ladevorgang.

**Ein-/Ausgänge**

```
VAR_INOUT
    Axis:             Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
    Done:             BOOL;
    Error:             BOOL;
    ErrorID:           UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird das erfolgreiche Laden der Parameter signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

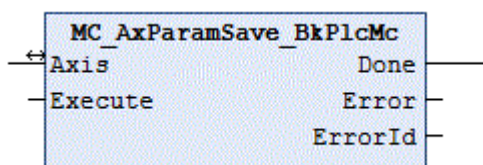
Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn die Datei nicht zum Lesen geöffnet werden kann wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** oder **dwTcHydErrCdPtrMcPlc** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnte wird der Ladevorgang initiiert. Dabei wird die Version der Datei ermittelt und nicht durch die Datei festgelegte Parameter durch neutrale Default-Werte ersetzt. Sollte die Datei nicht mehr oder noch nicht verwendete Parameter enthalten werden diese ignoriert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Ladevorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

**4.5.8 MC\_AiParamSave\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein schreibt die Parameter einer Achse in eine Datei. Für das Einlesen der Datei ist ein Baustein vom Typ `MC_AxParamLoad_BkPlcMc` [► 270] zu verwenden.

 **Eingänge**

```
VAR_INPUT
  Execute:          BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:            Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Done:           BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird das erfolgreiche Schreiben der Parameter signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

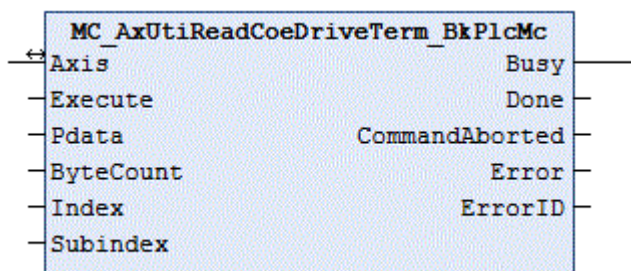
Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn die Datei nicht zum Schreiben geöffnet werden kann wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** oder **dwTcHydErrCdPtrMcPlc** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnte wird der Schreibvorgang initiiert. Dabei wird die Version der abgespeicherten Parameter festgehalten.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Schreibvorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

**4.5.9 MC\_AxUtiReadCoeDriveTerm\_BkPlcMc (ab V3.0)**





Der Funktionsbaustein liest den Inhalt eines Registers aus der EL-Klemme, die als Antriebsschnittstelle für die Achse dient.

 **Eingänge**

```
VAR_INPUT
  Execute:      BOOL;
  Pdata:        POINTER TO BYTE:=0;
  ByteCount:    BYTE:=0;
  Index:        WORD:=0;
  Subindex:     BYTE:=0;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Lesevorgang.
Pdata	POINTER	Hier ist die Adresse der Variablen anzugeben, in die der gelesene Wert ausgegeben werden soll.
ByteCount	BYTE	Hier ist die Größe der Variablen in Bytes anzugeben.
Index	WORD	Hier ist die Adressierung des Parameters in der Klemme anzugeben.
Subindex	BYTE	

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:         Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis Ref BkPlcMc [► 84]</a> zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:         BOOL;
  Done:         BOOL;
  CommandAborted: BOOL;
  Error:        BOOL;
  ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Laden des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

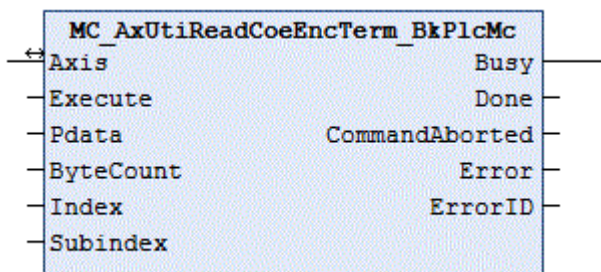
Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Index** oder **Subindex** ausserhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTbllllegalIndex** reagiert.
- Wenn **ByteCount** oder **Pdata** ausserhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTbllllegalIndex** reagiert.
- Wenn in den Achsparametern als nDrive\_Type eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.

- Wenn es bei der ADS-Kommunikation mit der Klemme zu Problemen kommt wird der entsprechende ADS Error Code als **ErrorID** zurückgegeben und dies mit **Error** kenntlich gemacht. Dabei können unter anderen folgende Codes [▶ 321] auftreten:
  - 16#0006 = 6 = Die Portnummer der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0007 = 7 = Die AmsNetID der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0702 = 1794 = dwTcHydAdsErrInvalidIdxGroup = Die Klemme unterstützt nicht das CoE Protokoll.
  - 16#0703 = 1795 = dwTcHydAdsErrInvalidIdxOffset = Die Adresse in Index und Subindex ist in der Klemme nicht unterstützt.
  - 16#0745 = 1861 = dwTcHydAdsErrTimeout = Zeitüberschreitung.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Ladevorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Done**, **CommandAborted**, **Error**, **ErrorID**) werden für einen Zyklus gegeben.

### 4.5.10 MC\_AxUtiReadCoeEncTerm\_BkPlcMc (ab V3.0)



Der Funktionsbaustein liest den Inhalt eines Registers aus der EL-Klemme, die als Encoder-Schnittstelle für die Achse dient.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  Pdata:        POINTER TO BYTE:=0;
  ByteCount:    BYTE:=0;
  Index:        WORD:=0;
  Subindex:     BYTE:=0;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Lesevorgang.
Pdata	POINTER	Hier ist die Adresse der Variablen anzugeben, in die der gelesene Wert ausgegeben werden soll.
ByteCount	BYTE	Hier ist die Größe der Variablen in Bytes anzugeben.
Index	WORD	Hier ist die Adressierung des Parameters in der Klemme anzugeben.
Subindex	BYTE	

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:         Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  CommandAborted: BOOL;
  Error:         BOOL;
  ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Laden des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

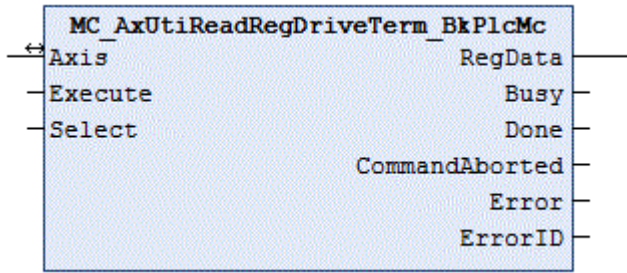
**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Index** oder **Subindex** außerhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTbllllegalIndex** reagiert.
- Wenn **ByteCount** oder **Pdata** außerhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTbllllegalIndex** reagiert.
- Wenn in den Achsparametern als **nEncoder\_Type** eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.
- Wenn es bei der ADS-Kommunikation mit der Klemme zu Problemen kommt wird der entsprechende ADS Error Code als **ErrorID** zurückgegeben und dies mit **Error** kenntlich gemacht. Dabei können unter anderen folgende Codes [► 321] auftreten:
  - 16#0006 = 6 = Die Portnummer der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0007 = 7 = Die AmsNetID der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0702 = 1794 = dwTcHydAdsErrInvalidIdxGroup = Die Klemme unterstützt nicht das CoE Protokoll.
  - 16#0703 = 1795 = dwTcHydAdsErrInvalidIdxOffset = Die Adresse in Index und Subindex ist in der Klemme nicht unterstützt.
  - 16#0745 = 1861 = dwTcHydAdsErrTimeout = Zeitüberschreitung.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Ladevorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Done**, **CommandAborted**, **Error**, **ErrorID**) werden für einen Zyklus gegeben.

### 4.5.11 MC\_AxUtiReadRegDriveTerm\_BkPlcMc (ab V3.0)



Der Funktionsbaustein liest den Inhalt eines Registers aus der KL-Klemme, die als Antriebsschnittstelle für die Achse dient.

#### Eingänge

```
VAR_INPUT
    Execute:      BOOL;
    Select:       INT;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Lesevorgang.
Select	INT	Hier ist die Registernummer zu übergeben.

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
    RegData:      WORD;
    Busy:         BOOL;
    Done:         BOOL;
    CommandAborted: BOOL;
    Error:        BOOL;
    ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
RegData	WORD	Hier wird der gelesene Wert ausgegeben.
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Laden des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

#### Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn einer der Pointer ST\_TcPlcDeviceInput [[▶ 138](#)] und ST\_TcPlcDeviceOutput [[▶ 140](#)] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.

- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Select** außerhalb des zulässigen Bereichs von 0 bis 63 liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdTbllllegalIndex** reagiert.
- Wenn in den Achsparametern als nDrive\_Type eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.

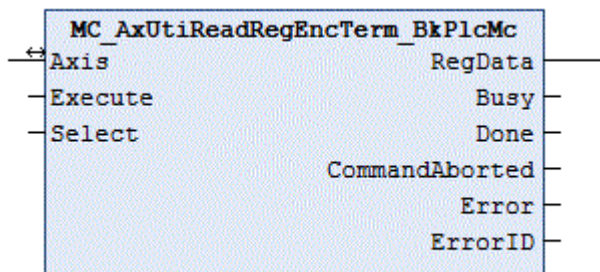
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird der Lesevorgang initiiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Ladevorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**RegData**, **Done**, **CommandAborted**, **Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.



Die Drivetypes iTcMc\_DriveKL2521, iTcMc\_DriveKL4032, iTcMc\_DriveKL2531 und iTcMc\_DriveKL2541 unterstützen die Parameterkommunikation.

### 4.5.12 MC\_AxUtiReadRegEncTerm\_BkPlcMc (ab V3.0)



Der Funktionsbaustein liest den Inhalt eines Registers aus der KL-Klemme, die als Encoderschnittstelle für die Achse dient.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  Select:      INT;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Lesevorgang.
Select	INT	Hier ist die Registernummer zu übergeben.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc [► 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  RegData:      WORD;
  Busy:      BOOL;
  Done:      BOOL;
```

```

CommandAborted: BOOL;
Error:          BOOL;
ErrorID:       UDINT;
END_VAR

```

Name	Typ	Beschreibung
RegData	WORD	Hier wird der gelesene Wert ausgegeben.
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Laden des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn einer der Pointer **ST\_TcPlcDeviceInput** [▶ 138] und **ST\_TcPlcDeviceOutput** [▶ 140] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.
- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Select** ausserhalb des zulässigen Bereichs von 0 bis 63 liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn in den Achsparametern als **nEncoder\_Type** eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird der Lesevorgang initiiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Ladevorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**RegData**, **Done**, **CommandAborted**, **Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

**i** Die Drivetypes **iTcMc\_EncoderKL3002**, **iTcMc\_EncoderKL3042**, **iTcMc\_EncoderKL3062**, **iTcMc\_EncoderKL3162**, **iTcMc\_EncoderKL5101**, **iTcMc\_EncoderKL5111**, **iTcMc\_EncoderKL2521**, **iTcMc\_EncoderKL2531** und **iTcMc\_EncoderKL2541** unterstützen die Parameterkommunikation.

**4.5.13 MC\_AxUtiUpdateRegDriveTerm\_BkPlcMc (ab V3.0.7)**



Der Funktionsbaustein schreibt einen Parametersatz in die Register einer KL-Klemme. Dazu benutzt er **MC\_AxUtiReadRegDriveTerm\_BkPlcMc** [▶ 276] und **MC\_AxUtiWriteRegDriveTerm\_BkPlcMc** [▶ 285] Bausteine.

**Eingänge**

```

VAR_INPUT
Execute:      BOOL;
END_VAR

```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.

 Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
  RegData:      ST_TcPlcRegDataTable;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis Ref BkPlcMc</u> [▶ 84] zu übergeben.
RegData	ST_TcPlcRegDataTable	Hier ist die Adresse des Parametersatzes anzugeben, deren Inhalt in die Klemme geschrieben werden soll.

 Ausgänge

```
VAR_OUTPUT
  Done:          BOOL;
  Busy:          BOOL;
  CommandAborted: BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird das erfolgreiche Schreiben des Parameters signalisiert.
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

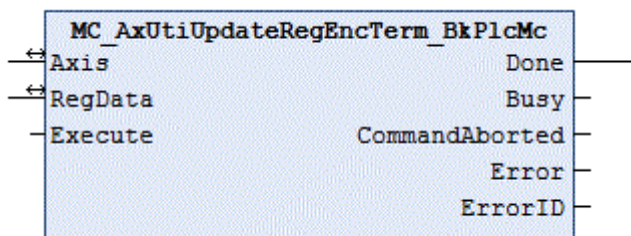
Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn einer der Pointer ST\_TcPlcDeviceInput [▶ 138] und ST\_TcPlcDeviceOutput [▶ 140] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.
- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Select** außerhalb des zulässigen Bereichs von 0 bis 63 liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn in den Achsparametern als nDrive\_Type eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.
- Der Wert in ST\_TcPlcRegDataTable [▶ 144].RegDataItem[...].**Access** legt fest, wie das Element behandelt wird.
  - 0: Element wird ignoriert.
  - 1: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem ungleichen Inhalt wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
  - 2: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht größer ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
  - 3: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht kleiner ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.

- 4: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht größer oder gleich ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
- 5: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht kleiner oder gleich ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
- 10: Das durch **Select** adressierte Register wird mit **RegData** geschrieben.
- Andere Werte werden derzeit ignoriert. Zukünftige Versionen der Bibliothek werden jedoch möglicherweise weitere Funktionen unterstützen. Ein leeres Element sollte daher immer mit 0 gekennzeichnet werden.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Schreibvorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Done**, **CommandAborted**, **Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

### 4.5.14 MC\_AxUtiUpdateRegEncTerm\_BkPlcMc (ab V3.0.7)



Der Funktionsbaustein schreibt einen Parametersatz in die Register einer KL-Klemme. Dazu benutzt er [MC\\_AxUtiReadRegEncTerm\\_BkPlcMc \[ 277 \]](#) und [MC\\_AxUtiWriteRegEncTerm\\_BkPlcMc \[ 286 \]](#) Bausteine.

#### Eingänge

```
VAR_INPUT
    Execute:      BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.

#### Ein-/Ausgänge

```
VAR_INOUT
    Axis:      Axis_Ref_BkPlcMc;
    RegData:   ST_TcPlcRegDataTable;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis Ref BkPlcMc [ 84 ]</a> zu übergeben.
RegData	ST_TcPlcRegDataTable	Hier ist die Adresse des Parametersatzes anzugeben, deren Inhalt in die Klemme geschrieben werden soll.

#### Ausgänge

```
VAR_OUTPUT
    Done:      BOOL;
    Busy:      BOOL;
    CommandAborted: BOOL;
    Error:     BOOL;
    ErrorID:   UDINT;
END_VAR
```



Name	Typ	Beschreibung
Done	BOOL	Hier wird das erfolgreiche Schreiben des Parameters signalisiert.
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

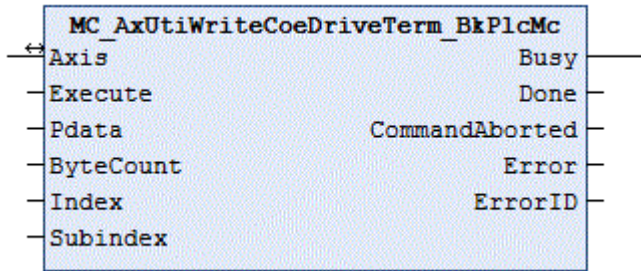
**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn einer der Pointer `ST_TcPlcDeviceInput` [▶ 138] und `ST_TcPlcDeviceOutput` [▶ 140] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.
- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Select** außerhalb des zulässigen Bereichs von 0 bis 63 liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn in den Achsparametern als `nDrive_Type` eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.
- Der Wert in `ST_TcPlcRegDataTable` [▶ 144].`RegDataItem[...].Access` legt fest, wie das Element behandelt wird.
  - 0: Element wird ignoriert.
  - 1: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem ungleichen Inhalt wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
  - 2: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht größer ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
  - 3: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht kleiner ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
  - 4: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht größer oder gleich ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
  - 5: Das durch **Select** adressierte Register wird gelesen. Sein Inhalt wird mit **RegData** verglichen. Bei einem Inhalt der nicht kleiner oder gleich ist wird der Schreibvorgang mit **Error** und **ErrorID:=16#FFFFFFFF** abgebrochen.
  - 10: Das durch **Select** adressierte Register wird mit **RegData** geschrieben.
  - Andere Werte werden derzeit ignoriert. Zukünftige Versionen der Bibliothek werden jedoch möglicherweise weitere Funktionen unterstützen. Ein leeres Element sollte daher immer mit 0 gekennzeichnet werden.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Schreibvorgang auf FALSE gesetzt, wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Done**, **CommandAborted**, **Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.

### 4.5.15 MC\_AxUtiWriteCoeDriveTerm\_BkPlcMc (ab V3.0)



Der Funktionsbaustein schreibt den Inhalt eines Registers der EL-Klemme, die als Antriebsschnittstelle für die Achse dient.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  Pdata:        POINTER TO BYTE:=0;
  ByteCount:    BYTE:=0;
  Index:        WORD:=0;
  Subindex:     BYTE:=0;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.
Pdata	POINTER	Hier ist die Adresse der Variablen anzugeben, deren Inhalt in die Klemme geschrieben werden soll.
ByteCount	BYTE	Hier ist die Größe der Variablen in Bytes anzugeben.
Index	WORD	Hier ist die Adressierung des Parameters in der Klemme anzugeben.
Subindex	BYTE	

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:         Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:         BOOL;
  Done:         BOOL;
  CommandAborted: BOOL;
  Error:        BOOL;
  ErrorID:      UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Schreiben des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

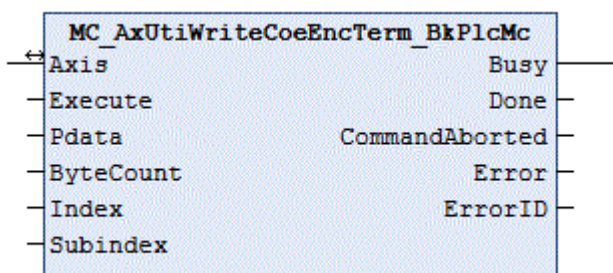
**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Index** oder **Subindex** außerhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn **ByteCount** oder **Pdata** außerhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn in den Achsparametern als nDrive\_Type eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.
- Wenn es bei der ADS-Kommunikation mit der Klemme zu Problemen kommt wird der entsprechende ADS Error Code als **ErrorID** zurückgegeben und dies mit **Error** kenntlich gemacht. Dabei können unter anderen folgende Codes [▶ 321] auftreten:
  - 16#0006 = 6 = Die Portnummer der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0007 = 7 = Die AmsNetID der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0702 = 1794 = dwTcHydAdsErrInvalidIdxGroup = Die Klemme unterstützt nicht das CoE Protokoll.
  - 16#0703 = 1795 = dwTcHydAdsErrInvalidIdxOffset = Die Adresse in Index und Subindex ist in der Klemme nicht unterstützt.
  - 16#0745 = 1861 = dwTcHydAdsErrTimeout = Zeitüberschreitung.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Schreibvorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Done**, **CommandAborted**, **Error**, **ErrorID**) werden für einen Zyklus gegeben.

**4.5.16 MC\_AxUtiWriteCoeEncTerm\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein schreibt den Inhalt eines Registers der EL-Klemme, die als Encoderschnittstelle für die Achse dient.

**Eingänge**

```
VAR_INPUT
  Execute:      BOOL;
  Pdata:       POINTER TO BYTE:=0;
  ByteCount:   BYTE:=0;
  Index:       WORD:=0;
  Subindex:    BYTE:=0;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.

Name	Typ	Beschreibung
Pdata	POINTER	Hier ist die Adresse der Variablen anzugeben, deren Inhalt in die Klemme geschrieben werden soll.
ByteCount	BYTE	Hier ist die Größe der Variablen in Bytes anzugeben.
Index	WORD	Hier ist die Adressierung des Parameters in der Klemme anzugeben.
Subindex	BYTE	

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:           BOOL;
  Done:           BOOL;
  CommandAborted: BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Schreiben des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Index** oder **Subindex** außerhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn **ByteCount** oder **Pdata** außerhalb des zulässigen Bereichs liegen wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn in den Achsparametern als nEncoder\_Type eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.
- Wenn es bei der ADS-Kommunikation mit der Klemme zu Problemen kommt wird der entsprechende ADS Error Code als **ErrorID** zurückgegeben und dies mit **Error** kenntlich gemacht. Dabei können unter anderen folgende Codes [► 321] auftreten:
  - 16#0006 = 6 = Die Portnummer der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0007 = 7 = Die AmsNetID der verwendeten ADS-Adresse ist ungültig: Mapping des InfoData Elements der Klemme überprüfen!
  - 16#0702 = 1794 = dwTcHydAdsErrInvalidIdxGroup = Die Klemme unterstützt nicht das CoE Protokoll.
  - 16#0703 = 1795 = dwTcHydAdsErrInvalidIdxOffset = Die Adresse in Index und Subindex ist in der Klemme nicht unterstützt.

- 16#0745 = 1861 = dwTcHydAdsErrTimeout = Zeitüberschreitung.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Schreibvorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**Done**, **CommandAborted**, **Error**, **ErrorID**) werden für einen Zyklus gegeben.

### 4.5.17 MC\_AxUtiWriteRegDriveTerm\_BkPlcMc (ab V3.0)



Der Funktionsbaustein schreibt den Inhalt eines Registers der KL-Klemme, die als Antriebsschnittstelle für die Achse dient.

#### Eingänge

```
VAR_INPUT
  Execute:      BOOL;
  Select:       INT;
  RegData:      WORD;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.
Select	INT	Hier ist die Registernummer zu übergeben.
RegData	WORD	Hier ist der zu schreibende Wert zu übergeben.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  CommandAborted: BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Schreiben des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn einer der Pointer ST\_TcPlcDeviceInput [▶ 138] und ST\_TcPlcDeviceOutput [▶ 140] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.
- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Select** außerhalb des zulässigen Bereichs von 0 bis 63 liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn in den Achsparametern als nDrive\_Type eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.

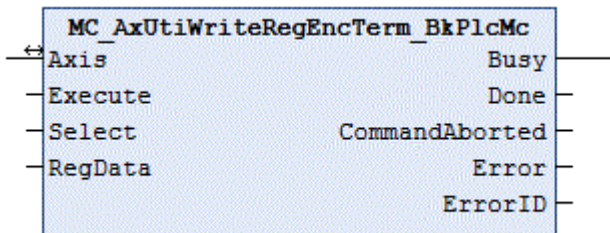
Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird der Schreibvorgang initiiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Schreibvorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**RegData**, **Done**, **CommandAborted**, **Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.



Die Drivetypes iTcMc\_DriveKL2521, iTcMc\_DriveKL4032, iTcMc\_DriveKL2531 und iTcMc\_DriveKL2541 unterstützen die Parameterkommunikation.

**4.5.18 MC\_AxUtiWriteRegEncTerm\_BkPlcMc (ab V3.0)**



Der Funktionsbaustein schreibt den Inhalt eines Registers der KL-Klemme, die als Encoderschnittstelle für die Achse dient.

**Eingänge**

```
VAR_INPUT
    Execute:      BOOL;
    Select:       INT;
    RegData:     WORD;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Schreibvorgang.
Select	INT	Hier ist die Registernummer zu übergeben.
RegData	WORD	Hier ist der zu schreibende Wert zu übergeben.

**Ein-/Ausgänge**

```
VAR_INOUT
    Axis:      Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Busy:          BOOL;
  Done:          BOOL;
  CommandAborted: BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Done	BOOL	Hier wird das erfolgreiche Schreiben des Parameters signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Lesevorgangs signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlerursache bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

- Wenn einer der Pointer `ST_TcPlcDeviceInput` [► 138] und `ST_TcPlcDeviceOutput` [► 140] nicht initialisiert ist wird mit **Error** und **ErrorID:=dwTcHydErrCdPtrPlcMc** reagiert.
- Wenn die Achse für den Betrieb freigegeben ist wird mit **Error** und **ErrorID:=dwTcHydErrCdNotReady** reagiert.
- Wenn **Select** außerhalb des zulässigen Bereichs von 0 bis 63 liegt wird mit **Error** und **ErrorID:=dwTcHydErrCdTblIllegalIndex** reagiert.
- Wenn in den Achsparametern als `nEncoder_Type` eine E/A-Baugruppe eingestellt ist, die keine Parameterkommunikation unterstützt wird mit **Error** und **ErrorID:=dwTcHydErrCdNotCompatible** reagiert.

Wenn diese Überprüfungen ohne Problem durchgeführt werden konnten wird der Schreibvorgang initiiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktivem Schreibvorgang auf FALSE gesetzt wird der eingeleitete Vorgang unbeeinflusst weiter bearbeitet. Die Signale am Ende der Operation (**RegData**, **Done**, **CommandAborted**, **Error**, **ErrorID**, **Done**) werden für einen Zyklus gegeben.



Die Drivetypes `iTcMc_EncoderKL3002`, `iTcMc_EncoderKL3042`, `iTcMc_EncoderKL3062`, `iTcMc_EncoderKL3162`, `iTcMc_EncoderKL5101`, `iTcMc_EncoderKL5111`, `iTcMc_EncoderKL2521`, `iTcMc_EncoderKL2531` und `iTcMc_EncoderKL2541` unterstützen die Parameterkommunikation.

## 4.6 Part 5 Homing

### 4.6.1 FinalizingFunctions

#### 4.6.1.1 MC\_AbortHoming\_BkPlcMc



Der Funktionsbaustein dient zum Abbrechen eines Referenziervorgangs.

#### Eingänge

```
VAR_INPUT
  Execute:          BOOL;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet den Abbruch.

#### Ein-/Ausgänge

```
VAR_INOUT
  Axis:            Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <a href="#">Axis_Ref_BkPlcMc [► 84]</a> zu übergeben.

#### Ausgänge

```
VAR_OUTPUT
  Done:           BOOL;
  Busy:           BOOL;
  Active:         BOOL;
  CommandAborted: BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.



**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface ob eine aktive Bewegung ausgeführt wird. Ist die Achse in keiner aktiven Bewegung, so wird das Referenzieren direkt abgebrochen. Sollte die Achse in einer aktiven Bewegung sein, wird diese Bewegung über einen MC\_Stop\_BkPlcMc [► 81] abgebrochen. Ist der Stop erfolgreich meldet der Baustein **Done**. Entsteht während des Stops ein Fehler so wird dieser Fehler über **Error** und **ErrorId** signalisiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Bewegung auf FALSE gesetzt, wird das eingeleitete Kommando unbeeinflusst weiter bearbeitet

**4.6.1.2 MC\_FinishHoming\_BkPlcMc**



Der Funktionsbaustein dient zum Abbrechen eines referenzier Vorgangs.

**Eingänge**

```
VAR_INPUT
  Execute:          BOOL;
  Distance:         LREAL;
  Velocity:         LREAL;
  Acceleration:    LREAL;
  Deceleration:    LREAL;
  Jerk:            LREAL;
  BufferMode:       MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Bewegung und beendet das Referenzieren.
Distance	LREAL	[mm] Die Entfernung zur Zielposition der Bewegung in Istwert-Einheiten der Achse.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] Die geforderte Ruck in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
BufferMode	MC_BufferMode_BkPlcMc	reserviert

**Ein-/Ausgänge**

```
VAR_INOUT
  Axis:            Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis sLog	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">► 84</a> ] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Done:          BOOL;
  Busy:          BOOL;
  Active:        BOOL;
  CommandAborted:  BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin werden die Parameter **Distance, Velocity, Acceleration** und **Deceleration** an dem Funktionsblock MC\_MoveRelative\_BkPlcMc [[► 76](#)] übergeben. Nach Überprüfung der übergebenen Parameter wird die Bewegung ausgeführt.

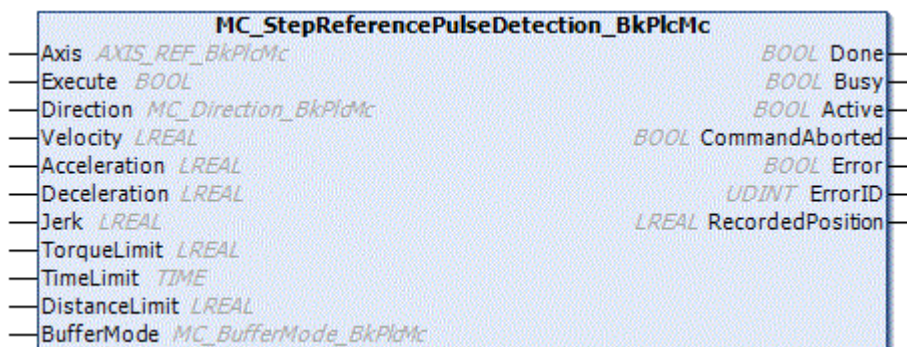
Sollte während der Abarbeitung der Bewegung vom Bewegungsalgorithmus ein Fehlercode signalisiert werden, wird mit **Error** und **ErrorID:=Bewegungsalgorithmus-Fehlercode** reagiert. Wird die vollständige Bewegung durch die Aktivität eines anderen Bausteins verhindert, wird mit **CommandAborted** reagiert. Erreicht der Bewegungsalgorithmus die Zielbedingungen der Achse, wird mit **Done** reagiert.

Eine fallende Flanke an **Execute** löscht alle anstehenden Ausgangssignale. Wird **Execute** bereits bei noch aktiver Bewegung auf FALSE gesetzt, wird die eingeleitete Bewegung unbeeinflusst weiter bearbeitet. Die Signale am Ende der Bewegung (**Error, ErrorID, CommandAborted, Done**) werden für einen Zyklus gegeben.

Die Achse befindet sich während der Bewegung im Zustand McState\_Homing [[► 96](#)], am Ende wechselt der Zustand nach McState\_Standstill [[► 96](#)].

## 4.6.2 StepFunctions

### 4.6.2.1 MC\_StepAbsoluteSwitch\_BkPlcMc



Der Funktionsbaustein dient zum Referenzieren über einen Endschalter. Dabei löst der Baustein intern nach auffinden der Nocke ein Positionsetzen aus.

 **Eingänge**

```

VAR_INPUT
  Execute:          BOOL;
  Direction:        MC_Direction_BkPlcMc;
  SwitchMode:       MC_SwitchMode_BkPlcMc;
  ReferenceSignal:  MC_Ref_Signal_Ref_BkPlcMc;
  Velocity:         LREAL;
  Acceleration:     LREAL;
  Deceleration:     LREAL;
  Jerk:             LREAL;
  SetPosition:      LREAL;
  TorqueLimit:      LREAL;
  TimeLimit:        TIME;
  DistanceLimit:    LREAL;
  BufferMode:        MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Referenzierfahrt.
Direction	MC_Direction_BkPlcMc	Über MC_Direction_BkPlcMc [▶ 115] wird die Richtung vorgegeben.
SwitchMode	MC_SwitchMode_BkPlcMc	Über MC_SwitchMode_BkPlcMc [▶ 115] wird die Art der Signalerkennung vorgegeben.
ReferenceSignal	MC_Ref_Signal_Ref_BkPlcMc	Über MC_Ref_Signal_Ref_BkPlcMc [▶ 149] wird der Signalzustand der Nocke mitgeteilt.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] Die geforderte Ruck in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
SetPosition	LREAL	Geforderte Position an der Referenziernocke.
TorqueLimit	LREAL	reserviert
TimeLimit	TIME	Nach dieser Zeit wird die Referenzierung mit Fehler abgerochen. Bei null ist die Zeitüberwachung deaktiviert.
DistanceLimit	LREAL	Nach dieser Wegstrecke wird die Referenzierung mit Fehler abgerochen. Bei null ist die Wegstreckenüberwachung deaktiviert.
BufferMode	MC_BufferMode_BkPlcMc	reserviert

 **Ein-/Ausgänge**

```

VAR_INOUT
  Axis:             Axis_Ref_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ Axis_Ref_BkPlcMc [▶ 84] zu übergeben.

**Ausgänge**

```
VAR_OUTPUT
  Done:          BOOL;
  Busy:          BOOL;
  Active:        BOOL;
  CommandAborted:  BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
END_VAR
```

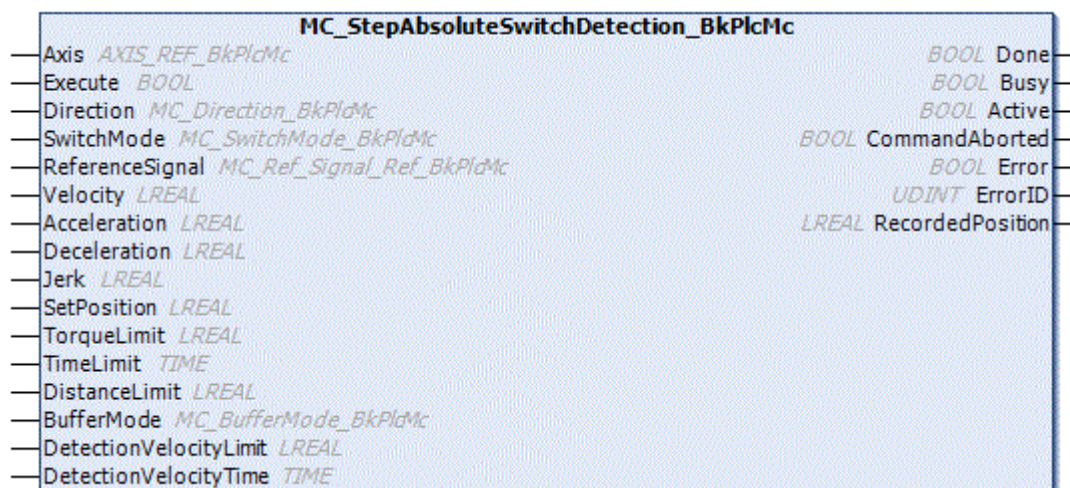
Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin werden die Parameter **Direction, SwitchMode, ReferenceSignal, Velocity, Acceleration** und **Deceleration** an dem Funktionsblock [MC\\_StepAbsoluteSwitchDetection\\_BkPlcMc](#) [► 292] übergeben. Bei erfolgreicher Abarbeitung des internen Bausteins [MC\\_StepAbsoluteSwitchDetection\\_BkPlcMc](#) wird die ermittelte Position über [MC\\_SetPosition\\_BkPlcMc](#) [► 43] entsprechend gesetzt.

Während der Bearbeitung meldet der Baustein **Busy** und **Active**. Nach erfolgreichem Positionsetzen wird **Done** zurückgemeldet. Meldet ein unterlagerter Baustein einen Fehler so wird dieser über **Error** und **ErrorID** zurückgemeldet.

**4.6.2.2 MC\_StepAbsoluteSwitchDetection\_BkPlcMc**



Der Funktionsbaustein dient zum Referenzieren über einen Endschalter. Dabei gibt der Baustein die Position der Nocke über RecordedPosition nach außen. Es wird keine Position gesetzt.

**Eingänge**

```
VAR_INPUT
  Execute:          BOOL;
  Direction:        MC_Direction_BkPlcMc;
  SwitchMode:       MC_SwitchMode_BkPlcMc;
  ReferenceSignal:  MC_Ref_Signal_Ref_BkPlcMc;
  Velocity:         LREAL;
  Acceleration:     LREAL;

```

```

Deceleration:      LREAL;
Jerk:              LREAL;
SetPosition:      LREAL;
TorqueLimit:      LREAL;
TimeLimit:        TIME;
DistanceLimit:    LREAL;
BufferMode:       MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Referenzierfahrt.
Direction	MC_Direction_BkPlcMc	Über <u>MC_Direction_BkPlcMc</u> [► 115] wird die Richtung vorgegeben.
SwitchMode	MC_SwitchMode_BkPlcMc	Über <u>MC_SwitchMode_BkPlcMc</u> [► 115] wird die Art der Signalerkennung vorgegeben.
ReferenceSignal	MC_Ref_Signal_Ref_BkPlcMc	Über <u>MC_Ref_Signal_Ref_BkPlcMc</u> [► 149] wird der Signalzustand der Nocke mitgeteilt.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] Die geforderte Ruck in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
SetPosition	LREAL	Geforderte Position an der Referenziernocke.
TorqueLimit	LREAL	reserviert
TimeLimit	TIME	Nach dieser Zeit wird die Referenzierung mit Fehler abgerochen. Bei Null ist die Zeitüberwachung deaktiviert.
DistanceLimit	LREAL	Nach dieser Wegstrecke wird die Referenzierung mit Fehler abgerochen. Bei Null ist die Wegstreckenüberwachung deaktiviert.
BufferMode	MC_BufferMode_BkPlcMc	

 **Ein-/Ausgänge**

```

VAR_INOUT
  Axis:          Axis_Ref_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```

VAR_OUTPUT
  Done:          BOOL;
  Busy:          BOOL;
  Active:        BOOL;
  CommandAborted:  BOOL;
  Error:         BOOL;
  ErrorID:       UDINT;
  RecordedPosition: LREAL;
END_VAR
    
```

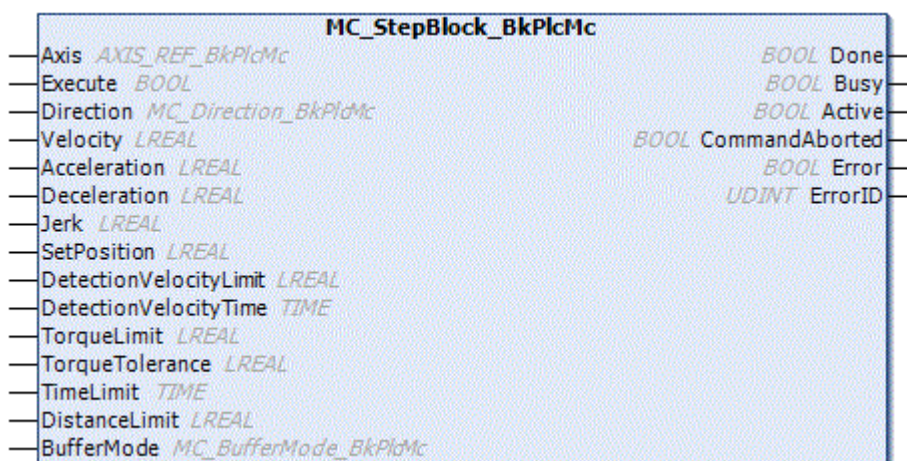
Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.
RecordedPosition	LREAL	[mm] Ermittelte Position der Referenziernocke.

### Verhalten des Bausteins

Auf eine steigende Flanke an **Execute** hin wird beim Parameter **Direction** lediglich MC\_Positive\_Direction\_BkPlcMc, MC\_Negative\_Direction\_BkPlcMc, MC\_SwitchPositive\_Direction\_BkPlcMc, MC\_SwitchNegative\_Direction\_BkPlcMc akzeptiert. Die Parameter **Velocity**, **Acceleration**, **Deceleration** und **Jerk** werden an MC\_MoveVelocity\_BkPlcMc [► 78] übergeben. Nachdem die Nocke erkannt wurde, wird die Position über **RecordedPosition** mitgeteilt und ein MC\_Halt\_BkPlcMc [► 67] bricht die Bewegung ab.

Während der Bearbeitung meldet der Baustein **Busy** und **Active**. Nach erfolgreicher Bearbeitung wird **Done** zurückgemeldet. Meldet ein unterlagerter Baustein ein Fehler so wird dieser über **Error** und **ErrorID** zurückgemeldet.

### 4.6.2.3 MC\_StepBlock\_BkPlcMc



Der Funktionsbaustein dient zum Referenzieren über einen Festanschlag. Dabei löst der Baustein intern nach Auffinden der Nocke ein Positionssetzen aus.

#### Eingänge

```

VAR_INPUT
    Execute:          BOOL;
    Direction:        MC_Direction_BkPlcMc;
    Velocity:         LREAL;
    Acceleration:     LREAL;
    Deceleration:     LREAL;
    Jerk:             LREAL;
    SetPosition:      LREAL;
    DetectionVelocityLimit: LREAL;
    DetectionVelocityTime: LREAL;
    TorqueLimit:      LREAL;
    TorqueTolerance:  LREAL;
    TimeLimit:        TIME;
    DistanceLimit:    LREAL;
    BufferMode:        MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR

```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Referenzierfahrt.
Direction	MC_Direction_BkPlcMc	Über <u>MC_Direction_BkPlcMc</u> [► 115] wird die Richtung vorgegeben.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] Die geforderte Ruck in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
SetPosition	LREAL	Geforderte Position an der Referenziernocke.
DetectionVelocityLimit	LREAL	[mm/s] Geschwindigkeitsgrenze, um einen Anschlag sicher zu erkennen
DetectionVelocityTime	LREAL	[s] Zeit in der die Istgeschwindigkeit unterhalb von der DetectionVelocityLimit liegen muss, um den Festanschlag sicher zu erkennen.
TorqueLimit	LREAL	[Bar] Limitierung für den Druck
TorqueTolerance	LREAL	[Bar] Toleranz für den Druck
TimeLimit	TIME	Nach dieser Zeit wird die Referenzierung mit Fehler abgerochen. Bei null ist die Zeitüberwachung deaktiviert.
DistanceLimit	LREAL	Nach dieser Wegstrecke wird die Referenzierung mit Fehler abgebrochen. Bei null ist die Wegstreckenüberwachung deaktiviert.
BufferMode	MC_BufferMode_BkPlcMc	reserviert

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Done:           BOOL;
  Busy:           BOOL;
  Active:         BOOL;
  CommandAborted: BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.

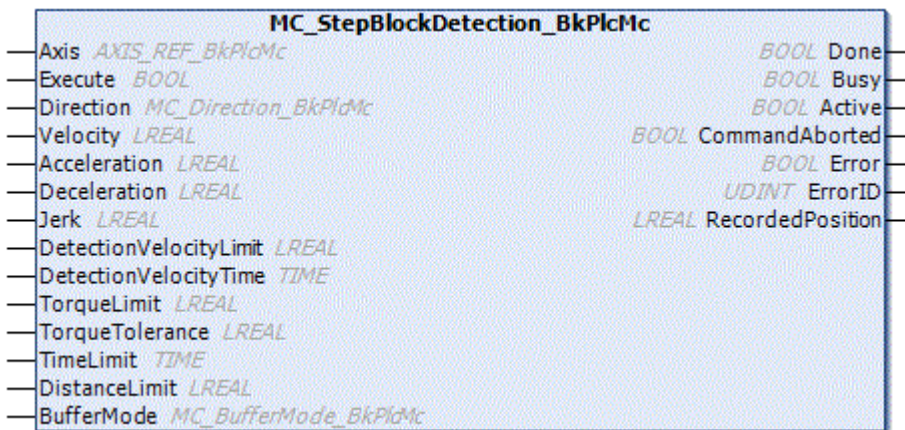
Name	Typ	Beschreibung
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin werden alle Parameter den Funktionsblock MC\_StepBlockDetection\_BkPlcMc [► 296] übergeben. Bei erfolgreicher Abarbeitung des internen Bausteins MC\_StepBlockDetection\_BkPlcMc wird die ermittelte Position über MC\_SetPosition\_BkPlcMc [► 43] entsprechend gesetzt.

Während der Bearbeitung meldet der Baustein **Busy** und **Active**. Nach erfolgreichem Positionsetzen wird **Done** zurückgemeldet. Meldet ein unterlagerter Baustein ein Fehler so wird dieser über **Error** und **ErrorID** zurückgemeldet.

**4.6.2.4 MC\_StepBlockDetection\_BkPlcMc**



Der Funktionsbaustein dient zum Referenzieren über einen Endschalter. Dabei gibt der Baustein die Position der Nocke über **RecordedPosition** nach außen. Es wird keine Position gesetzt.

**Eingänge**

```

VAR_INPUT
  Execute:          BOOL;
  Direction:        MC_Direction_BkPlcMc;
  Velocity:         LREAL;
  Acceleration:    LREAL;
  Deceleration:    LREAL;
  Jerk:            LREAL;
  SetPosition:     LREAL;
  DetectionVelocityLimit: LREAL;
  DetectionVelocityTime:  LREAL;
  TorqueLimit:     LREAL;
  TorqueTolerance: LREAL;
  TimeLimit:       TIME;
  DistanceLimit:   LREAL;
  BufferMode:       MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Referenzierfahrt.
Direction	MC_Direction_BkPlcMc	Eine entsprechend <u>MC_Direction_BkPlcMc</u> [► 115] kodierte Richtungsvorgabe.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.



Name	Typ	Beschreibung
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] Die geforderte Ruck in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
SetPosition	LREAL	Geforderte Position an der Referenziernocke.
DetectionVelocityLimit	LREAL	[mm/s] Geschwindigkeitsgrenze, um einen Anschlag sicher zu erkennen
DetectionVelocityTime	LREAL	[s] Zeit in der die Istgeschwindigkeit unterhalb von der DetectionVelocityLimit liegen muss um den Festanschlag sicher zu erkennen.
TorqueLimit	LREAL	[Bar] Limitierung für den Druck
TorqueTolerance	LREAL	[Bar] Toleranz für den Druck
TimeLimit	TIME	Nach dieser Zeit wird die Referenzierung mit Fehler abgerochen. Bei Null ist die Zeitüberwachung deaktiviert.
DistanceLimit	LREAL	Nach dieser Wegstrecke wird die Referenzierung mit Fehler abgerochen. Bei Null ist die Wegstreckenüberwachung deaktiviert.
BufferMode	MC_BufferMode_BkPlcMc	

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:                Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <code>Axis_Ref_BkPlcMc</code> [► 84] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Done:                BOOL;
  Busy:                BOOL;
  Active:              BOOL;
  CommandAborted:     BOOL;
  Error:               BOOL;
  ErrorID:             UDINT;
  RecordedPosition:   LREAL;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.
RecordedPosition	LREAL	[mm] Ermittelte Position der Referenziernocke.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können eine Reihe von Problemen erkannt und gemeldet werden:

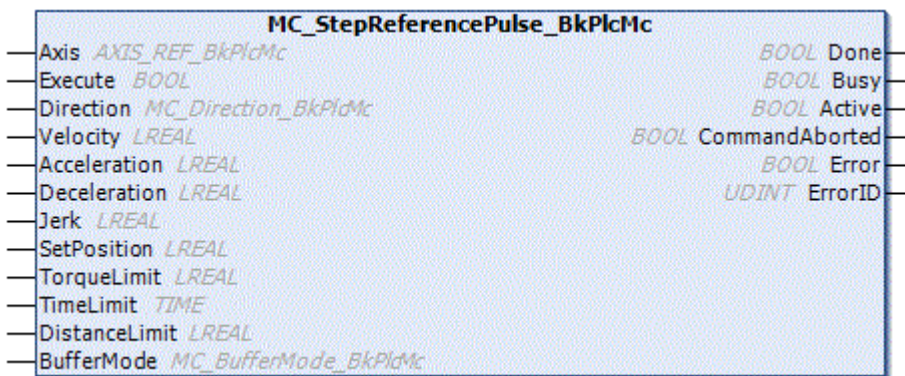
- Bei der **Direction** wird MC\_Positive\_Direction\_BkPlcMc, MC\_Negative\_Direction\_BkPlcMc, akzeptiert
- Die DistanceLimit muss einen Wert größer Inkrementauflösung haben.

Die Parameter **Velocity**, **Acceleration**, **Deceleration** und **Jerk** werden an MC\_MoveVelocity\_BkPlcMc [► 78] übergeben. Nachdem die Nocke erkannt wurde, wird die Position über **RecordedPosition** mitgeteilt und ein MC\_Halt\_BkPlcMc [► 67] hält die Bewegung an.

Ein Festanschlag wird erkannt, wenn die Istgeschwindigkeit für die Zeit **DetectionVelocityTime** unterhalb **DetectionVelocityLimit** ist oder der Aktuelle Druck Größe als **TorqueLimit** minus **TorqueTolerance**. Wird innerhalb von **DistanceLimit** oder **TimeLimit** kein Festanschlag erkannt so wird die Referenzierung mit Fehler abgebrochen.

Während der Bearbeitung meldet der Baustein **Busy** und **Active**. Nach erfolgreicher Bearbeitung wird **Done** zurückgemeldet. Meldet ein unterlagerter Baustein ein Fehler so wird dieser über **Error** und **ErrorID** zurückgemeldet.

**4.6.2.5 MC\_StepReferencePulse\_BkPlcMc**



Der Funktionsbaustein dient zum Referenzieren eines inkrementellen Wegmesssystems. Über den Referenzierpuls wird ein Istwertsetzen intern durchgeführt.

**Eingänge**

```

VAR_INPUT
    Execute:          BOOL;
    Direction:        MC_Direction_BkPlcMc;
    Velocity:         LREAL;
    Acceleration:     LREAL;
    Deceleration:     LREAL;
    Jerk:             LREAL;
    SetPosition:     LREAL;
    TorqueLimit:     LREAL;
    TimeLimit:       TIME;
    DistanceLimit:   LREAL;
    BufferMode:       MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Referenzierfahrt.
Direction	MC_Direction_BkPlcMc	Über MC_Direction_BkPlcMc wird die Richtung vorgegeben.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.

Name	Typ	Beschreibung
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] Die geforderte Ruck in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
SetPosition	LREAL	Geforderte Position an der Referenziernocke.
TorqueLimit	LREAL	reserviert
TimeLimit	TIME	Nach dieser Zeit wird die Referenzierung mit Fehler abgerochen. Bei null ist die Zeitüberwachung deaktiviert.
DistanceLimit	LREAL	Nach dieser Wegstrecke wird die Referenzierung mit Fehler abgerochen. Bei null ist die Wegstreckenüberwachung deaktiviert.
BufferMode	MC_BufferMode_BkPlcMc	reserviert

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Done:           BOOL;
  Busy:           BOOL;
  Active:         BOOL;
  CommandAborted: BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
END_VAR
```

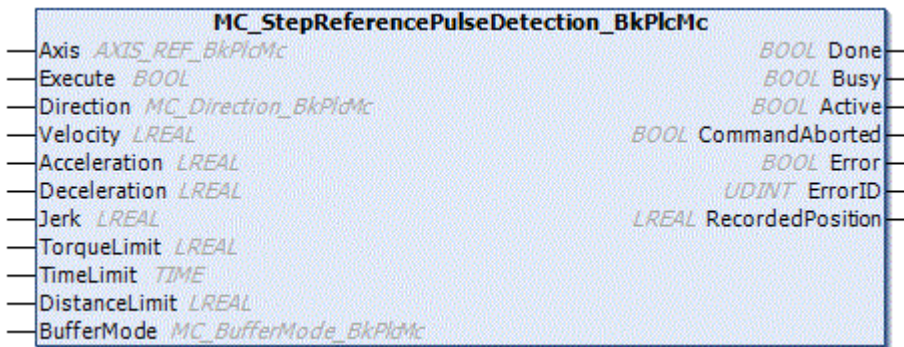
Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin werden die Parameter **Direction, Velocity, Acceleration** und **Deceleration** an dem Funktionsblock MC\_StepReferencePulseDetection\_BkPlcMc [[▶ 300](#)] übergeben. Bei erfolgreicher Abarbeitung des internen Bausteins MC\_StepReferencePulseDetection\_BkPlcMc wird die ermittelte Position über MC\_SetPosition\_BkPlcMc [[▶ 43](#)] entsprechend gesetzt.

Während der Bearbeitung meldet der Baustein **Busy** und **Active**. Nach erfolgreichem Positionssetzen wird **Done** zurückgemeldet. Meldet ein unterlagerter Baustein ein Fehler so wird dieser über **Error** und **ErrorID** zurückgemeldet.

#### 4.6.2.6 MC\_StepReferencePulseDetection\_BkPlcMc



Der Funktionsbaustein dient zum Referenzieren eines inkrementellen Wegmesssystems. Dabei gibt der Baustein die Position des Referenzpuls über **RecordedPosition** nach außen. Es wird keine Position gesetzt.

#### Eingänge

```

VAR_INPUT
    Execute:          BOOL;
    Direction:        MC_Direction_BkPlcMc;
    Velocity:         LREAL;
    Acceleration:     LREAL;
    Deceleration:     LREAL;
    Jerk:             LREAL;
    SetPosition:      LREAL;
    TorqueLimit:      LREAL;
    TimeLimit:        TIME;
    DistanceLimit:    LREAL;
    BufferMode:        MC_BufferMode_BkPlcMc:=Aborting_BkPlcMc;
END_VAR
    
```

Name	Typ	Beschreibung
Execute	BOOL	Eine steigende Flanke an diesem Eingang startet die Referenzierfahrt.
Direction	MC_Direction_BkPlcMc	Über <i>MC_Direction_BkPlcMc</i> [► 115] wird die Richtung vorgegeben.
Velocity	LREAL	[mm/s] Die geforderte Geschwindigkeit der Bewegung in Istwert-Einheiten der Achse pro Sekunde.
Acceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Beschleunigung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Deceleration	LREAL	[mm/s <sup>2</sup> ] Die geforderte Verzögerung in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
Jerk	LREAL	[mm/s <sup>3</sup> ] Die geforderte Ruck in Istwert-Einheiten der Achse pro Quadrat-Sekunde. Ist dieser Parameter 0.0 wird er durch einen Defaultwert aus den Achsparametern ersetzt.
SetPosition	LREAL	Geforderte Position an der Referenziernocke.
TorqueLimit	LREAL	reserviert
TimeLimit	TIME	Nach dieser Zeit wird die Referenzierung mit Fehler abgerochen. Bei null ist die Zeitüberwachung deaktiviert.

Name	Typ	Beschreibung
DistanceLimit	LREAL	Nach dieser Wegstrecke wird die Referenzierung mit Fehler abgerochen. Bei null ist die Wegstreckenüberwachung deaktiviert.
BufferMode	MC_BufferMode_BkPlcMc	reserviert

 **Ein-/Ausgänge**

```
VAR_INOUT
  Axis:           Axis_Ref_BkPlcMc;
END_VAR
```

Name	Typ	Beschreibung
Axis	Axis_Ref_BkPlcMc	Hier ist die Adresse einer Variablen vom Typ <u>Axis_Ref_BkPlcMc</u> [ <a href="#">▶ 84</a> ] zu übergeben.

 **Ausgänge**

```
VAR_OUTPUT
  Done:           BOOL;
  Busy:           BOOL;
  Active:         BOOL;
  CommandAborted: BOOL;
  Error:          BOOL;
  ErrorID:        UDINT;
  RecordedPosition: LREAL;
END_VAR
```

Name	Typ	Beschreibung
Done	BOOL	Hier wird die erfolgreiche Abarbeitung signalisiert
Busy	BOOL	Hier wird signalisiert, dass ein Kommando abgearbeitet wird.
Active	BOOL	Hier wird die Betriebsbereitschaft signalisiert.
CommandAborted	BOOL	Hier wird ein Abbruch des Kommandos signalisiert.
Error	BOOL	Hier wird das Auftreten eines Fehlers signalisiert.
ErrorID	UDINT	Hier wird eine codierte Fehlermeldung bereitgestellt.
RecordedPosition	LREAL	[mm] Ermittelte Position am Referenzpuls.

**Verhalten des Bausteins**

Auf eine steigende Flanke an **Execute** hin untersucht der Baustein das übergebene Achsinterface. Dabei können Probleme erkannt und gemeldet werden:

- Bei der **Direction** wird MC\_Positive\_Direction\_BkPlcMc, MC\_Negative\_Direction\_BkPlcMc, akzeptiert

Die Parameter **Velocity**, **Acceleration**, **Deceleration** und **Jerk** werden an MC\_MoveVelocity\_BkPlcMc [[▶ 78](#)] übergeben. Nachdem die Nocke erkannt wurde, wird die Position über **RecordedPosition** mitgeteilt und ein MC\_Halt\_BkPlcMc [[▶ 67](#)] hält die Bewegung an. Wird innerhalb von **DistanceLimit** oder **TimeLimit** keine Referenzpuls erkannt so wird die Referenzierung mit Fehler abgebrochen

Während der Bearbeitung meldet der Baustein **Busy** und **Active**. Nach erfolgreichem Positionsetzen wird **Done** zurückgemeldet. Meldet ein unterlagerter Baustein ein Fehler so wird dieser über **Error** und **ErrorID** zurückgemeldet.

## 5 Knowledge Base

### Knowledge Base der SPS Bibliothek TcPlcHydraulics (ab V3.0)

Hier finden Sie eine Reihe von Antworten auf immer wiederkehrende Fragen.

#### Themengebiete

Name	Beschreibung
<a href="#">Global Constants [▶ 319]</a>	Vordefinierte Fehlercodes, Masken zur Bitabfrage, ADS-Codes usw.
Setup	Inbetriebnahme-Hinweise
<a href="#">SampleList [▶ 353]</a>	Programm-Beispiele
<a href="#">Ideensammlung [▶ 303]</a>	Tipps und Tricks
<a href="#">HMI-Tool [▶ 349]</a>	Der PlcMcManager

#### Probleme beim Update der Bibliothek

Bei einem Update der Bibliothek kann es zu Problemen beim Übersetzen kommen. Die Ursache ist dann möglicherweise eine Änderung des Namens eines oder mehrerer Bausteine oder Datentypen. Diese Änderungen sind nicht immer vermeidbar und werden in der Regel aus einem der folgenden Gründe durchgeführt:

- Anpassung an die Regeln der PLC Open Motion Control Definitionen.
- Verfolgen der Weiterentwicklung der PLC Open Motion Control Definitionen.
- Weiterentwicklung der bereitgestellten Technologie.
- Anpassung an die verwendete Technologie, insbesondere Unterstützung weiterer E/A-Geräte.
- Vermeidung von Namenskollisionen und anderen Kompatibilitätsproblemen mit anderen Bibliotheken.

Ab V3.0 Build 22 verwendet die Bibliothek die TcEtherCAT.LIB zur Kommunikation via EtherCAT-Feldbus. In älteren TwinCAT-Umgebungen ist diese Bibliothek noch nicht verfügbar. Soll die Bibliothek TcPlcHydraulics in einer solchen Umgebung verwendet werden ist die mitgelieferte TcEtherCatDummy.LIB in das Projektverzeichnis kopiert und in TcEtherCAT.LIB umbenannt werden. Diese Bibliothek ist dann **VOR** der TcPlcHydraulics.LIB im Projekt einzufügen.

**i** Diese Vorgehensweise darf nicht in TwinCAT-Umgebungen angewendet werden, die EtherCAT unterstützen. Die mitgelieferte Datei darf **NICHT** dazu verwendet werden, eine vorhandene funktionstüchtige TcEtherCAT.LIB zu ersetzen.

Es sind keine Funktionen verfügbar, die EtherCAT Technologien voraussetzen.

**i** Die in einem Projekt verwendete Version der Bibliothek sollte in das Projektverzeichnis kopiert und mit dem Projekt zusammen gesichert werden. Dadurch wird ein unbeabsichtigter Versionswechsel vermieden, der sonst durch ein in der Zwischenzeit durchgeführtes TwinCAT Update verursacht werden könnte. Ist ein Update der Bibliothek gewollt ist dann die neue Version gezielt in das Projektverzeichnis zu kopieren.

**i** Es wird dringend empfohlen, nach einem Update der Bibliothek eine probeweise Übersetzung des gesamten Projekts durchzuführen. Weiterhin sollte mit dem System Manager das Mapping aktualisiert werden. Sollte die unten stehende Tabelle eine Größenänderung einer der Strukturen anzeigen ist unbedingt die Adressvergabe zu überprüfen.

**i** Wenn ein Update der Bibliothek auf einen Stand erfolgt, der sich nicht nur in der dritten (Build), sondern in den ersten zwei (Major, Minor) Versionsangaben unterscheidet ist davon auszugehen, dass die vom System Manager erzeugten Mappings nicht mehr korrekt sind. Es ist zwingend erforderlich, die Verknüpfungen neu wirksam zu machen.

Alter Name	Neuer Name	Grund der Änderung
ST_TcMcAxInterface	Axis_Ref_BkPlcMc	Anpassung an PLC Open Motion Control Definitionen.
ST_TcPlcMcCamId	MC_CAM_ID_BkPlcMc	Anpassung an PLC Open Motion Control Definitionen.
ST_TcPlcMcCamRef	MC_CAM_REF_BkPlcMc	Anpassung an PLC Open Motion Control Definitionen.
E_TcMCDirection	MC_Direction_BkPlcMc	Anpassung an PLC Open Motion Control Definitionen.
E_TcMCStartMode	MC_StartMode_BkPlcMc	Anpassung an PLC Open Motion Control Definitionen.
ST_TcPlcMcEncoderIn	---	entfällt, Aufgabe wird von ST_TcPlcDeviceInput übernommen
ST_TcPlcMcEncoderOut	---	entfällt, Aufgabe wird von ST_TcPlcDeviceOutput übernommen
ST_TcPlcMcDriveIn	---	entfällt, Aufgabe wird von ST_TcPlcDeviceInput übernommen
ST_TcPlcMcDriveOut	---	entfällt, Aufgabe wird von ST_TcPlcDeviceOutput übernommen
ST_TcPlcMcAx2000In	---	entfällt, Aufgabe wird von ST_TcPlcDeviceInput übernommen
ST_TcPlcMcAx2000Out	---	entfällt, Aufgabe wird von ST_TcPlcDeviceOutput übernommen
MC_AxUtiCancelMonitoring_BkPlcMc	---	entfällt, durch PLC Open Definitionen überflüssig geworden.

**Größe der E/A-Strukturen in Bytes**

Name	V 2.1.X	ab V3.0.0	ab V3.1.0 (geplant)
ST_TcPlcMcEncoderIn	16	-	-
ST_TcPlcMcEncoderOut	1	-	-
ST_TcPlcMcDriveIn	23	-	-
ST_TcPlcMcDriveOut	40	-	-
ST_TcPlcMcAx2000In	37	-	-
ST_TcPlcMcAx2000Out	26	-	-
ST_TcPlcDeviceInput <a href="#">[▶ 138]</a>	-	143	?
ST_TcPlcDeviceOutput <a href="#">[▶ 140]</a>	-	103	?

**5.1 FAQs (ab V3.0)**

Hier finden Sie eine Reihe von Antworten auf immer wieder auftretende Fragen.

Name	Beschreibung
<a href="#">FAQ #1 [▶ 304]</a>	Wie integriere ich eine oder mehrere Achsen in eine PLC-Applikation?
<a href="#">FAQ #2 [▶ 305]</a>	Welche Daten müssen in der PLC-Applikation für die Achsen angelegt werden?
<a href="#">FAQ #3 [▶ 305]</a>	Wie initialisiere ich die Daten und lade ich die Parameter einer Achse beim Start der PLC?

Name	Beschreibung
<a href="#">FAQ #4</a> [ <a href="#">▶ 306</a> ]	Wie wird die Istposition der Achsen ermittelt?
<a href="#">FAQ #5</a> [ <a href="#">▶ 308</a> ]	Wie wird der Stellwert für eine Achse erzeugt?
<a href="#">FAQ #6</a> [ <a href="#">▶ 308</a> ]	Wie wird der Stellwert einer Achse für die Ausgabe vorbereitet?
<a href="#">FAQ #7</a> [ <a href="#">▶ 309</a> ]	Wie wird der Stellwert an die Achse ausgegeben?
<a href="#">FAQ #8</a> [ <a href="#">▶ 310</a> ]	In welcher Reihenfolge sind die Bausteine einer Achse aufzurufen?
<a href="#">FAQ #9</a> [ <a href="#">▶ 310</a> ]	Wie steuere ich eine Ventilstufe (Onboard oder extern) an?
<a href="#">FAQ #10</a> [ <a href="#">▶ 311</a> ]	Wie lege ich einen Meldungspuffer an?
<a href="#">FAQ #11</a> [ <a href="#">▶ 311</a> ]	Wie breche ich die Überwachung einer Funktion ab?
<a href="#">FAQ #12</a> [ <a href="#">▶ 312</a> ]	Wie überwache ich die Kommunikation mit einem E/A-Gerät?
<a href="#">FAQ #13</a> [ <a href="#">▶ 312</a> ]	Wie verseehe ich die kundenspezifischen Achsparameter mit selbstgewählten Bezeichnungen?
<a href="#">FAQ #14</a> [ <a href="#">▶ 312</a> ]	Wie steuere ich ein Stromventil an?
<a href="#">FAQ #15</a> [ <a href="#">▶ 313</a> ]	Welche Variablen einer Achse sollten mit dem Scope aufgezeichnet werden?
<a href="#">FAQ #16</a> [ <a href="#">▶ 313</a> ]	Welche Bedeutung hat die Variable nDebugTag in Axis_Ref_BkPlcMc?
<a href="#">FAQ #17</a> [ <a href="#">▶ 313</a> ]	Welche Besonderheiten sind beim Einsatz von Sercos-Antrieben zu berücksichtigen?
<a href="#">FAQ #18</a> [ <a href="#">▶ 314</a> ]	Wie wird ein Druck oder eine Kraft ermittelt?
<a href="#">FAQ #19</a> [ <a href="#">▶ 314</a> ]	Welche Besonderheiten sind beim Einsatz von AX5000 Antrieben zu berücksichtigen?
<a href="#">FAQ #20</a> [ <a href="#">▶ 315</a> ]	Wie wird eine Achse für die Nutzung des Blendings nach PLC Open vorbereitet?
<a href="#">FAQ #21</a> [ <a href="#">▶ 317</a> ]	Wie kann auf Register einer Klemme zugegriffen werden an der ein Encoder oder ein Ventil einer Achse angeschlossen ist?
<a href="#">FAQ #22</a> [ <a href="#">▶ 317</a> ]	Wie ist die ASCII Datei für eine Linearisierungstabelle aufgebaut?
<a href="#">FAQ #23</a> [ <a href="#">▶ 318</a> ]	Wie können Kommandos des PlcMcManagers abgeblockt werden?
<a href="#">FAQ #24</a> [ <a href="#">▶ 319</a> ]	Welches Format haben Import / Export-Dateien mit Kennlinien-Daten.
Setup	Wie wird die Achse in Betrieb genommen und optimiert?

### FAQ #1 Wie integriere ich eine oder mehrere Achsen in eine PLC-Applikation?

Die Vorgehensweise ist grundsätzlich anders als bei einer von der NC Task geführten Achse, weil hier alle Aufgaben der NC Task von der PLC mit übernommen werden. Für die meisten Themen stehen jedoch vorgefertigte Bausteine zur Verfügung, so dass sich der zusätzliche Programmieraufwand in Grenzen hält. Im Einzelnen sind die folgenden Punkte zu beachten:

- Daten der Achse in der PLC-Applikation ([FAQ #2](#) [[▶ 305](#)])
- Initialisierung und Laden der Achs-Parameter beim Start der PLC-Applikation ([FAQ #3](#) [[▶ 305](#)])
- Istwerterfassung ([FAQ #4](#) [[▶ 306](#)])
- Stellwert-Generierung ([FAQ #5](#) [[▶ 308](#)])
- Aufbereitung des Stellwerts für die Ausgabe ([FAQ #6](#) [[▶ 308](#)])
- Inbetriebnahme der Achse (Setup)
- Inbetriebnahme einer Istdruckermittlung mit Bausteinen vom Typ [MC\\_AxRtReadPressureSingle\\_BkPlcMc](#) [[▶ 207](#)] oder [MC\\_AxRtReadPressureDiff\\_BkPlcMc](#) [[▶ 204](#)].
- Organisation des Bewegungsablaufs ([FAQ #7](#) [[▶ 309](#)])



Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollte zur Vereinfachung ein Baustein des Typs [MC\\_AxStandardBody\\_BkPlcMc](#) verwendet werden.



## FAQ #2 Welche Daten müssen in der PLC-Applikation für die Achse angelegt werden?

Pro Achse muss je eine Variable der Typen [Axis\\_Ref\\_BkPlcMc](#) [▶ 84], [ST\\_TcPlcDeviceInput](#) [▶ 138] und [ST\\_TcPlcDeviceOutput](#) [▶ 140] angelegt werden. Für mehrere Achsen ist die Verwendung von Variablenfeldern unbedingt zu empfehlen. Je ein Beispiel für eine und fünf Achsen finden Sie in den ersten Programm-Beispielen.

Die in diesen Beispielen gezeigte Vorgehensweise unter Verwendung von [MC\\_AxUtiStandardInit\\_BkPlcMc](#) [▶ 238] Bausteinen sorgt für eine korrekte Initialisierung beim Start der PLC und veranlasst das Laden der Parameter der Achsen aus Dateien.

Für die Realisierung einer Meldungsaufzeichnung sind weitere Daten erforderlich. Siehe hierzu [FAQ #10](#) [▶ 311].

Um im PlcMcManager kundenspezifische Achsparameter mit selbstgewählten Bezeichnungen zu versehen sind weitere Daten erforderlich. Siehe hierzu [FAQ #13](#) [▶ 312].

Um das Blending nach PLC Open nutzen zu können sind weitere Daten erforderlich. Siehe hierzu [FAQ #20](#) [▶ 315].

## FAQ #3 Wie initialisiere ich die Daten einer Achse?

Beim Start der PLC Applikation sind eine Reihe von Initialisierungen vorzunehmen. Dies geschieht sinnvollerweise in drei Stufen, die von einem [MC\\_AxUtiStandardInit\\_BkPlcMc](#) [▶ 238] Baustein bereitgestellt werden und nur in Sonderfällen durch die Applikation direkt realisiert werden sollten. Hier werden sie also nur aus Gründen der Vollständigkeit beschrieben.

1. Es sind eine Reihe von Pointern korrekt einzustellen, um die Bestandteile der Achsen zu verbinden. Diese Aufgabe sollte mit einem Baustein vom Typ [MC\\_AxUtiStandardInit\\_BkPlcMc](#) [▶ 238] gelöst werden, der eine Verschiebung oder Größenänderung im Speicher oder die Änderung eines Typcodes bei einem späteren Online Change erkennt und dann für eine Reinitialisierung der Pointer und ein Nachladen von Parametern sorgt.
2. Die Parameter der Achsen müssen eingestellt werden. Eine hart codierte Zuweisung der Werte durch die Applikation ist zwar technisch machbar, aber in der Regel nicht sinnvoll. Besser ist die Speicherung der Einstellungen in Dateien, die beim Systemstart unter Kontrolle der Applikation durch den [MC\\_AxUtiStandardInit\\_BkPlcMc](#) [▶ 238] Baustein geladen werden. Hinweise zur Inbetriebnahme finden Sie unter Setup.
3. Die Zykluszeit der Task ist in die Parameter der Achse zu übernehmen. Dies sollte am Ende der Parameterladeprozedur erfolgen um diesen für die Funktion vieler Bausteine wichtigen Wert korrekt einzustellen. Ein [MC\\_AxUtiStandardInit\\_BkPlcMc](#) [▶ 238] Baustein übernimmt diese Aufgabe automatisch.

---

**i** Wenn in der Applikation ein Baustein vom Typ [MC\\_AxAdsCommServer\\_BkPlcMc](#) genutzt wird, muss dieser Baustein in derselben Task aufgerufen werden, die auch die Zuweisung der Pointer vornimmt. Ist dies nicht möglich oder nachteilig muss der Aufruf des Bausteins für die Dauer der Zuweisungen gesperrt werden. Andernfalls kann es zum Absturz der PLC-Applikation durch Auslösung von schweren Laufzeitfehlern (Page Fault Exception) kommen.

---

**i** Sämtliche hier aufgeführten Aktivitäten sollten durch einen [MC\\_AxUtiStandardInit\\_BkPlcMc](#) Baustein verwirklicht und koordiniert werden. Wenn die Variable `nInitState` im [Axis\\_Ref\\_BkPlcMc](#) der Achse den Wert 2 oder -2 angenommen hat ist die Initialisierung erfolgreich bzw. mit Fehler beendet. Bei einer erfolgreichen Initialisierung sind [MC\\_AxUtiStandardInit\\_BkPlcMc.Ready](#) und [bParamsEnable](#) in [Axis\\_Ref\\_BkPlcMc](#) auf TRUE, andernfalls bleibt diese Variable FALSE.

---

**i** Die zur Verfügung gestellten Beispielprogramme legen den Namen der Achse und den Namen (inklusive Pfad) der zugehörigen Parameterdatei fest. Diese Festlegungen sind unbedingt an die jeweilige Applikation anzupassen.

---

**FAQ #4: Wie wird die Istposition der Achsen ermittelt?**

Als Positionssensor kommt eine Reihe von Signalgebern in Frage, die nach diversen physikalischen Prinzipien eine positionsabhängige elektrische Größe erzeugen. Diese Größe bestimmt den Typ der einzusetzenden E/A-Komponente. Die pro Achse anzulegenden Variablen der Typen [ST\\_TcPlcDeviceInput](#) [[▶ 138](#)] und [ST\\_TcPlcDeviceOutput](#) [[▶ 140](#)] enthalten Elemente, die mit den Istwert-, Zähl-, Latch-, Control- und Status-Variablen der E/A-Hardware zu verknüpfen sind.

Hier einige Beispiele:

E/A-Komponente	Signal	Encoder Type
AX2000 B110 mit Absolut-Encoder	EtherCAT	<a href="#">iTcMc_EncoderAx2000_B110A</a> [ <a href="#">▶ 184</a> ]
AX2000 B110 mit Resolver	EtherCAT	<a href="#">iTcMc_EncoderAx2000_B110R</a> [ <a href="#">▶ 173</a> ]
AX2000 B200 mit Resolver	EtherCAT	<a href="#">iTcMc_EncoderAx2000_B200R</a> [ <a href="#">▶ 173</a> ]
AX2000 B750 mit Absolut-Encoder	EtherCAT	<a href="#">iTcMc_EncoderAx2000_B750A</a> [ <a href="#">▶ 186</a> ]
AX2000 B900 mit Resolver	EtherCAT	<a href="#">iTcMc_EncoderAx2000_B900R</a> [ <a href="#">▶ 173</a> ]
AX5000 B110 mit Multiturn-Absolut-Encoder	EtherCAT	<a href="#">iTcMc_EncoderAX5000_B110A</a> [ <a href="#">▶ 187</a> ]
EtherCAT-Servoregler mit CoE DS402 Support und Multiturn-Encoder	EtherCAT	<a href="#">iTcMc_EncoderCoE_DS402A</a> [ <a href="#">▶ 188</a> ]
EtherCAT-Servoregler mit CoE DS402 Support und Resolver oder Singleturn-Encoder	EtherCAT	<a href="#">iTcMc_EncoderCoE_DS402SR</a> [ <a href="#">▶ 189</a> ]
EL3102	-10V .. 10V	<a href="#">iTcMc_EncoderEL3102</a> [ <a href="#">▶ 192</a> ]
EL3142	0mA .. 20mA	<a href="#">iTcMc_EncoderEL3142</a> [ <a href="#">▶ 192</a> ]
EL3162	0 .. 10V	<a href="#">iTcMc_EncoderEL3162</a> [ <a href="#">▶ 192</a> ]
EL3255	Potentiometrischer Weggeber	<a href="#">iTcMc_EncoderEL3162</a> [ <a href="#">▶ 192</a> ]
EL5001	SSI	<a href="#">iTcMc_EncoderEL5001</a> [ <a href="#">▶ 192</a> ]
EL5101	A/B-Inkremente, RS422="TTL"	<a href="#">iTcMc_EncoderEL5101</a> [ <a href="#">▶ 194</a> ]
EL7041	A/B-Inkremente, RS422="TTL"	<a href="#">iTcMc_EncoderEL7041</a> [ <a href="#">▶ 194</a> ]
EtherCAT-Encoder mit CoE_DS406 Profil	EtherCAT	<a href="#">iTcMc_EncoderCoE_DS406</a> [ <a href="#">▶ 190</a> ]
IE5009	SSI	<a href="#">iTcMc_EncoderIx5009</a> [ <a href="#">▶ 195</a> ]
IP5009	SSI	<a href="#">iTcMc_EncoderIx5009</a> [ <a href="#">▶ 195</a> ]
KL10xx	2 Bit, A/B-Inkremente	<a href="#">iTcMc_EncoderDigIncrement</a> [ <a href="#">▶ 191</a> ]
KL11xx	2 Bit, A/B-Inkremente	<a href="#">iTcMc_EncoderDigIncrement</a> [ <a href="#">▶ 191</a> ]
KL12xx	2 Bit, A/B-Inkremente	<a href="#">iTcMc_EncoderDigIncrement</a> [ <a href="#">▶ 191</a> ]
KL13xx	2 Bit, A/B-Inkremente	<a href="#">iTcMc_EncoderDigIncrement</a> [ <a href="#">▶ 191</a> ]
KL14xx	2 Bit, A/B-Inkremente	<a href="#">iTcMc_EncoderDigIncrement</a> [ <a href="#">▶ 191</a> ]

E/A-Komponente	Signal	Encoder Type
KL17xx	2 Bit, A/B-Inkmente	<a href="#">iTcMc_EncoderDigIncrement</a> [ <a href="#">► 191</a> ]
KL10xx	4 Bit, Positionsnocken	<a href="#">iTcMc_EncoderDigCam</a> [ <a href="#">► 191</a> ]
KL11xx	4 Bit, Positionsnocken	<a href="#">iTcMc_EncoderDigCam</a> [ <a href="#">► 191</a> ]
KL12xx	4 Bit, Positionsnocken	<a href="#">iTcMc_EncoderDigCam</a> [ <a href="#">► 191</a> ]
KL13xx	4 Bit, Positionsnocken	<a href="#">iTcMc_EncoderDigCam</a> [ <a href="#">► 191</a> ]
KL14xx	4 Bit, Positionsnocken	<a href="#">iTcMc_EncoderDigCam</a> [ <a href="#">► 191</a> ]
KL17xx	4 Bit, Positionsnocken	<a href="#">iTcMc_EncoderDigCam</a> [ <a href="#">► 191</a> ]
KL2521	Pulse Train	<a href="#">iTcMc_EncoderKL2521</a> [ <a href="#">► 195</a> ]
KL2531	Schrittmotor, direkt (Encoder durch Pulszähler emuliert)	<a href="#">iTcMc_EncoderKL2531</a> [ <a href="#">► 195</a> ]
KL2541	Schrittmotor, direkt (mit Encoder oder Encoder durch Pulszähler emuliert)	<a href="#">iTcMc_EncoderKL2541</a> [ <a href="#">► 196</a> ]
KL2542	DC Motor, direkt mit Encoder	<a href="#">iTcMc_EncoderKL2542</a> [ <a href="#">► 196</a> ]
KL3001	-10V .. 10V	<a href="#">iTcMc_EncoderKL3002</a> [ <a href="#">► 197</a> ]
KL3002	-10V .. 10V	<a href="#">iTcMc_EncoderKL3002</a> [ <a href="#">► 197</a> ]
KL3011	0mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3012	0mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3021	4mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3022	4mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3041	0mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3042	0mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3044	0mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3051	4mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3052	4mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3054	4mA .. 20mA	<a href="#">iTcMc_EncoderKL3042</a> [ <a href="#">► 197</a> ]
KL3061	0V .. 10V	<a href="#">iTcMc_EncoderKL3062</a> [ <a href="#">► 197</a> ]
KL3062	0V .. 10V	<a href="#">iTcMc_EncoderKL3062</a> [ <a href="#">► 197</a> ]
KL3064	0V .. 10V	<a href="#">iTcMc_EncoderKL3062</a> [ <a href="#">► 197</a> ]
KL3162	0V .. 10V	<a href="#">iTcMc_EncoderKL3162</a> [ <a href="#">► 197</a> ]
KL5001	SSI	<a href="#">iTcMc_EncoderKL5001</a> [ <a href="#">► 198</a> ]
KL5101	A/B-Inkmente, RS422="TTL"	<a href="#">iTcMc_EncoderKL5101</a> [ <a href="#">► 198</a> ]
KL5111	A/B-Inkmente, RS422="HTL"	<a href="#">iTcMc_EncoderKL5111</a> [ <a href="#">► 198</a> ]
M2510	-10V .. 10V	<a href="#">iTcMc_EncoderM2510</a> [ <a href="#">► 199</a> ]
M3100	A/B-Inkmente, RS422="TTL"	<a href="#">iTcMc_EncoderM3120</a> [ <a href="#">► 199</a> ]
M3120	A/B-Inkmente, RS422="TTL"	<a href="#">iTcMc_EncoderM3120</a> [ <a href="#">► 199</a> ]

Wenn eine der hier genannten Komponenten eingesetzt wird kann in der Regel einer der bereitgestellten Encoder-Bausteine verwendet werden. Die Schnittstellen dieser Bausteine sind nicht garantiert und sollten somit nicht direkt von der Applikation aufgerufen werden. Besser ist es, den Encoder-Typ entsprechend der Konstanten in [E\\_TcMcEncoderType](#) [[► 93](#)] unter `nEnc_Type` in [ST\\_TcHydAxParam](#) [[► 120](#)] einzustellen und einen Baustein vom Typ [MC\\_AxRtEncoder\\_BkPlcMc](#) [[► 183](#)] zu verwenden. Dieser ruft dann den typrichtigen Unterbaustein automatisch auf.

Alle Encoder-Bausteine verwenden die Parameter `fEnc_IncWeighting` und `fEnc_IncInterpolation` als Inkrementbewertung. Bei absoluten Wegsensoren wird zusätzlich `fEnc_ZeroShift` als Nullpunktverschiebung verwendet. Inkrementelle Sensoren erfordern in der Regel eine Referenzfahrt mit einem [MC\\_Home\\_BkPlcMc](#)

[▶ 68] Baustein, in deren Verlauf `fEnc_RefShift` in `ST_TcHydAxRtData` [▶ 131] ermittelt wird. Dieser Wert übernimmt dann die Funktion der Nullpunktverschiebung. Selbstverständlich kann die Nullpunktverschiebung in Sonderfällen auch mit einem `MC_SetPosition_BkPlcMc` [▶ 43] Baustein festgelegt werden. Der Referenziert-Status der Achse ist mit `MC_SetReferenceFlag_BkPlcMc` [▶ 44]() zu definieren.

Ist es aus technischen Gründen nicht möglich, die Istposition mit Bausteinen der Library zu ermitteln kann diese Aufgabe auch durch Applikationsbausteine ausgeführt und das Ergebnis in `fActPos` und bei Bedarf auch `fActVelo` in `ST_TcHydAxRtData` [▶ 131] eingetragen werden. Um die Durchgängigkeit zu erhalten sollte auch hier wenn möglich auf die Parameter `fEnc_IncWeighting`, `fEnc_IncInterpolation` und `fEnc_ZeroShift` bzw. `fEnc_RefShift` zurückgegriffen werden.



Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollte zur Vereinfachung ein Baustein des Typs `MC_AxStandardBody_BkPlcMc` verwendet werden.



Die Inbetriebnahme einer Istdruckermittlung mit Bausteinen vom Typ `MC_AxRtReadPressureSingle_BkPlcMc` oder `MC_AxRtReadPressureDiff_BkPlcMc` ist in der Dokumentation des Bausteins beschrieben.

---

#### FAQ #5: Wie wird der Stellwert für eine Achse erzeugt?

Durch die PLC-Applikation muss in jedem Zyklus für jede Achse ein Baustein vom Typ `MC_AxRuntime_BkPlcMc` [▶ 221] oder ersatzweise ein geeigneter Reglerbaustein (z. B. ein Druckregler) aufgerufen werden. Der Parameter `nProfileType` in `ST_TcHydAxParam` [▶ 120] legt fest, nach welchem Verfahren der Stellwert generiert wird. Dabei werden je nach Typ und in Abhängigkeit von weiteren Parametern der Achse und den Daten der Bewegung angepasste Geschwindigkeitsstellwerte errechnet. Diese Stellwerte sind jedoch auf den abstrakten Zahlenbereich  $\pm 1.0$  normiert und noch nicht für die direkte Ausgabe auf einer E/A-Hardware vorbereitet.



Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollen, verwenden Sie zur Vereinfachung ein Baustein des Typs `MC_AxStandardBody_BkPlcMc`. [▶ 237]

---

#### FAQ #6: Wie wird der Stellwert einer Achse für die Ausgabe vorbereitet?

Nach dem Aufruf des `MC_AxRuntime_BkPlcMc` [▶ 221] Bausteins ist für jede Achse ein Baustein vom Typ `MC_AxRtFinish_BkPlcMc` [▶ 230] aufzurufen. Dieser Baustein fasst mehrere Geschwindigkeitskomponenten (Stellwert, Reglerausgabe, Offsetkompensation, Überdeckungskompensation) zusammen und berücksichtigt dabei eventuelle Knickpunkte der Vorsteuercharakteristik.

Für die Ausgabe auf einer E/A-Baugruppe ist in der Regel eine numerische Anpassung erforderlich. Zu diesem Zweck ist für jede Achse ein `MC_AxRtDrive_BkPlcMc` [▶ 172] Baustein aufzurufen. Der Wert in `nDrive_Type` in `ST_TcHydAxParam` [▶ 120] wählt den zu benutzenden hardwarespezifischen Unterbaustein aus.

Die pro Achse anzulegenden Variablen der Typen `ST_TcPlcDeviceInput` [▶ 138] und `ST_TcPlcDeviceOutput` [▶ 140] enthalten Elemente, die mit den Sollwert- und Steuer-Variablen der E/A-Hardware zu verknüpfen sind.



Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollen, verwenden Sie zur Vereinfachung ein Baustein des Typs `MC_AxStandardBody_BkPlcMc`. [▶ 237]

---

**FAQ #7: Wie wird der Stellwert an die Achse ausgegeben?**

Als Steller kommen eine Reihe von Geräten und Einrichtungen in Frage, die abhängig von einer elektrischen Größe nach diversen physikalischen Prinzipien eine variable Geschwindigkeit dosieren. Diese Größe bestimmt den Typ der einzusetzenden E/A-Komponente. Die pro Achse anzulegenden Variablen der Typen [ST\\_TcPlcDeviceInput \[► 138\]](#) und [ST\\_TcPlcDeviceOutput \[► 140\]](#) enthalten Elemente, die mit den Variablen der E/A-Hardware zu verknüpfen sind.

Hier einige Beispiele:

<b>E/A-Komponente</b>	<b>Signal</b>	<b>Drive Type</b>
AX2000 B110 mit Absolut-Encoder	EtherCAT	<a href="#">iTcMc_DriveAX2000_B110A [► 173]</a>
AX2000 B110 mit Resolver	EtherCAT	<a href="#">iTcMc_DriveAX2000_B110R [► 173]</a>
AX2000 B200 mit Resolver	EtherCAT	<a href="#">iTcMc_DriveAX2000_B200R [► 173]</a>
AX2000 B750 mit Absolut-Encoder	EtherCAT	<a href="#">iTcMc_DriveAx2000_B750A [► 173]</a>
AX2000 B900 mit Resolver	EtherCAT	<a href="#">iTcMc_DriveAX2000_B900R [► 173]</a>
AX5000 B110 mit Absolut-Encoder	EtherCAT	<a href="#">iTcMc_DriveAX5000_B110A [► 174]</a>
EtherCAT-Servoregler mit CoE DS402 Support und Resolver, Singleturn- oder Multiturn-Encoder	EtherCAT	<a href="#">iTcMc_DriveCoE_DS402 [► 174]</a>
EtherCAT-Ventil mit CoE_DS408 Profil	EtherCAT	<a href="#">iTcMc_Drive_CoE_DS408 [► 174]</a>
EL2535	PWM	<a href="#">iTcMc_DriveEL2535</a>
EL4031, EL4032, EL4034, EL4038 EL4131, EL4132, EL4134	-10V .. 10V	<a href="#">iTcMc_DriveEL4132 [► 175]</a>
EL4011, EL4012, EL4014, EL4018, EL4112 EL4021, EL4022, EL4024, EL4028, EL4122, EL4124	0..20mA  4..20mA	<a href="#">iTcMc_DriveEL4x22</a>
EL7031	Schrittmotor, direkt	<a href="#">iTcMc_DriveEL7031 [► 177]</a>
EL7041	Schrittmotor, direkt	<a href="#">iTcMc_DriveEL7041 [► 177]</a>
IE2512	PWM	<a href="#">iTcMc_DriveIex2512_1Coil [► 175]</a> <a href="#">iTcMc_DriveIex2512_2Coil [► 175]</a>
IP2512	PWM	<a href="#">iTcMc_DriveIex2512_1Coil [► 175]</a> <a href="#">iTcMc_DriveIex2512_2Coil [► 175]</a>
KL20xx, KL21xx, KL22xx, KL24xx	5 Bit zum Betrieb eines Frequenzumrichters mit Festfrequenzen	<a href="#">iTcMc_DriveLowCostInverter [► 181]</a>
KL20xx, KL21xx, KL22xx, KL24xx	4 Bit zum Betrieb eines spannungsgesteuerten Schrittmotors	<a href="#">iTcMc_DriveLowCostStepper [► 181]</a>
KL2521	Pulse Train	<a href="#">iTcMc_DriveKL2521 [► 178]</a>
KL2531	Schrittmotor, direkt	<a href="#">iTcMc_DriveKL2531 [► 179]</a>
KL2532	DC Motor, direkt mit Encoder	<a href="#">iTcMc_DriveKL2532 [► 179]</a>
KL2535	PWM	<a href="#">iTcMc_DriveKL2535_1Coil [► 180]</a> <a href="#">iTcMc_DriveKL2535_2Coil [► 180]</a>
KL2541	Schrittmotor, direkt	<a href="#">iTcMc_DriveKL2541 [► 180]</a>
KL2542	DC Motor, direkt mit Encoder	<a href="#">iTcMc_DriveKL2542 [► 180]</a>
KL4031	-10V .. 10V	<a href="#">iTcMc_DriveKL4032 [► 181]</a>
KL4032	-10V .. 10V	<a href="#">iTcMc_DriveKL4032 [► 181]</a>
KL4034	-10V .. 10V	<a href="#">iTcMc_DriveKL4032 [► 181]</a>

E/A-Komponente	Signal	Drive Type
M2400	-10V .. 10V	iTcMc_DriveM2400_D1 [▶ 182], iTcMc_DriveM2400_D2, iTcMc_DriveM2400_D3, iTcMc_DriveM2400_D4

Wenn eine der hier genannten Komponenten eingesetzt wird kann in der Regel einer der bereitgestellten Drive-Bausteine verwendet werden. Diese Schnittstellen dieser Bausteine sind nicht garantiert und sollten somit nicht direkt von der Applikation aufgerufen werden. Besser ist es, den Drive-Typ entsprechend der Konstanten in [E\\_TcMcDriveType \[▶ 90\]](#) unter `nDrive_Type` in [ST\\_TcHydAxParam \[▶ 120\]](#) einzustellen und einen Baustein vom Typ [MC\\_AxRtDrive\\_BkPlcMc \[▶ 172\]](#) zu verwenden.



Wenn nur die üblichen Bausteine (Encoder, Generator, Finish, Drive) für die Achse aufgerufen werden sollen, verwenden Sie zur Vereinfachung ein Baustein des Typs [MC\\_AxStandardBody\\_BkPlcMc \[▶ 237\]](#)

### FAQ #8: In welcher Reihenfolge sind die Bausteine einer Achse aufzurufen?

1. Obligatorisch: Alle Bausteine, die den Istzustand der Achse feststellen. Dazu gehören Bausteine der Typen [MC\\_AxRtEncoder\\_BkPlcMc \[▶ 183\]](#), [MC\\_AxRtReadPressureDiff\\_BkPlcMc \[▶ 204\]](#) oder [MC\\_AxRtReadPressureSingle\\_BkPlcMc \[▶ 207\]](#).
2. Üblich: Bausteine oder Befehle, die für eine Aktualisierung der Freigabe-Signale der Achse sorgen. Dies ist in der Regel ein Baustein vom Typ [MC\\_Power\\_BkPlcMc \[▶ 27\]](#). Bei Achsen mit einem Inkremental-Encoder, der unter Verwendung eines Nocken referenziert wird, kommt ein Funktionsaufruf [MC\\_AxRtSetReferencingCamSignal\\_BkPlcMc](#) dazu.
3. Optional: Bausteine, die aus einem Istzustand der Achse, einem E/A-Signal oder einem Signal der Applikation eine Entscheidung ableiten und einen Befehl auslösen. Zum Beispiel kann als Reaktion auf das Signal eines Näherungsschalters ein Achsstart ausgelöst werden oder in Abhängigkeit vom Druckanstieg eine Achsbewegung vor dem Erreichen der Zielposition gestoppt werden.
4. Obligatorisch: Stellwertgeneratoren wie Bausteine des Typs [MC\\_AxRuntime\\_BkPlcMc \[▶ 221\]](#).
5. Optional: Bei Bedarf können an dieser Stelle verschiedene Regler aufgerufen werden. Dabei kann es sich um einen Baustein der Typen [MC\\_AxCtrlSlowDownOnPressure\\_BkPlcMc \[▶ 162\]](#) oder ähnliches handeln.
6. Obligatorisch: Ein Anpassungsbaustein des Typs [MC\\_AxRtFinish\\_BkPlcMc \[▶ 230\]](#).
7. Optional: Bei Bedarf kann an dieser Stelle ein Baustein für die automatische Inbetriebnahme aufgerufen werden.
8. Obligatorisch: Ein Ausgabebaustein des Typs [MC\\_AxRtDrive\\_BkPlcMc \[▶ 172\]](#).

Anstelle der Bausteine der Library können auch Applikations-Bausteine treten. Dabei ist die Notwendigkeit jedoch sorgfältig zu überprüfen und eine Kompatibilität zur Library sicher zu stellen. In einigen Anwendungsfällen kann dies notwendig werden, um einen nicht standardisierten Sensor oder Aktor anzupassen oder um ein besonderes Regelungsproblem zu lösen.

### FAQ #9: Wie steuere ich eine Ventilstufe (Onboard oder extern) an?

In der [ST\\_TcPlcDeviceOutput \[▶ 140\]](#) Struktur sind die Signale **bPowerOn** und **bEnable** für die Steuerung der Endstufenversorgung und -freigabe vorgesehen. Beide Signale werden durch Bausteine vom Typ [MC\\_Power\\_BkPlcMc \[▶ 27\]](#) gesetzt, wenn der Eingang **Enable** gesetzt wird. Gleichzeitig setzt dieser Baustein die Software-Reglerfreigabe in [ST\\_TcHydAxRtData \[▶ 131\]](#). [nDeCtrlDWord \[▶ 320\]](#).

In der [ST\\_TcPlcDeviceInput \[▶ 138\]](#) Struktur sind die Signale **bPowerOk**, **bEnAck** und **bReady** für die Kontrolle der Endstufenversorgung, die Rückmeldung der Endstufenfreigabe und des Statussignals vorgesehen. Die von verschiedenen Herstellern bereitgestellten Signale unterscheiden sich jedoch sehr stark. Derzeit wird deshalb nur das **bPowerOk** Signal bei der Festlegung des **Status** Ausgangs des [MC\\_Power\\_BkPlcMc \[▶ 27\]](#) Bausteins verwendet. Steht auch hierfür kein geeignetes Signal zur Verfügung oder soll keine Überwachung realisiert werden ist [ST\\_TcHydAxParam \[▶ 120\]](#). `bDrive_DefaultPowerOk` zu setzen.

## FAQ #10: Wie lege ich einen Meldungspuffer an?

Eine direkte Ausgabe von Meldungen aus den Bausteinen würde zu nicht kalkulierbaren Laufzeitschwankungen führen. Aus diesem Grund werden die Meldungen in einem Puffer abgelegt und, falls dies gewünscht wird, nacheinander in die Ereignisanzeige von Windows ausgegeben.

Um einen Meldungspuffer nutzen zu können ist eine Variable des Typs [ST\\_TcPlcMcLogBuffer \[▶ 142\]](#) anzulegen. In diesen Puffer werden die Meldungen **aller** Achsen aufgenommen. Es ist wichtig, unabhängig von der Anzahl der Achsen, nur eine solche Variable im Projekt anzulegen. Die Adresse dieses Puffers ist den Bausteinen [MC\\_AxUtiStandardInit\\_BkPlcMc \[▶ 238\]](#) aller Achsen zusammen mit den Adressen der anderen individuellen Achsbestandteile zu übergeben. Diese Bausteine werden in der Regel im Initialisierungsteil des Projekts aufgerufen. In der Struktur [Axis\\_Ref\\_BkPlcMc \[▶ 84\]](#) wird diese Adresse im Element `pStAxLogBuffer` abgelegt und durch den Baustein überwacht.

In `nLogLevel` in [Axis\\_Ref\\_BkPlcMc \[▶ 84\]](#) wird festgelegt, bis zu welcher Bedeutsamkeit anfallende Meldungen in den Puffer aufgenommen werden. Die dabei zu verwendenden [Werte \[▶ 328\]](#) sind in den Globalen Variablen der Bibliothek definiert. Es ist zu beachten, dass diese Einstellung pro Achse zu erfolgen hat.

Die Bausteine der Bibliothek erkennen die oben genannten Vorbereitungen und werden die Ausgabe von Meldungen aufnehmen. Allerdings würde der Puffer bei aktivierter Meldungsabgabe kurzer Zeit vollständig gefüllt sein und keine weiteren Meldungen aufnehmen. Um dies zu vermeiden sind zwei Wege nutzbar.

### FAQ #10.1: Weitergabe der Meldungen an die Ereignisanzeige von Windows

Um die angefallenen Meldungen aus dem LogBuffer der Library in die Ereignisanzeige von Windows zu transportieren ist zyklisch ein Baustein des Typs [MC\\_AxRtLoggerSpool\\_BkPlcMc \[▶ 244\]](#) aufzurufen. Bei jedem Aufruf wird so eine Meldung aus dem LogBuffer entfernt.



Auch auf Rechnern unter Windows CE kann eine Ereignisanzeige für die von TwinCAT erzeugten Meldungen ergänzt werden. Dazu wird dieser Dienst vom TwinCAT System Service nachgebildet. Allerdings wird in der Regel hier nur eine FlashDisk zur Verfügung stehen. Um den hier relativ kleinen Meldungsvorrat der Ereignisanzeige nicht zu überlasten sollten nur Fehler aufgezeichnet werden.

---

### FAQ #10.2: Löschen der jeweils ältesten Meldungen

Um einen Minimalwert an aufnehmbaren Meldungen sicher zu stellen kann zyklisch ein Baustein des Typs [MC\\_AxRtLoggerDespool\\_BkPlcMc \[▶ 242\]](#) aufgerufen werden. Dieser Baustein entfernt pro Aufruf die jeweils älteste Meldung im LogBuffer, bis eine übergebene Anzahl von freien Meldungen zur Verfügung steht. Die entfernten Meldungen gehen dabei verloren.

### FAQ #10.3: Erzeugen von Loggereinträgen durch die Applikation

Eine Meldung kann durch die Applikation entweder achsbezogen oder nicht achsbezogen ausgegeben werden. Hierzu stehen die Bausteine [MC\\_AxRtLogAxisEntry\\_BkPlcMc \[▶ 240\]](#) und [MC\\_AxRtLogEntry\\_BkPlcMc \[▶ 242\]](#) zur Verfügung.

### FAQ #10.4: Bibliotheksinterner Meldungspuffer

Bei einem zyklischen Aufruf des Bausteins [MC\\_Communications\\_BkPlcMc \[▶ 264\]](#) wird ein bibliotheksinterner Meldungspuffer genutzt. Dazu referenziert der Baustein `MC_Communications_BkPlcMc` den internen Meldungspuffer in den übergebenen Achsen und ruft den Baustein [MC\\_AxRtLoggerSpool\\_BkPlcMc \[▶ 244\]](#) auf. Ein in der PLC-Applikation angelegter Meldungspuffer ist hierbei nicht mehr erforderlich. Außerdem wird der beim Aufruf von [MC\\_AxUtiStandardInit\\_BkPlcMc \[▶ 238\]](#) referenzierte Meldungspuffer ignoriert, wenn der Baustein `MC_Communications_BkPlcMc` aufgerufen wird.

## FAQ #11: Wie breche ich die Überwachung einer Funktion ab?

Einige Bausteine der Bibliothek starten eine Aktivität, für deren Abarbeitung ihr zyklischer Aufruf jedoch nicht mehr zwingend erforderlich ist. Allerdings sind auch diese Bausteine entsprechend den Regeln der PLCopen Motion Control Richtlinien so ausgeführt, dass sie die Aktivität vollständig überwachen und an ihren Ausgängen darstellen. Erkennbar ist dies durch den Ausgang `Busy`, den die meisten Bausteine bereitstellen.

Den zyklischen Aufruf eines in diesem Überwachungszustand befindlichen Bausteins zu unterlassen wird zumeist erhebliche Probleme erzeugen. Der nächste Funktionsstart mit dem betreffenden Baustein wird Probleme bei der Auswertung der Flanken an seinen Eingängen haben oder er wird feststellen, dass die Achse inzwischen eine andere Funktion ausgeführt hat und ein nicht vorhandenes Problem (CommandAborted) signalisieren.

In älteren Versionen der Library wurde ein Baustein des Typs `MC_AxUtiCancelMonitoring_BkPlcMc()` bereitgestellt, der für einige wenige Motion Funktionen die Überwachung durch den die Funktion einleitenden Baustein abgebrochen hat. Dieser Baustein wird durch die inzwischen vollständigere Implementation der PLC Open Regeln nicht mehr benötigt.

Soll ein Baustein die Ausführung seiner Funktion nicht mehr überwachen ist es in den meisten Fällen ausreichend ihn ein- oder mehrmals mit **Execute:=FALSE** aufzurufen. Dies gilt insbesondere für `MC_MoveAbsolute BkPlcMc [▶ 72]()`, `MC_MoveRelative BkPlcMc [▶ 76]()` und `MC_MoveVelocity BkPlcMc [▶ 78]()`.

Anschließend kann in gleichen oder einem späteren Zyklus mit demselben Baustein oder einer Instanz desselben oder eines anderen Typs eine neue Funktionalität gestartet werden. Diese Vorgehensweise ist beliebig wiederholbar.

---

**i** Komplexe aus mehreren Teilaktionen zusammengesetzte Funktionen wie `MC_Home_BkPlcMc()` erfordern den kontinuierlichen Aufruf des Bausteins da dieser die erforderlichen Abläufe selbst organisiert. (`MC_Home_BkPlcMc()`) [▶ 68]

---

#### FAQ #12: Wie überwache ich die Kommunikation mit einem E/A-Gerät?

Sowohl `ST_TcPlcDeviceInput [▶ 138]` als auch `ST_TcPlcDeviceOutput [▶ 140]` Variablen stellen ein Element mit Namen **uiBoxState** bereit. Wenn die eingesetzten Buskoppler bzw. die Interface-Karten der Leistungsteile eine entsprechende Variable anbieten und beim eingesetzten Feldbus die Variable bei ungestörter Kommunikation den Wert 0 annimmt sollte eine Verknüpfung hergestellt werden. Diese Möglichkeit besteht z.B. beim **Beckhoff Lightbus** und bei **Realtime Ethernet**. Wird für die Achse ein `MC_Power_BkPlcMc [▶ 27]` Baustein verwendet überwacht dieser den **uiBoxState** und meldet Störungen der Kommunikation. In einem solchen Fall wird die Achse in einen Stöorzustand versetzt.

**EtherCAT** bietet hier erheblich erweiterte Möglichkeiten.

#### FAQ #13: Wie versee ich die kundenspezifischen Achsparameter mit selbstgewählten Bezeichnungen?

Die `Axis_Ref_BkPlcMc [▶ 84]` Struktur unterstützt mit dem `pAuxLabels` Pointer die Verwendung eines Array von Texten, die vom `PlcMcManager` mit angezeigt werden. Diese Texte können durch den `MC_AxUtiStandardInit_BkPlcMc [▶ 238]` Baustein beim Start der Applikation aus einer Datei geladen werden. Dazu muss diesem Baustein die Adresse einer `ST_TcMcAuxDataLabels [▶ 137]` Variablen übergeben und eine geeignete Datei bereitgestellt werden.

Selbstverständlich ist es auch möglich, die Elemente der `ST_TcMcAuxDataLabels [▶ 137]` Variablen durch direktes Zuweisen aus der Applikation zu definieren. In diesem Fall wird die Datei nicht benötigt.

---

**i** Eine Reihe von Reglerbausteinen der Library definiert die Texte des Arrays automatisch.

---

#### FAQ #14: Wie steuere ich ein Stromventil an?

Im Gegensatz zu einem 4/2 oder 3/2 Proportionalwegeventil oder Servoventil wird ein Stromventil mit einem 0..10V Signal (bei vorhandener Ventilendstufe) angesteuert oder mit einem eingepprägten Strom von  $0 \dots I_{Nenn}$  betätigt. In dieser Ansteuerung wird nur der Betrag der Geschwindigkeit übertragen. Die Richtung wird nicht mit dem Vorzeichen, sondern auf andere Weise transportiert. Hierfür werden zumeist digitale Signale benötigt, die für die Ansteuerung von Schaltventilen verwendet werden. In der `ST_TcPlcDeviceOutput [▶ 140]` Struktur stehen hierfür unter anderem die Elemente **bBrakeOff**, **bMovePos** und **bMoveNeg** bereit. Für die Erzeugung eines Absolut-Stellwerts ist `bDrive_AbsoluteOutput` in den Achsparametern zu setzen.





Auf diesem Weg ist auch der Einsatz von klassischen Frequenzumrichtern mit Asynchronmotor, Encoder und Bremse möglich, wenn der Umrichter einen Analogeingang bereitstellt.

#### FAQ #15: Welche Variablen einer Achse sollten mit dem Scope aufgezeichnet werden?

Die folgende Signalzusammenstellung ist zu empfehlen:

- Immer: **Achsisposition**: Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData.fActPos: In Istwerteinheiten wie sie durch die Encoder-Skalierung festgelegt werden.
- Vor allem bei zeitgeführter Sollwertgenerierung, Getriebe- oder Gleichlaufkopplung, Kurvenscheibe: **Achssollposition**: Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData.fSetPos: In Istwerteinheiten wie sie durch die Encoder-Skalierung festgelegt werden.
- Vor allem bei der Inbetriebnahme: **Geschwindigkeitswert**: Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData.fActVelo: Geschwindigkeit in physikalischer Darstellung.
- Vor allem bei der Inbetriebnahme: **Restweg** oder **Zielposition**: Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData.fDistanceToTarget oder Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData.fTargetPos: In Istwerteinheiten wie sie durch die Encoder-Skalierung festgelegt werden.
- Nur bei aktiver Druck/Krafterfassung: **diverse Druck- und Kraftistwerte**: In Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData: Nach Bedarf fActPressure fActPressureA fActPressureB fActForce fValvePressure fSupplyPressure: Drücke und Kräfte, Einheit wird durch die Parametrierung der Erfassungsbausteine festgelegt.
- Vor allem bei der Inbetriebnahme: **Geschwindigkeitsstellwert**: Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData.fSetVelo: Geschwindigkeit in physikalischer Darstellung.
- Vor allem bei der Inbetriebnahme: **Reglerausgabe**: Axis\_Ref\_BkPlcMc.ST\_TcHydAxRtData.fLagCtrlOutput: Geschwindigkeit in physikalischer Darstellung.



Die Signalauswahl im ScopeView wird erleichtert, wenn die Axis\_Ref\_BkPlcMc Variablen einen Namen erhalten, der mit aaa\_ beginnt. Diese Vorgehensweise wird in den Programmbeispielen benutzt und sorgt dafür, dass die Variablen in der Symbol-Liste schnell auffindbar sind.



In der Signalzusammenstellung des ScopeViews können Kanäle zeitweise deaktiviert werden. So ist es möglich, eine umfassende Zusammenstellung vorzuhalten, aber nur die aktuell interessanten Daten aufzuzeichnen.

#### FAQ #16: Welche Bedeutung hat die Variable nDebugTag in Axis\_Ref\_BkPlcMc?



In einigen Versionen ist diese Variable vorhanden. Sie hat für das PLC-Projekt keine Bedeutung.

In diesen Variablen hinterlegen fast alle Bausteine der Bibliothek für die Dauer ihrer Ausführung eine eindeutige Kennung. Dazu wird der vorgefundene Inhalt in einer lokalen Variablen des Bausteins gesichert und unmittelbar vor dem Verlassen des Bausteins wiederhergestellt.

Sollte es zu einem Programm-Absturz kommen und es besteht der Verdacht, dass es zu einem Problem in einem Baustein der Bibliothek gekommen ist, sind die **nDebugTag** Variablen aller Achsen zu überprüfen. Ist ein Wert <> 0 vorhanden, war der Baustein vom Absturz betroffen und die Ursache ist zu klären. Die verwendeten numerischen Werte sind in der Bibliothek unter "Globale\_Konstanten" aufgeführt. Es sollte zusätzlich der Inhalt von ST\_TcHydAxRtData.fTopBlockName festgestellt werden. Hier ist in der Regel der Name des von der Applikation direkt aufgerufenen Bausteins zu finden.

#### FAQ #17: Welche Besonderheiten sind beim Einsatz von Sercos-Antrieben zu berücksichtigen?

Beim Betrieb von Sercos-Antrieben (ab V3.0.26) sind die nachfolgenden Regeln zu beachten:

- Die Sercos-Masteranschaltung (z.B. FC7501 usw.) muss im System Manager die textuelle Bezeichnung "SercosMaster" erhalten. Andernfalls ist weder eine Steuerung der Sercos Phase noch eine Parameter- und Diagnose-Kommunikation möglich.
- Es kann nur ein Sercos Strang mit der Bibliothek verwendet werden.
- Die Antriebsgeräte am Sercos Strang müssen im System Manager den textuellen Namen erhalten, unter dem sie der Bibliothek durch den Aufruf des `MC_AxUtiStandardInit_BkPlcMc()` Bausteins bekannt sind. Andernfalls ist weder eine Steuerung der Sercos Phase noch eine Parameter- und Diagnose-Kommunikation möglich.
- Die Eingangs-Variable `SystemState` [► 173] der Sercos-Masteranschaltung ist für jedes Antriebsgerät des Sercos Strangs zu verknüpfen.
- Beim Reset eines von mehreren Antrieben am Sercos Strang kann dieser den Feldbus unterbrechen. Die Sercos-Masteranschaltung wird dann einen entsprechenden Phasenwechsel durchlaufen. In der Regel wird der Hochlauf bis zur Phase 4 automatisch erfolgen. Anschließend wird:
  - die vom Reset adressierte Achse fehlerfrei sein sofern nicht weiterbestehende Probleme dagegen sprechen.
  - sich jede andere Achse am Sercos Strang im Fehlerzustand (Feldbusausfall, Achse nicht betriebsbereit) befinden. Wenn der auslösende Reset der ersten Achse abgearbeitet ist können die anderen Achsen in der Regel ohne einen Phasenwechsel durch einen Reset in einen fehlerfreien Zustand gebracht werden.

Dieses Verhalten ist durch Eigentümlichkeiten des Sercos Feldbus festgelegt und durch die Bibliothek nicht zu beeinflussen. Es muss durch die Applikation in geeigneter Weise berücksichtigt werden.
- In Abhängigkeit von bestimmten Parametereinstellungen des Antriebsstellers werden Achsparameter automatisch ermittelt oder sind manuell vorzugeben:
  - S-0-0076, Bits 0 bis 2 legen die Wichtungsart der Lagedaten fest. Unterstützt wird:
    - a) 0 0 1 translatorische Gewichtung:  
S-0-0123 definiert die Umdrehungsauflösung (Encoder-Interpolation). Aus dieser Anzahl und der Gewichtung (S-0-0077, S-0-0078) wird der Umdrehungsvorschub errechnet.
    - b) 0 1 0 rotatorische Gewichtung:  
S-0-0079 definiert die Umdrehungsauflösung (Encoder-Interpolation). Der Umdrehungsvorschub ist manuell einzustellen.
  - S-0-0044, Bits 0 bis 2 legen die Wichtungsart der Geschwindigkeitsdaten fest. Unterstützt wird:
    - a) 0 0 1 translatorische Gewichtung:  
Der Geschwindigkeitsstellwert wird mit dem Umdrehungsvorschub und der Umdrehungsauflösung in eine Geschwindigkeiten in Encoder-Inkrementen pro Zeit umgerechnet. Diese Information wird mit der Geschwindigkeitsauflösung (S-0-0045, S-0-0046) verrechnet und ausgegeben.
    - b) 0 1 0 rotatorische Gewichtung  
Der Geschwindigkeitsstellwert wird mit dem Umdrehungsvorschub in eine Drehzahl umgerechnet und ausgegeben.
  - S-0-0091 wird mit der oben beschriebenen Umrechnungsmethode für Geschwindigkeitsstellwerte bewertet und als Bezugsgeschwindigkeit übernommen. Sollte die Maximalgeschwindigkeit den so ermittelten Wert übersteigen wird sie entsprechend begrenzt.

#### FAQ #18: Wie wird ein Druck oder eine Kraft ermittelt?

Zur Ermittlung eines Ist-Drucks oder einer Ist-Kraft sind pro Achse einer oder mehrere Bausteine der Typen `MC_AxRtReadPressureDiff_BkPlcMc` [► 204], `MC_AxRtReadForceDiff_BkPlcMc` [► 199], `MC_AxRtReadForceSingle_BkPlcMc` [► 202] oder `MC_AxRtReadPressureSingle_BkPlcMc` [► 207] aufzurufen. Details zur Aufrufreihenfolge sind unter [FAQ #8](#) [► 310] zu finden.

Die an die Bausteine zu übergebenden AD-Wandlerwerte sind auf allozierte Variablen der Applikation zu verknüpfen. Details zur Wahl und zur Parametrierung sind in den Beschreibungen der Bausteine zu finden.

#### FAQ #19: Welche Besonderheiten sind beim Einsatz von AX5000 Antrieben zu berücksichtigen?

Bei AX5000 Geräten wird beim Herstellen der Kommunikation eine Reihe von IDNs aus dem Gerät gelesen und es werden verschiedene Parameter automatisch berechnet.

IDN	verwendet für Parameter
44	Bezugsgeschwindigkeit, intern: Skalierung der Geschwindigkeitsausgabe
45	intern: Skalierung der Geschwindigkeitsausgabe
46	intern: Skalierung der Geschwindigkeitsausgabe
76	Encoderinterpolation
79	Encoderinterpolation
91	Bezugsgeschwindigkeit

Die folgenden Parameter werden somit automatisch eingestellt und sind über den PlcMcManager nicht beeinflussbar:

Parameter		beeinflusst welche weiteren Parameter
Global: Bezugsgeschw.	wird aus der Maximal-Drehzahl des Geräts und dem Umdrehungsvorschub berechnet	Handgeschwindigkeiten, Max. Appl. Geschw.
Encoder: Inc.Zwischenteilung	wird aus IDN79 des Geräts gelesen	<b>Achtung:</b> der Umdrehungsvorschub ist als Inc.Bewertung einzugeben

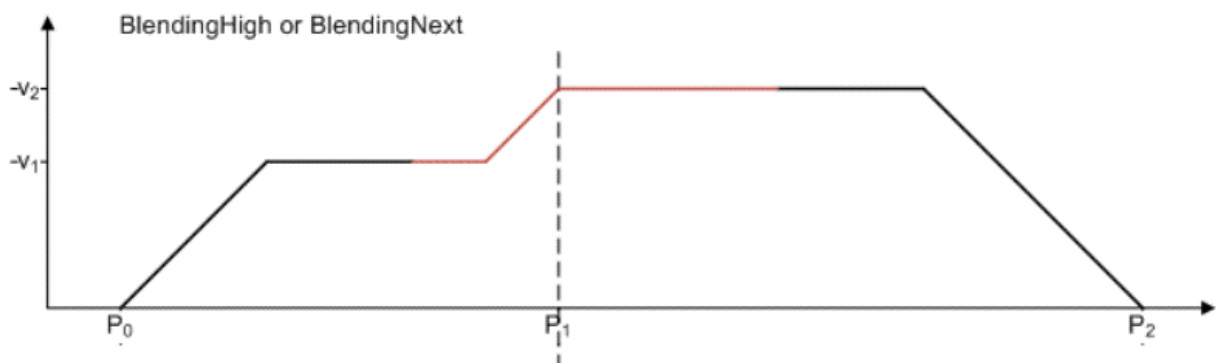
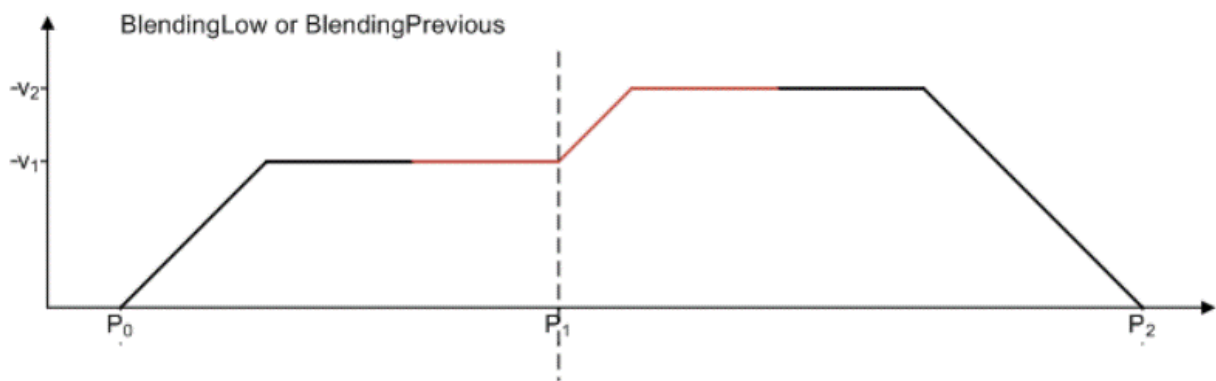
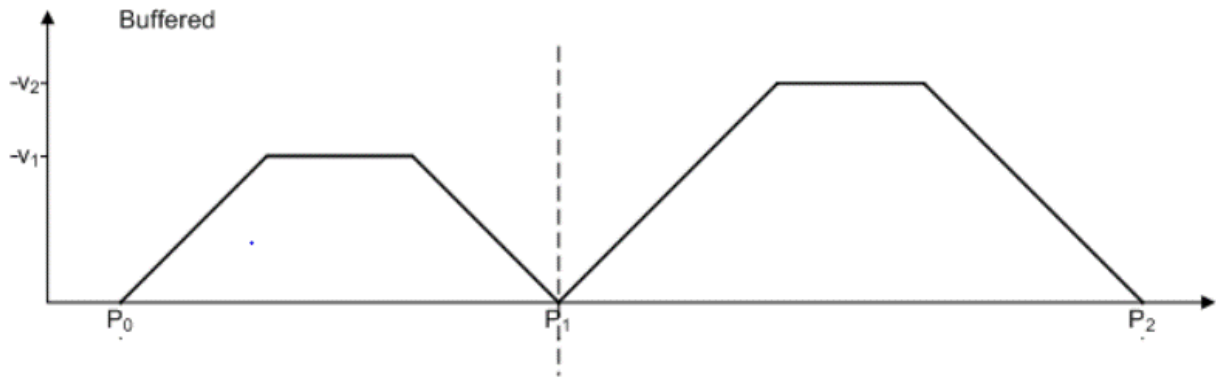
**FAQ #20: Wie wird eine Achse für die Nutzung des Blendings nach PLC Open vorbereitet?**

In der Hydraulik.lib ist es möglich, bis zu 12 gepufferte Bewegungen zu kommandieren. Hierzu muss dem [MC\\_AxUtiStandardInit\\_BkPlcMc \[▶ 238\]](#) Baustein zum Aktualisieren der Achse-Referenz ein Kommandopuffer vom Typ `ST_TcPlcCmdBuffer_BkPlcMc` übergeben werden und zyklisch ein Baustein `MC_AxRtCmdBufferExecute_BkPlcMc` aufgerufen werden.

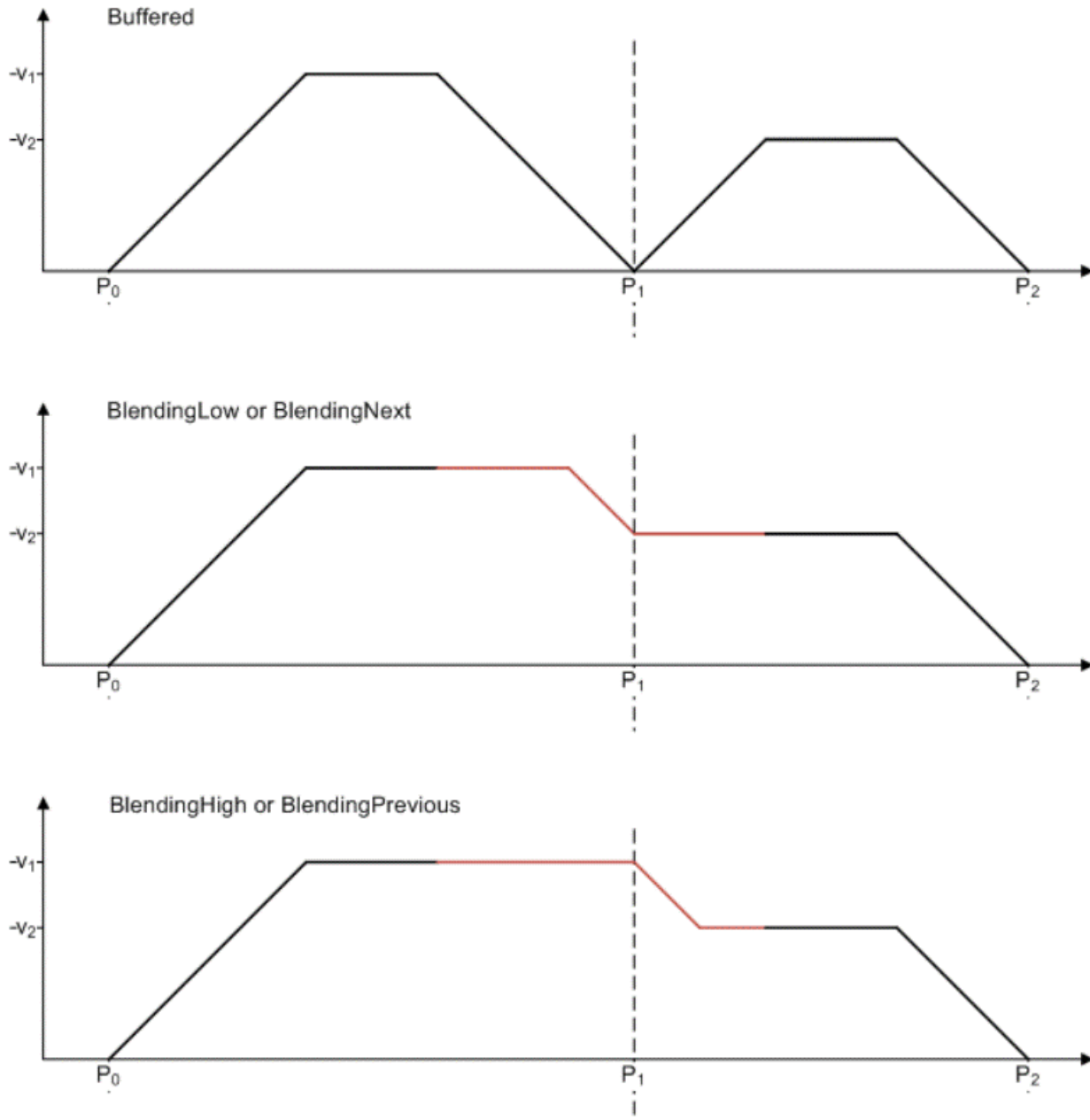
Wenn jetzt Move-Bausteine wie [MC\\_MoveAbsolute\\_BkPlcMc \[▶ 72\]](#), [MC\\_MoveRelative\\_BkPlcMc \[▶ 76\]](#) oder [MC\\_MoveVelocity\\_BkPlcMc \[▶ 78\]](#) aktiviert werden tragen sie ihre Daten in den Kommandopuffer ein.

Es ist im gepufferten Betrieb unbedingt drauf zu achten, dass die Move-Bausteine und der [MC\\_AxRuntime\\_BkPlcMc \[▶ 221\]](#) der Achse in einer PLC-Task laufen.

Übergang zwischen einem langsamen und einem schnellen Abschnitt.



Übergang zwischen einem schnellen und einem langsamen Abschnitt.



**FAQ #21: Wie kann auf Register einer Klemme zugegriffen werden an der ein Encoder oder ein Ventil einer Achse angeschlossen ist?**

Für die Registerkommunikation mit Klemmen an denen der Encoder oder das Ventil einer Achse angeschlossen sind wird die Verwendung von Bausteinen der Typen [MC\\_AxUtiReadRegDriveTerm\\_BkPlcMc \[▶ 276\]\(\)](#), [MC\\_AxUtiReadRegEncTerm\\_BkPlcMc \[▶ 277\]\(\)](#), [MC\\_AxUtiWriteRegDriveTerm\\_BkPlcMc \[▶ 285\]\(\)](#) und [MC\\_AxUtiWriteRegEncTerm\\_BkPlcMc \[▶ 286\]\(\)](#) empfohlen.

**FAQ #22: Wie ist die ASCII Datei für eine Linearisierungstabelle aufgebaut?**

Das Format einer ASCII Datei einer Linearisierungstabelle ist wie folgt festgelegt:

- Ein Linearisierungspunkt pro Zeile.
- Pro Zeile zuerst ein Geschwindigkeitswert und dann ein Ausgabewert.
- Die Geschwindigkeitswerte sind auf die Referenzgeschwindigkeit normalisiert. Sie liegen somit im Bereich -1.000 bis 1.000 einschließlich.
- Die Ausgabewerte sind auf die Vollaussteuerung normalisiert. Sie umfassen somit den Bereich -1.000 bis 1.000 vollständig.

- Vor dem ersten Wert einer Zeile dürfen White Space Characters (Leerzeichen, Tabulator) stehen.
- Zwischen den beiden Werten einer Zeile muss mindestens ein White Space Character (Leerzeichen, Tabulator) stehen.
- Zwischen den beiden Werten einer Zeile dürfen weitere White Space Characters (Leerzeichen, Tabulator) stehen.
- Als Dezimal-Trennzeichen sind Punkt und Komma zulässig.
- Zwischen einem negativen Vorzeichen und der ersten Ziffer sind keine Nichtziffern zulässig.
- Der erste Punkt legt das negative Tabellenende fest.
- Der Geschwindigkeitswert jedes weiteren Punktes muss höher (d.h. weniger negativ oder mehr positiv) sein als der seines Vorgängers.
- Es ist sinnvoll wenn der Ausgabewert eines Punktes höher (d.h. weniger negativ oder mehr positiv) als der seines Vorgängers ist, da es sonst zu einer negativen Steigung in diesem Bereich kommt. Dies würde durch einen Vorzeichenwechsel der Verstärkung eine Instabilität in einer eventuell aktiven Regelung erzeugen.
- Der Nullpunkt (d.h. beide Koordinaten des Punktes sind 0.000) ist mit anzugeben.

**Beispiel:** Die folgende (idealisierte) Tabelle beschreibt einen Zylinder der durch asymmetrische Wirkungsflächen (bedingt durch einseitige Kolbenstange) in negativer Bewegungsrichtung nur die halbe Geschwindigkeit der positiven Richtung erreicht. Es wird hier angenommen, dass der Zylinder mit einem Nullschnitt-Ventil mit Kennlinienknick bei 40% betrieben wird

normalisierte Geschwindigkeit	normalisierte Ausgabe
-0,500	-1,000
-0,430	-0,900
-0,360	-0,800
-0,290	-0,700
-0,220	-0,600
-0,150	-0,500
-0,080	-0,400
-0,060	-0,300
-0,040	-0,200
-0,020	-0,100
0,000	0,000
0,040	0,100
0,080	0,200
0,120	0,300
0,160	0,400
0,300	0,500
0,440	0,600
0,580	0,700
0,720	0,800
0,860	0,900
1,000	1,000

**FAQ #23: Wie können Kommandos des PlcMcManagers abgeblockt werden?**

In einigen Situationen kann die Auslösung von Kommandos durch den PlcMcManager problematisch sein. Dies wäre z. B. der Fall, wenn eine bestimmte Abfolge von Aktionen zwingend vollständig abgearbeitet werden muss. Um in diesen Fällen die versehentliche Kommandogabe durch den PlcMcManager zu verhindern kann mit der `MC_AxRtCommandsLocked_BkPlcMc` [► 246] Funktion eine Verriegelung im Status-Doppelwort der Achse eingetragen werden. Ist diese Verriegelung aktiv wird ein vom PlcMcManager gesendetes Kommando mit einem Schreibschutz-Fehler abgelehnt.



Es ist unbedingt erforderlich die Verriegelung nach Abarbeitung der zu schützenden Aktion aufzuheben. Dies gilt auch und besonders beim Auftreten von Fehlern.

Es steht ein [Beispiel \[►\\_357\]](#) zur Verfügung.

**FAQ #24: Welches Format haben Dateien mit Kennlinien-Daten**

Wenn eine Achse mit Komponenten für eine kennliniengesteuerte Linearisierung ausgerüstet ist können die Stützpunkte mit einem Baustein vom Typ MC\_LinTableExportToAsciiFile\_BkPlcMc() in eine Datei exportiert werden. Sollen solche Daten aus einer auf diese oder andere Weise erzeugten oder veränderten Datei importiert werden, steht der Baustein MC\_LinTableImportFromAsciiFile\_BkPlcMc() zur Verfügung.

Damit solche Dateien genutzt werden können ist das nachstehende Format streng einzuhalten.

- Für jeden Punkt ist eine Zeile zu verwenden.
- Jede Zeile muss mit CR/LF beendet werden.
- Jede Zeile enthält zuerst den normierten Geschwindigkeitswert im Bereich -1.0 bis 1.0.
- Nach mindestens einem Trennzeichen (Leerzeichen oder Tabulator) folgt der normierte Ausgabewert für den vollständigen Bereich von -1.0 bis 1.0.
- Der Ausgabewert muss eine konstante Schrittweise (Distanz zwischen zwei Punkten) aufweisen.
- Die erste Zeile enthält den negativen Endwert. Ihr Ausgabewert muss -1.0 sein.
- Die letzte Zeile enthält den positiven Endwert. Ihr Ausgabewert muss 1.0 sein.
- Es ist eine Zeile für den Ausgabewert 0.0 vorzusehen.
- Somit muss die Datei gleich viele Punkte im negativen und im positiven Bereich enthalten. Die Gesamt-Anzahl muss ungerade sein.
- Sowohl die Geschwindigkeit als auch der Ausgabewert folgen den gleichen Regeln:
  - Vorzeichen bei negativen Zahlen
  - Eine Ziffer
  - Eine Dezimalteilung: Komma oder Punkt
  - Mindestens eine Nachkommastelle
  - Optional: Ein Exponent, gekennzeichnet mit „e“, einem Minuszeichen und einer Ziffer
  - Beispiel: -1.81408951053528e-1 -5.0e-1
  - Beispiel: 0.333 0.5

## 5.2 Globale Konstanten (ab V3.0)

**Bit-Masken für Positionsnocken**

Diese Masken sind von der Applikation für die Bereitstellung von digitalen Wegnocken für bActPosCams in ST\_TcHydAxRtData zu verwenden.

Konstante	Beschreibung
bTcHydActPosCamPos	Zusammenfassung von bTcHydActPosCamHigh und bTcHydActPosCamUp.
bTcHydActPosCamHigh	Achse hat obere Zielposition erreicht.
bTcHydActPosCamUp	Achse befindet sich in der Nähe der oberen Zielposition.
bTcHydActPosCamDown	Achse befindet sich in der Nähe der unteren Zielposition.
bTcHydActPosCamLow	Achse hat untere Zielposition erreicht.
bTcHydActPosCamNeg	Zusammenfassung von bTcHydActPosCamLow und bTcHydActPosCamDown.

### Bit-Masken für Achsstatusinformationen

Diese Masken sind von der Applikation für die Abfrage von Status-Signalen in nStateDWord in ST\_TcHydAxRtData zu verwenden.

Konstante	Beschreibung
dwTcHydNsDwFunctional	Achse ist betriebsbereit.
dwTcHydNsDwReferenced	Achse ist referenziert.
dwTcHydNsDwSteady	Achse ist nicht aktiv.
dwTcHydNsDwInTargRng	Die Achse befindet sich innerhalb einer von fMonPositionRange in ST_TcHydAxParam vorgegebenen Entfernung von der Zielposition.
dwTcHydNsDwInTarget	Die Achse befindet sich seit einer von fMonTargetFilter vorgegebenen Zeit ununterbrochen innerhalb einer von fMonTargetRange in ST_TcHydAxParam vorgegebenen Entfernung von der Zielposition.
dwTcHydNsDwDontTouchProtected	Reserviert, nicht unterstützt.
dwTcHydNsDwStopped	Die letzte Bewegung der Achse wurde gestoppt, ohne die vorgegebene Zielposition zu erreichen.
dwTcHydNsDwBusy	Achse ist aktiv.
dwTcHydNsDwMoveUp	Achse bewegt sich in Richtung größerer Positionen.
dwTcHydNsDwMoveDown	Achse bewegt sich in Richtung kleinerer Positionen.
dwTcHydNsDwReferencing	Achse führt eine Referenzfahrt aus.
dwTcHydNsDwConstVelo	Achse bewegt sich mit konstanter Geschwindigkeit.
dwTcHydNsDwExtSetpointActive	Die Achse wird durch einen <a href="#">MC_AxRtSetExtGenValues_BkPlcMc</a> [▶ 236] Baustein kontrolliert.
dwTcHydNsDwStartedOver	Die Achse wurde durchgestartet, d.h. das letzte akzeptierte Kommando wurde wirksam während die Achse noch eine aktive Bewegung ausführte.
dwTcHydNsDwControlActive	Reserviert, nicht unterstützt.
dwTcHydNsDwErrState	Achse im Störzustand.

### Bit-Masken für Achsfreigabeinformationen

Diese Masken sind von der Applikation für die Bereitstellung von Freigabe-Signalen in nDeCtrlDWord in ST\_TcHydAxRtData zu verwenden.

Konstante	Beschreibung
dwTcHydDcDwCtrlEnable	Reglerfreigabe. Diese Freigabe ist Voraussetzung für die Ausgabe von Stellwert- und Regler-Ausgaben.
dwTcHydDcDwFdPosEna	Vorschubfreigabe in positiver Richtung. Diese Freigabe ist Voraussetzung für eine Ausgabe von Stellwert- und Regler-Ausgaben in Richtung steigender Positionswerte.
dwTcHydDcDwCtrlPosEna	Zusammenfassung von dwTcHydDcDwCtrlEnable und dwTcHydDcDwFdPosEna.
dwTcHydDcDwFdNegEna	Vorschubfreigabe in negativer Richtung. Diese Freigabe ist Voraussetzung für eine Ausgabe von Stellwert- und Regler-Ausgaben in Richtung sinkender Positionswerte.
dwTcHydDcDwCtrlNegEna	Zusammenfassung von dwTcHydDcDwCtrlEnable und dwTcHydDcDwFdNegEna.
dwTcHydDcDwRefIndex	Referenziernocken.
dwTcHydDcDwAcceptBlockedDrive	Reserviert, nicht unterstützt.
dwTcHydDcDwBlockedDriveDetected	Reserviert, nicht vollständig unterstützt. Dieses Signal unterdrückt einen eventuell wirksamen Geschwindigkeitsregler.



**Error-Codes**

Diese Konstanten werden für ErrorID-Ausgänge von Bausteinen und nErrorCode in ST\_TcHydAxRtData zu verwenden.

Konstante	Hexadezimal	Dezimal	Beschreibung
dwTcHydAdsErrNoError	0	0	Kein Fehler.
dwTcHydAdsErrUnknownPort	16#0006	6	ADS Port nicht bekannt. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• AMS NetID / ADS Port adressieren das falsche Laufzeitsystem oder den falschen Rechner</li> <li>• in der adressierten PLC läuft ein anderes Projekt</li> <li>• die Applikation ruft keinen <code>MC_AxAdsCommServer_BkPlcMc [▶ 263]()</code> Baustein auf</li> </ul>
dwTcHydAdsErrUnknownTarget	16#0007	7	Target Maschine nicht bekannt. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• AMS NetID / ADS Port adressieren das falsche Laufzeitsystem oder den falschen Rechner</li> <li>• das Zielsystem ist nicht gestartet</li> <li>• TwinCAT ist nicht gestartet</li> <li>• die Verbindung ist elektrisch / mechanisch unterbrochen</li> <li>• bei Kommunikation via Ethernet: Die TCP/IP Verbindung arbeitet nicht</li> </ul>
dwTcHydAdsErrInvalidIdxGroup	16#0702	1794	Unzulässiger IndexGroup. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• AMS NetID / ADS Port adressieren das falsche Laufzeitsystem oder den falschen Rechner</li> <li>• in der adressierten PLC läuft ein anderes Projekt</li> <li>• Softwarefehler der Applikation (falsche Kombination von ADS Port / IdxGroup / IdxOffset)</li> </ul>
dwTcHydAdsErrInvalidIdxOffset	16#0703	1795	Unzulässiger IndexOffset. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• AMS NetID / ADS Port adressieren das falsche Laufzeitsystem oder den falschen Rechner</li> <li>• in der adressierten PLC läuft ein anderes Projekt</li> <li>• Softwarefehler der Applikation (falsche Kombination von ADS Port / IdxGroup / IdxOffset)</li> <li>• versuchter Zugriff auf ein Array Element mit unzulässigem Index (out of bounds)</li> </ul>
dwTcHydAdsErrRdWrNotPermitted	16#0704	1796	Zugriff (Schreiben, Lesen) nicht zulässig. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• es wurde ein Schreibzugriff auf eine nicht beschreibbare Variable angefordert</li> </ul>
dwTcHydAdsErrInvalidSize	16#0705	1797	Nicht zulässige Größe (Anzahl Bytes). Mögliche Ursachen: <ul style="list-style-type: none"> <li>• Softwarefehler der Applikation (falsche Kombination von ADS Port / IdxGroup / IdxOffset)</li> </ul>
dwTcHydAdsErrIllegalValue	16#0706	1798	Nicht zulässiger Wert. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• der übertragene Wert liegt außerhalb von absoluten Grenzen des Parameters</li> <li>• der übertragene Wert liegt außerhalb von Grenzen des Parameters die durch andere bereits gültig gewordene Parameter festgelegt sind</li> </ul>
dwTcHydAdsErrNotReady	16#0707	1799	Nicht betriebsbereit. Mögliche Ursachen:

Konstante	Hexadezimal	Dezimal	Beschreibung
			<ul style="list-style-type: none"> <li>ein MC_Power_BkPlcMc Baustein wurde durch seinen Enable Eingang veranlasst eine nicht betriebsbereite Achse zu aktivieren</li> </ul>
dwTcHydAdsErrBusy	16#0708	1800	Bereits aktiv. Mögliche Ursachen: <ul style="list-style-type: none"> <li>die Achse könnte einen Auftrag nicht akzeptieren weil sie bereits mit einer anderen Aufgabe beschäftigt ist</li> </ul>
dwTcHydAdsErrNoFile	16#070C	1804	Reserviert: Datei fehlt / nicht zugreifbar.
dwTcHydAdsErrSyntax	16#070D	1805	Syntax in Kommando oder Datei unzulässig. Mögliche Ursachen: <ul style="list-style-type: none"> <li>beim Lesen einer im ASCII Format gespeicherten Kennlinien-Datei wurden nicht zulässige Zeichen oder Zeichenkombinationen erkannt</li> <li>beim Lesen einer im ASCII Format gespeicherten Kennlinien-Datei wurden unvollständige Angaben erkannt</li> </ul>
dwTcHydAdsErrTimeout	16#0745	1861	Zeitüberschreitung. Mögliche Ursachen: <ul style="list-style-type: none"> <li>in einer Kommunikation ist die Antwort nicht innerhalb einer vorgesehenen Zeit eingetroffen               <ul style="list-style-type: none"> <li>die Zeit ist zu kurz gewählt</li> <li>die Verbindung ist unterbrochen</li> </ul> </li> <li>der Prozess hat die Abarbeitung des Kommandos verhindert oder über das vorgesehene Maß hinaus verzögert</li> <li>die vorgegebenen Parameter des Kommandos haben den Zeitbedarf über das vorgesehene Maß hinaus erhöht</li> </ul>
dwTcHydAdsErrNoAmsAddr	16#0749	1865	AMS/ADS Adresse fehlt: <ul style="list-style-type: none"> <li>Die ADS Adresse des Geräts wurde nicht auf die entsprechende Variable der Input-Struktur gemappt.</li> </ul>
dwTcHydErrCdNotCompatible	16#4040	16448	Achse ist zur geforderten Funktion nicht kompatibel. Mögliche Ursachen: <ul style="list-style-type: none"> <li>Softwarefehler der Applikation</li> </ul>
dwTcHydErrCdIllegalOutputNumber	16#4104	16644	Ausgangsnummer außerhalb des zulässigen Bereichs. Mögliche Ursachen: <ul style="list-style-type: none"> <li>Ein MC_ReadDigitalOutput_BkPlcMc oder MC_WriteDigitalOutput_BkPlcMc Baustein wurde mit einem unzulässigen Parameter aufgerufen.</li> </ul>
dwTcHydErrCdNotSupport	16#4107	16647	Funktion oder Kommando wird nicht unterstützt. Mögliche Ursachen: <ul style="list-style-type: none"> <li>Softwarefehler der Applikation</li> </ul>
dwTcHydErrCdCycleTime	16#4205	16901	Zykluszeit (fCycleTime in ST_TcHydAxParam) nicht zulässig. Mögliche Ursachen: <ul style="list-style-type: none"> <li>Parametrierfehler</li> </ul>
dwTcHydErrCdMissingEnc	16#4210	16912	Es besteht keine Verbindung zu einem Encoder-Interface (pStDeviceInput und/oder pStDeviceOutput in <a href="#">Axis_Ref_BkPlcMc [► 84]</a> ). Mögliche Ursachen:

Konstante	Hexadezimal	Dezimal	Beschreibung
			<ul style="list-style-type: none"> <li>• Softwarefehler der Applikation (der MC_AxUtiStandardInit_BkPlcMc Baustein wurde nicht aufgerufen oder nicht mit der Adresse einer ST_TcPlcDeviceInput und einer ST_TcPlcDeviceOutput Struktur versorgt)</li> </ul>
dwTcHydErrCdMissingDrive	16#4212	16914	<p>Es besteht keine Verbindung zu einem Drive-Interface (pStDeviceInput und/oder pStDeviceOutput in <a href="#">Axis_Ref_BkPlcMc [► 84]</a>). Mögliche Ursachen:</p> <ul style="list-style-type: none"> <li>• Softwarefehler der Applikation (der MC_AxUtiStandardInit_BkPlcMc Baustein wurde nicht aufgerufen oder nicht mit der Adresse einer ST_TcPlcDeviceInput und einer ST_TcPlcDeviceOutput Struktur versorgt)</li> </ul>
dwTcHydErrCdCannotSynchronize	16#421A	16922	<p>Start-Distanz bei Aufruf eines MC_GearInPos_BkPlcMc() Bausteins nicht ausreichend. Mögliche Ursachen:</p> <ul style="list-style-type: none"> <li>• die Achse steht bei Aktivierung des Bausteins zu nah am Synchronisierungspunkt</li> <li>• die Dynamik-Parameter der Achse sind nicht ausreichend</li> </ul>
dwTcHydErrCdIllegalGearFactor	16#421B	16923	<p>Die Parameter einer Getriebekopplung sind nicht zulässig. Mögliche Ursachen:</p> <ul style="list-style-type: none"> <li>• der Parameter des Bausteins ist nicht zulässig</li> </ul>
dwTcHydErrCdSoftEnd	16#4222	16930	<p>Die Zielposition liegt jenseits eines aktiven Software-Endschalters und ist somit nicht zulässig.</p>
dwTcHydErrCdLowDist	16#4228	16936	<p>Die Fahrstrecke ist unzulässig klein.</p>
dwTcHydErrCdIllegalStartType	16#4239	16953	<p>Unzulässiger Start-Typ.</p>
dwTcHydErrCdCommandBufferOverflow	16#423F	16959	<p>Kommando Puffer ist voll.</p>
dwTcHydErrCdEncLostCam	16#4253	16979	<p>Reserviert, nicht unterstützt.</p>
dwTcHydErrCdCtrlEnaLost	16#4260	16992	<p>Reglerfreigabe wurde während Bewegung entzogen. Mögliche Ursachen:</p> <ul style="list-style-type: none"> <li>• auf Grund eines Signals der Maschinen-Logik wurde der Achse die Freigabe zu einem unerwarteten Zeitpunkt entzogen</li> <li>• Softwarefehler der Applikation</li> </ul>
dwTcHydErrCdEncNoCamFound	16#429C	17052	<p>Reserviert, nicht unterstützt.</p>
dwTcHydErrCdEncNoCamEnd	16#429D	17053	<p>Reserviert, nicht unterstützt.</p>
dwTcHydErrCdEncNoSyncPulse	16#429E	17054	<p>Reserviert, nicht unterstützt.</p>
dwTcHydErrCdAcc	16#4309	17161	<p>Die Beschleunigung ist nicht zulässig.</p>
dwTcHydErrCdDec	16#430A	17162	<p>Die Verzögerung ist nicht zulässig.</p>
dwTcHydErrCdJerk	16#430B	17163	<p>Die Ruckbegrenzung ist nicht zulässig.</p>
dwTcHydErrCdPtrPlcMc	16#4345	17221	<p>Es besteht keine Verbindung zu einem der benötigten Achs-Interfaces (pStDeviceInput oder pStDeviceOutput in <a href="#">Axis_Ref_BkPlcMc [► 84]</a>).</p>
dwTcHydErrCdPtrMcPlc	16#4346	17222	<p>Es besteht keine Verbindung zu einem der benötigten Achs-Interfaces (pStDeviceInput oder pStDeviceOutput in <a href="#">Axis_Ref_BkPlcMc [► 84]</a>).</p>

Konstante	Hexadezimal	Dezimal	Beschreibung
dwTcHydErrCdCtrlEna	16#4356	17238	Bewegung ohne Reglerfreigabe unzulässig.
dwTcHydErrCdNegFdEna	16#4357	17239	Bewegung in Richtung sinkender Positionen ohne Vorschubfreigabe in negativer Richtung unzulässig.
dwTcHydErrCdPosFdEna	16#4358	17240	Bewegung in Richtung steigender Positionen ohne Vorschubfreigabe in positiver Richtung unzulässig.
dwTcHydErrCdSetVelo	16#4359	17241	Die geforderte Geschwindigkeit ist nicht zulässig.
dwTcHydErrCdPehTimeout	16#435C	17244	Die Achse erreicht das Zielfenster nicht innerhalb der vorgegebenen Zeit.
dwTcHydErrCdNotMoving	16#435D	17245	Die Achse bewegt sich nicht oder nicht in die richtige Richtung.
dwTcHydErrCdConsequential	16#43A0	17312	Folgefehler: Die Achse ist durch ein Problem bei einer anderen Achse ebenfalls in den Stöorzustand versetzt worden.
dwTcHydErrCdEncType	16#4401	17409	Der Parametertyp ist ungültig.
dwTcHydErrCdEncScaling	16#4406	17414	Die Inkrementskalierung ist unzulässig.
dwTcHydErrCdEncSyncDist	16#4414	17428	Der Abstand zwischen Latch_Enable und Sync-Puls ist zu klein.
dwTcHydErrCdEncSetActPos	16#4422	17442	Beim Istwertsetzen ist ein Problem aufgetreten.
dwTcHydErrCdPtrPlcEncln	16#4442	17474	Die Achse besitzt keinen Zeiger auf ein Encodereingangsinterface
dwTcHydErrCdPtrPlcEncOut	16#4443	17475	Die Achse besitzt keinen Zeiger auf ein Encoderausgangsinterface.
dwTcHydErrCdEncUnderrun	16#4450	17488	Von einigen Encoder-Typen gemeldet: Die Istposition hat die untere Zählgrenze des Encoders passiert.
dwTcHydErrCdEncOverrun	16#4451	17489	Von einigen Encoder-Typen gemeldet: Die Istposition hat die obere Zählgrenze des Encoders passiert.
dwTcHydErrCdEncHdwFailed	16#4464	17508	Antriebssteller oder Encoder melden einen Hardwarefehler.
dwTcHydErrCdSsi	16#4470	17520	Beim Betrieb eines SSI-Encoders wurde ein Fehler erkannt.
dwTcHydErrCdPosLag	16#4550	17744	Der Schleppabstand überschreitet eine aktive Grenze.
dwTcHydErrCdDriveType	16#4601	17921	Der in nDrive_Type eingestellte Wert ist nicht zulässig.
dwTcHydErrCdRefVelo	16#4605	17925	Bezugsgeschwindigkeit (fRefVelo in ST_TcHydAxParam) nicht zulässig.
dwTcHydErrCdStepperStalled	16#4636	17974	Eine Stall-Situation wurde erkannt.
dwTcHydErrCdPtrPlcDriveIn	16#4642	17986	Die Achse besitzt keinen Zeiger auf ein Antriebseingangsinterface.
dwTcHydErrCdPtrPlcDriveOut	16#4643	17987	Die Achse besitzt keinen Zeiger auf ein Antriebsausgangsinterface.
dwTcHydErrCdDriveNotReady	16#4650	18000	Leistungsteil nicht betriebsbereit.
dwTcHydErrCdTblEntryCount	16#4A02	18946	Die Anzahl der Tabelleneinträge (Zeilen) ist nicht zulässig.
dwTcHydErrCdTblInvalidMasterStep	16#4A04	18948	Die Tabelle enthält Einträge mit einer unzulässigen Master-Schrittweite.
dwTcHydErrCdTblNoInit	16#4A10	18960	Die Tabelle ist nicht initialisiert.
dwTcHydErrCdTblIllegalIndex	16#4A13	18963	Tabellenindex nicht zulässig.

Konstante	Hexadezimal	Dezimal	Beschreibung
dwTcHydErrCdTblLineCount	16#4A15	18965	Die Anzahl der Tabelleneinträge ist zu groß.
dwTcHydErrCdNotStartable	16#4B01	19201	Achse ist in einem Zustand, der einen Start nicht zulässt.
dwTcHydErrCdFuncTimeout	16#4B07	19207	Funktion wurde nicht in der vorgegebenen Zeit fertig gemeldet.
dwTcHydErrCdNotReady	16#4B09	19209	Achse ist in einem nicht betriebsbereiten Zustand.
dwTcHydErrCdHomingType	16#4F00	20224	Referenziermethode (nEnc_HomingType in ST_TcHydAxParam) nicht zulässig.
dwTcHydErrCdEncCutOff	16#4F01	20225	Die Grenzfrequenz der Istwerterfassung wurde überschritten.
dwTcHydErrCdIllegalDistance	16#4F02	20226	Distanz ist unzulässig: Null oder negativ.
dwTcHydErrEncDisconnected	16#4FF0	20464	Encoderhardware ist abgekoppelt. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• die Feldbusverbindung ist unterbrochen</li> <li>• die Stromversorgung des Geräts ist nicht verfügbar</li> <li>• das Gerät ist ausgefallen</li> <li>• ein in der Feldbusverbindung zwischen der Steuerung und dem Gerät angeordnetes anderes Gerät ist nicht mit Strom versorgt oder ausgefallen</li> </ul>
dwTcHydErrDriveDisconnected	16#4FF1	20465	Antriebshardware ist abgekoppelt. Mögliche Ursachen: <ul style="list-style-type: none"> <li>• die Feldbusverbindung ist unterbrochen</li> <li>• die Stromversorgung des Geräts ist nicht verfügbar</li> <li>• das Gerät ist ausgefallen</li> <li>• ein in der Feldbusverbindung zwischen der Steuerung und dem Gerät angeordnetes anderes Gerät ist nicht mit Strom versorgt oder ausgefallen</li> </ul>
dwTcHydErrDistanceInsufficient	16#4FF2	20466	Fahrweg nicht ausreichend.
dwTcHydErrIllegalAreas	16#4FF3	20467	Unzulässige Wirkungsflächen: <ul style="list-style-type: none"> <li>• für die Zylinder-Flächen auf dem Ventil-Reiter sind unzulässige Werte eingetragen</li> <li>• die Kombination der eingetragenen Flächen ist so nicht zulässig</li> </ul>
dwTcHydErrIncompleteImplementation	16#4FF4	20468	Die Implementation der Achse ist unvollständig: <ul style="list-style-type: none"> <li>• obwohl die Achse in ihren Parametern mit bDrivelsHybrid gekennzeichnet ist wird kein MC_AxRtHybridAxisActuals_BkPlcMc-Baustein aufgerufen</li> <li>• die selbe Instanz vom Typ ST_TcPlcDeviceInput wird auch dem MC_AxUtiStandardInit_BkPlcMc Baustein einer anderen Achsen übergeben</li> <li>• die selbe Instanz vom Typ ST_TcPlcDeviceOutput wird auch dem MC_AxUtiStandardInit_BkPlcMc Baustein einer anderen Achsen übergeben</li> <li>• einer weiteren Achse wird eine andere Instanz von ST_TcPlcMcLogBuffer übergeben</li> <li>• einer weiteren Achse wurde der selbe gültige Pointer auf eine Instanz von ST_TcPlcCmdBuffer_BkPlcMc übergeben</li> </ul>

Konstante	Hexadezimal	Dezimal	Beschreibung
			<ul style="list-style-type: none"> <li>einer weiteren Achse wurde der selbe gültige Pointer auf eine Instanz von ST_TcMcAutoldent übergeben</li> </ul>

### Gerätespezifische Error-Codes des Bausteins MC\_Power\_BkPlcMc

Diese Werte erscheinen am **ErrorID** Ausgang eines MC\_Power\_BkPlcMc Bausteins, wenn ein Fehler vom externen Gerät gemeldet wird.

Konstante	Hexadezimal	Dezimal	Beschreibung
dwTcHydErrCdAX2000MainPwrTmOut	16#0001	1	Nur bei AX2000: Keine Rückmeldung durch das Netzschütz (Zeitüberschreitung beim Warten auf ST_TcPlcMcAx2000In.bPowerOk).
dwTcHydErrCdAX2000MainPwrFault	16#0002	2	Nur bei AX2000: Fallende Flanke an Rückmeldung des Netzschütz (ST_TcPlcMcAx2000In.bPowerOk).
dwTcHydErrCdAX2000PwrStageTmOut	16#0003	3	Nur bei AX2000: Keine Rückmeldung von AX-Endstufe (Zeitüberschreitung beim Warten auf ST_TcPlcMcAx2000In.DriveState[3].6, kein Ready).
dwTcHydErrCdAX2000PwrStageFault	16#0004	4	Nur bei AX2000: Fallende Flanke von AX-Endstufe (ST_TcPlcMcAx2000In.DriveState[3].6, kein Ready).
dwTcHydErrCdAX2000ReportsError	16#0005	5	Nur bei AX2000: Fehlermeldung durch AX-Gerät (ST_TcPlcMcAx2000In.DriveState[3].7 oder ST_TcPlcMcAx2000In.DriveError<>0).
dwTcHydErrCdAX2000ErrorI2T	16#0006	6	Nur bei AX2000: I <sup>2</sup> T-Fehlermeldung von AX-Endstufe (ST_TcPlcMcAx2000In.DriveState[0].0).
dwTcHydErrCdAX2000ErrorChopper	16#0007	7	Nur bei AX2000: Bremswiderstand der AX-Endstufe defekt (ST_TcPlcMcAx2000In.DriveState[0].1).
dwTcHydErrCdAX2000ErrorWatchDog	16#0008	8	Nur bei AX2000: Watchdog (Timeout bei Kommunikation) der AX-Endstufe hat angesprochen (ST_TcPlcMcAx2000In.DriveState[0].3).
dwTcHydErrCdAX2000ErrorPwrLine	16#0009	9	Nur bei AX2000: Versorgungsfehler durch AX-Endstufe gemeldet (ST_TcPlcMcAx2000In.DriveState[0].4).
dwTcHydErrCdAX2000ConnectionLost	16#000A	10	Nur bei AX2000: Die Verbindung zum AX-Gerät ist unterbrochen oder erheblich gestört (ST_TcPlcMcAx2000In.BoxState<>0).
dwTcHydErrCdAX2000ConnectionTmOut	16#000B	11	Nur bei AX2000: Die Kommunikation mit dem AX-Gerät konnte nicht aufgenommen werden (Zeitüberschreitung).
dwTcHydErrCdKL2531OverTemp	16#0001	1	Nur bei KL2531/KL2541: Die KL2531/KL2541-Klemme meldet Übertemperatur-Alarm.
dwTcHydErrCdKL2531UnderVoltage	16#0002	2	Nur bei KL2531/KL2541: Die KL2531/KL2541-Klemme meldet eine zu geringe Versorgungsspannung auf der Power Rail.
	16#0003	3	Nur bei KL2531/KL2541: Reserviert.
dwTcHydErrCdKL2531OpenLoadA	16#0004	4	Nur bei KL2531/KL2541: Die KL2531/KL2541-Klemme meldet Drahtbruch auf A-Seite.
dwTcHydErrCdKL2531OpenLoadB	16#0005	5	Nur bei KL2531/KL2541: Die KL2531/KL2541-Klemme meldet Drahtbruch auf B-Seite.
dwTcHydErrCdKL2531OverCurrentA	16#0006	6	Nur bei KL2531/KL2541: Die KL2531/KL2541-Klemme meldet Überstrom an Endstufe A.
dwTcHydErrCdKL2531OverCurrentB	16#0007	7	Nur bei KL2531/KL2541: Die KL2531/KL2541-Klemme meldet Überstrom an Endstufe B.

Konstante	Hexadezimal	Dezimal	Beschreibung
dwTcHydErrCdKL2531NotReady	16#0008	8	Nur bei KL2531/KL2541: Die Klemme meldet ein Endstufenproblem (Enabled, nicht Ready).
dwTcHydErrCdKL2531ConnectionLost	16#000A	10	Nur bei KL2531/KL2541: Die Verbindung zum Klemme ist unterbrochen oder erheblich gestört (ST_TcPlcMcDriveIn.uiBoxState<>0).
dwTcHydErrCdKL2531ConnectionTmOut	16#000B	11	Nur bei KL2531/KL2541: Die Kommunikation mit der Klemme konnte nicht aufgenommen werden (Zeitüberschreitung).

**ADS-Codes**

Diese Konstanten werden von den MC\_AxAdsReadDecoder und MC\_AxAdsWriteDecoder Bausteinen akzeptiert.

IndexGroup	IndexOffset	Typ	R/W	Beschreibung
16#4000 + Achsindex	2	STRING()	R	Achsname als Text.
	4	UDINT	R	Zykluszeit in Mikrosekunden.
	16#10003	UDINT	R	Encodertyp: nEnc_Type aus ST_TcHydAxParam.
	16#10006	LREAL	R	Inkrementbewertung: fEnc_IncWeighting aus ST_TcHydAxParam.
	16#30003	UDINT	R	Drivetyp: nDrive_Type aus ST_TcHydAxParam.
16#4100 + Achsindex	1	UDINT	R	Errorcode: nErrorCode aus ST_TcHydAxRtData.
	16#10002	LREAL	R	Istposition: fActPos aus ST_TcHydAxRtData.
	16#10005	LREAL	R	Istgeschwindigkeit: fActVelo aus ST_TcHydAxRtData.
16#4200 + Achsindex	1	-	W	Achsreset ausführen.
	16#10	-	W	Referenzfahrt starten.
	16#21	Struktur	W	Achsbewegung starten.
	16#FFFF0001	-	W	Parameter speichern.
	16#FFFF0002	-	W	Parameter laden.
16#4300 + Achsindex	16#81	UDINT	R	Statusdoppelwort: nStateDWord aus ST_TcHydAxRtData.
	16#B1	UDINT	R	Errorcode: nErrorCode aus ST_TcHydAxRtData.
16#F000 + Achsindex	1	Struktur	R	Die ST_TcHydAxRtData Variable der Achse.
	2	Struktur	R/W	Die ST_TcHydAxParam Variable der Achse.
16#800F0000 + Achsindex	<a href="#">E_TcMCPParameter</a> <a href="#">[► 97]</a>		R/W	Parameter und Aktualwerte der Achse.
16#FFFFFFFF	0	String()	R	Kennung des Servers.
	1	UINT	R	Major Version der Library.
	2	UINT	R	Minor Version der Library.
	3	UINT	R	Release der Library.
	4	UINT	R	Anzahl der unterstützten Achsen

## Array Dimensions

Die folgenden Konstanten werden für die Dimensionierung von Feldern verwendet und können von der Applikation genutzt werden.

Konstante	Beschreibung
ciBkPlcMc_CamSwitchRef_MinIdx	Lower boundary index on an array[] of <a href="#">CAMSWITCH_REF_BkPlcMc [▶ 114]</a> , supplied to blocks of type <a href="#">MC_DigitalCamSwitch_BkPlcMc [▶ 53]</a>
ciBkPlcMc_CamSwitchRef_MaxIdx	Upper boundary index on an array[] of <a href="#">CAMSWITCH_REF_BkPlcMc [▶ 114]</a> , supplied to blocks of type <a href="#">MC_DigitalCamSwitch_BkPlcMc [▶ 53]</a>
ciBkPlcMc_TrackRef_MinIdx	Lower boundary index on an array[] of <a href="#">TRACK_REF_BkPlcMc [▶ 116]</a> , supplied to blocks of type <a href="#">MC_DigitalCamSwitch_BkPlcMc [▶ 53]</a>
ciBkPlcMc_TrackRef_MaxIdx	Upper boundary index on an array[] of <a href="#">TRACK_REF_BkPlcMc [▶ 116]</a> , supplied to blocks of type <a href="#">MC_DigitalCamSwitch_BkPlcMc [▶ 53]</a>

## Logger Levels

Die folgenden Konstanten werden für die Festlegung des Levels verwendet, ab der Meldungen in die Logger-Funktion der Library aufgenommen werden.

Konstante	Beschreibung
dwTcHydLogLevel_None	Keine Aufzeichnung
dwTcHydLogLevel_Errors	Nur Fehlermeldungen
dwTcHydLogLevel_Warnings	Fehlermeldungen und Warnungen
dwTcHydLogLevel_Actions	Fehlermeldungen, Warnungen und Aktivitäten

## Logger Sources

Die folgenden Konstanten werden für die Festlegung der Quelle einer Meldung in der Logger-Funktion der Library verwendet.

Konstante	Beschreibung
dwTcHydLogSource_Library	Ein Baustein der Hydraulik-Library
dwTcHydLogSource_LibExt_2R2V	Ein Baustein der 2R2V Library
dwTcHydLogSource_Application	Ein Baustein der Applikation
dwTcHydLogSource_ApplicationFramework	Ein Baustein einer Applikation-Plattform

## Logger Argument Types

Die folgenden Konstanten werden für die Festlegung des Typs eines optionalen Parameters einer Meldung in der Logger-Funktion der Library verwendet.

Konstante	Beschreibung
dwTcHydLogArgType_DInt	Die Meldung enthält einen Parameter vom Typ DINT. Im Text der Meldung muss ein Platzhalter der Form %d erscheinen.
dwTcHydLogArgType_LReal	Die Meldung enthält einen Parameter vom Typ LREAL. Im Text der Meldung muss ein Platzhalter der Form %f erscheinen.
dwTcHydLogArgType_String	Die Meldung enthält einen Parameter vom Typ STRING. Im Text der Meldung muss ein Platzhalter der Form %s erscheinen.

## 5.3 Ventil

Das Ventil ist in der Regel das Stellglied, wodurch die Achse geregelt wird. Unter den Stetigventilen wird zwischen:



- Servoventil
- Proportionalventil
- Regelventil

unterschieden.

### Servoventil

Diese Ventile steuern über kleine elektrische Signale große Ölströme

- Ein kleiner Torque-Motor regelt das angeschlossene Steueröl, wodurch der Schieber der Hauptstufe verstellt wird.
- Oft mehrstufig ausgelegt
- hohe Reaktionsfähigkeit und Regelbarkeit

### Proportionalventil

Ein Spulenstrom erzeugt eine proportionale Kraft, die den Schieber des Ventils gegen die Kraft einer Feder verstellt.

Im Vergleich zum Servoventil:

- Größere Sprungantwortzeit
- Größerer Stromverbrauch
- Größere Hysterese
- Robuster gegenüber Verschmutzung
- Günstiger im Preis

### Regelventil:

Ist ein Proportionalventil, bei dem die Schieberposition gemessen und nachgeführt wird, dadurch:

- Kleinere Sprungantwortzeit
- Kleinere Hysterese
- Weniger Lastrückwirkung
- Aufwändiger und teurer als Proportionalventile
- Elektronik auf dem Ventil oder im Schaltschrank

### Grundlagen zum Lesen von Ventildatenblättern

Das Stetigventil ist in der Regel das Stellglied vom Regler und von Hersteller zu Hersteller und von Typ zu Typ sehr unterschiedlich gefertigt. Um die Ausgabeskalierung auf die Gegebenheiten dieses Ventils anzupassen muss bei der Inbetriebnahme das Ventildatenblatt des Stetigventils vorliegen. An einem Ventil gibt es verschiedene Hydraulische Anschlüsse. A und B sind die Ventilausgänge wobei A an die Zylinderseite mit der größeren Kolbenfläche und B an die Zylinderseite mit der kleineren Kolbenfläche angeschlossen wird. P und T stellen die Versorgungsanschlüsse darstellen. P ist die Druckleitung und T ist die Rückleitung in den Tank.



In der Hydraulikbibliothek wird unter A-Seite immer die Seite verstanden, die positiv fahrend ist und B-Seite ist die negativ fahrende Seite.

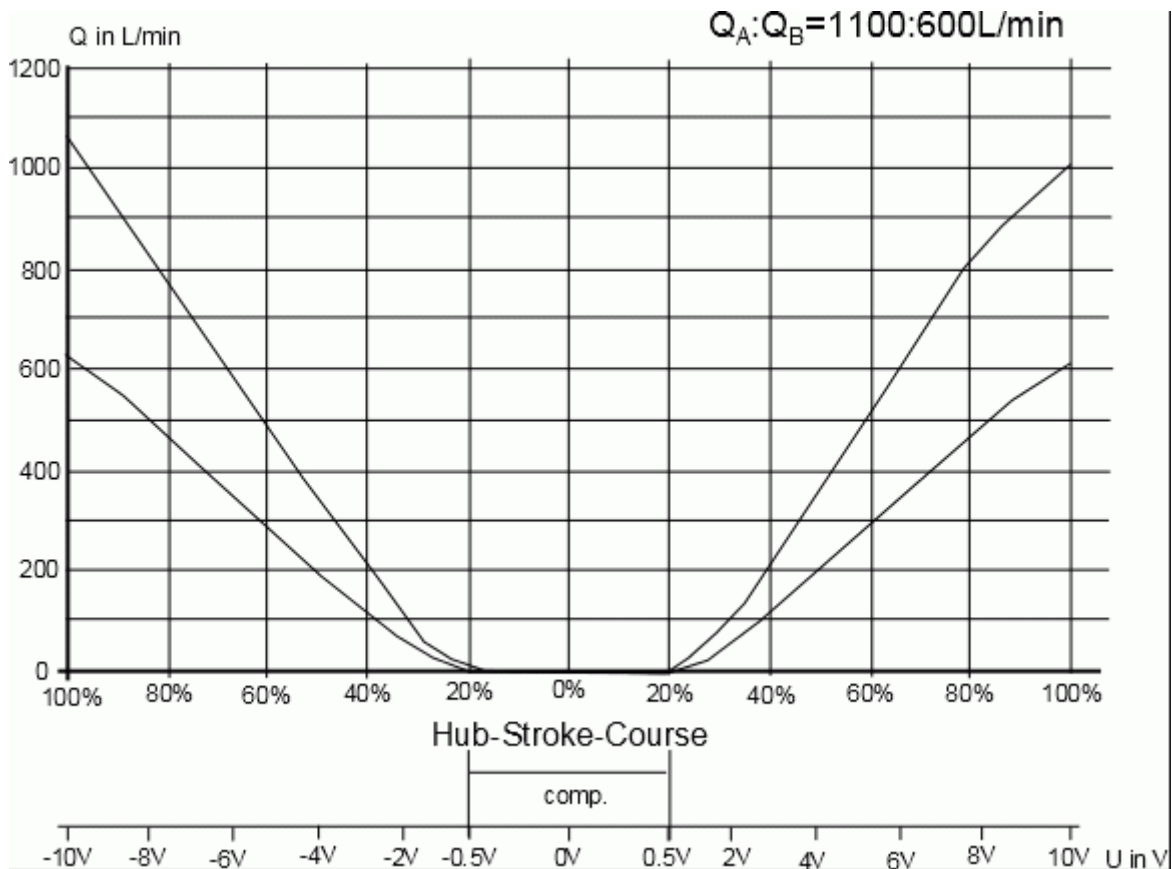
---

Häufig muss der Ventilschieber einen gewissen Hub vornehmen, bevor ein Ölfluss zu erkennen ist. Dieser Hub wird im Datenblatt des Ventils unter Überdeckung gelistet.



In einigen Fällen handelt es sich laut Datenblatt um ein überdecktes Ventil, dessen Überdeckung in der Elektronik des Ventils kompensiert wird.

---



Die Volumenstromkennlinie gibt die wichtigsten Informationen des Ventils wieder. In der Abbildung oben ist zu erkennen, dass der Kolben selbst eine Überdeckung von 20% aufweist, diese aber in der Elektronik des Ventils auf 5% verringert wurde. Dadurch ist keine Überdeckungskompensation über die Hydraulikbibliothek vorzunehmen.



Nur weil die Überdeckungskompensation im Ventil vorgenommen worden ist, handelt es sich nicht um ein Nullschnittventil und somit ist eine Achse auch hier nur in Grenzen lageregelungsfähig.

Es ist zu erkennen, dass der Ölstrom in die A-Kammer des Kolbens größer ist als der Ölstrom in die B-Kammer. Diese Asymmetrie ist eine Flächenkompensation im Ventil, in diesem Fall von 11:6.

## 5.4 Elektrisch-hydraulische Hybrid-Achsen

### Konzepte hybrider Achsen

Für den Aufbau einer solchen Achse wurde von den Herstellern hydraulischer Komponenten eine Reihe von Lösungen erarbeitet. Um nicht für jedes Modell jedes Herstellers eine dedizierte Lösung erstellen zu müssen wurden verallgemeinerte Konzepte entwickelt, die eine Gruppe von sich vergleichbar verhaltenden Modellen zusammenfassen. Dabei bezeichnen die nachgestellten Buchstaben (a, b, c, ...) äquivalente Varianten zu einem Konzept. Im Nachfolgenden werden diese Konzepte und Varianten anhand von Beispielen vorgestellt. Die Liste dieser Beispiele ist naturgemäß unvollständig.



Die nachstehend gezeigten Bildschirmhalte sind nur sichtbar, wenn auf dem Ventil-Reiter die Markierung ‚hybrid‘ gesetzt ist.

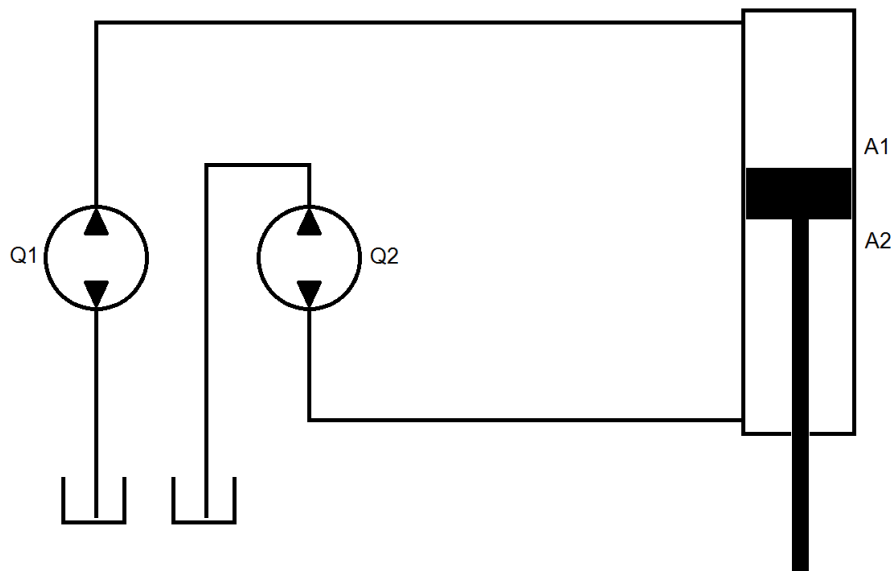
## Vereinfachte Darstellung

Die nachstehend aufgeführten Konzepte erfordern für ihre sichere und langfristige Funktion den Einsatz diverser Druckbegrenzer und Nachsaugventile. Diese Komponenten sind unverzichtbar, haben aber keinen direkten Einfluss auf die grundsätzliche Funktion der Achse. Zum besseren Verständnis sind alle Schaltungen mehr oder weniger vereinfacht und nicht als Dokumentation eines realen Produkts zu verstehen.

### 1: Nachbildung eines Gleichgang-Zylinders ohne Regenerativ-Schaltung

Hier wird unter Verwendung eines hydraulischen Gleichgang- oder Differential-Zylinders und einer angepassten Pumpenanordnung das steuerungstechnische Verhalten eines Gleichgang-Zylinders nachgebildet. Eine Getriebeschaltung kann durch eine Pumpenumschaltung realisiert werden, ergibt dann jedoch ein anderes Konzept.

#### 1a: Nachbildung des Gleichgang-Zylinders mit einem Differential-Zylinder



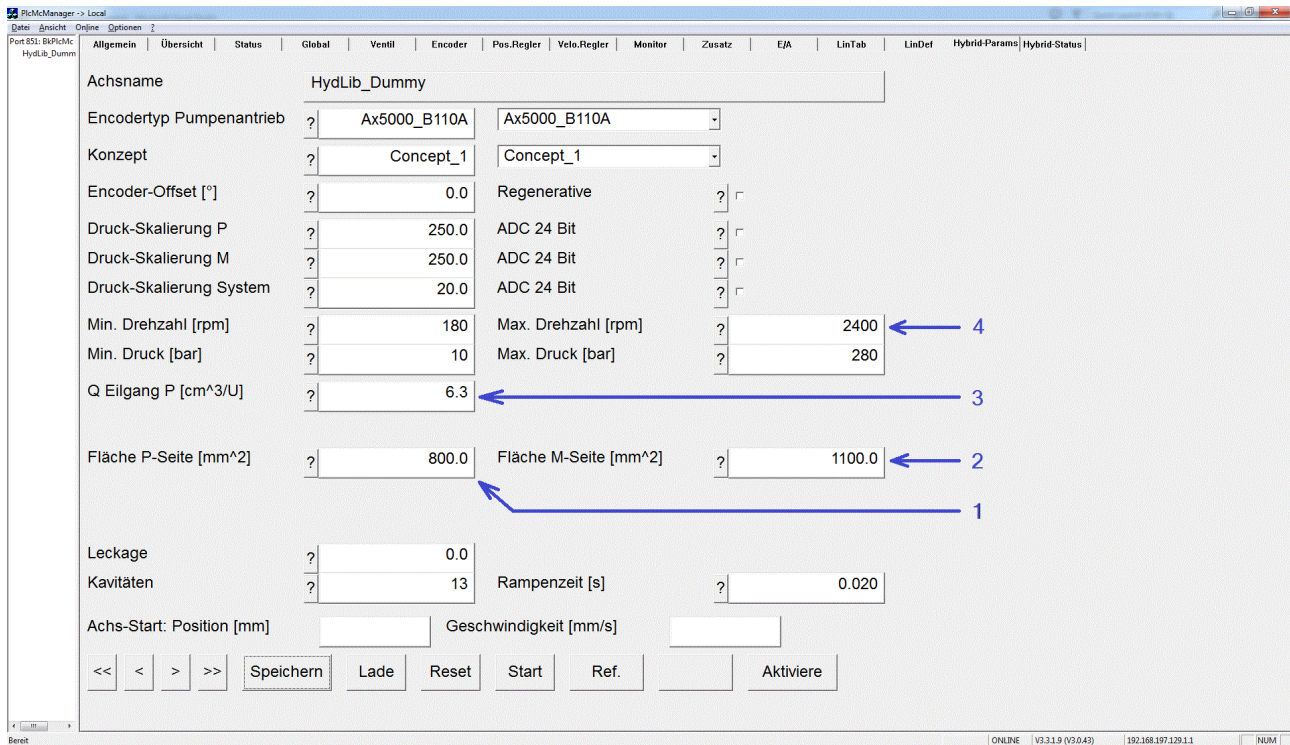
Hersteller / Produkt: Voith Turbo / CLDP.

Hier werden zwei in ihrem Volumenstrom an die Zylinderflächen angepasste Pumpen auf einer Motorwelle betrieben. Dabei sind die Fördermengen der Pumpen Q1 / Q2 an das Verhältnis der Wirkungsflächen anzupassen. Da dies nicht immer situationsunabhängig perfekt möglich ist können sich komplexe Druckverteilungen ergeben.

Für die Steuerung ergibt sich das Verhalten eines Gleichgang-Zylinders mit richtungsunabhängiger Vorschub-Konstante.

Hydraulisch liegt ein Differential-Zylinder vor und es ist ein Austauschvolumen zu speichern.

Es ist keine Getriebeschaltung verfügbar. Sie kann durch synchron mengenumschaltbare Pumpen realisiert werden. Dies ergibt ein anderes Konzept.

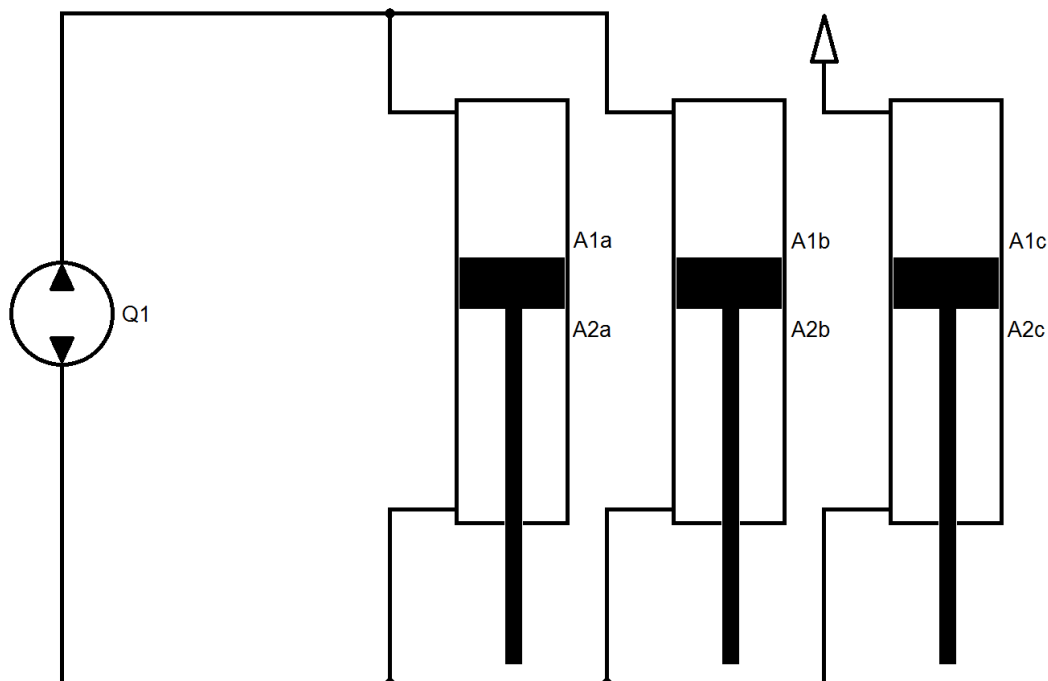


Benötigte Parameter: Wirkungsfläche in positiver Richtung (1), Wirkungsfläche in negativer Richtung (2), Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung (3), maximale Pumpendrehzahl (4).

Automatisch berechnete Parameter: Volumen pro Umdrehung an der Wirkungsfläche in negativer Richtung. Dabei wird das Verhältnis der Wirkungsflächen verwendet.

Automatisch festgelegte Parameter: Die zuschaltbaren Flächen sind 0, die Umdrehungs-Volumina für den Kraftgang sind gleich den Werten für den Eilgang.

**1b: Nachbildung eines Gleichgang-Zylinders mit mehreren Differential-Zylindern**



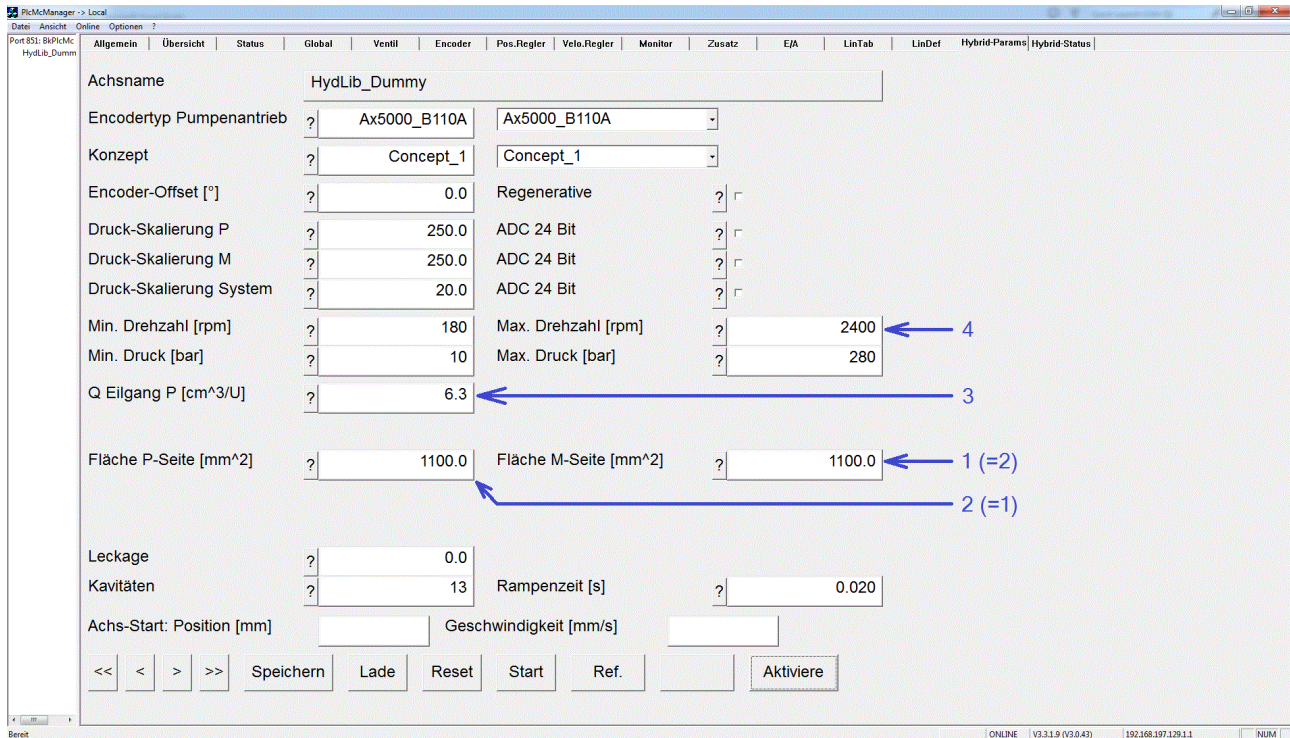
Hersteller / Produkt: Bucher / Demo HMI2018.

Hier werden drei Differential-Zylinder mit Flächenverhältnis 2:3 mechanisch parallel geschaltet. Die drei kleineren Flächen A2abc sind hydraulisch parallel geschaltet und bilden eine Wirkungsfläche. Von den größeren Flächen sind A1ab hydraulisch parallel geschaltet, während die dritte Fläche A1c belüftet ist.

Für die Steuerung ergibt sich ein zusammengesetzter Gleichgang-Zylinder mit richtungsunabhängiger Vorschub-Konstante.

Hydraulisch liegt ein Gleichgang-Zylinder vor und es ist kein Austauschvolumen zu speichern.

Laut Hersteller ist eine Getriebeschaltung möglich. Dies ergibt ein anderes Konzept.



Benötigte Parameter: Wirkungsfläche in positiver Richtung (1), Wirkungsfläche in negativer Richtung (2), Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung (3), maximale Pumpendrehzahl (4).

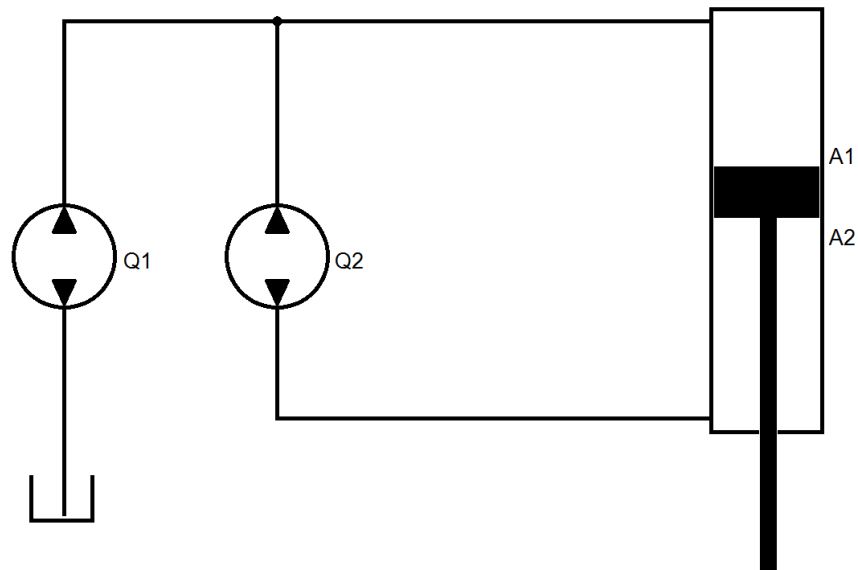
Automatisch berechnete Parameter: Volumen pro Umdrehung an der Wirkungsfläche in negativer Richtung. Dabei wird das Verhältnis der Wirkungsflächen (hier 1:1) verwendet.

Automatisch festgelegte Parameter: Die zuschaltbaren Flächen sind 0, die Umdrehungs-Volumen für den Kraftgang sind gleich den Werten für den Eilgang.

## 2: Nachbildung eines Gleichgang-Zylinders mit Regenerativ-Schaltung

Hier wird unter Verwendung eines hydraulischen Differential-Zylinders und einer angepassten Pumpenanordnung in einer Regenerativ-Schaltung das steuerungstechnische Verhalten eines Gleichgang-Zylinders nachgebildet. Eine Getriebeschaltung kann durch eine Pumpenumschaltung realisiert werden, ergibt dann jedoch ein anderes Konzept.

## 2a: Nachbildung eines Gleichgang-Zylinders mit Regenerativ-Schaltung



Hersteller / Produkt: Bosch Rexroth / Applikation.

Hier werden zwei in ihrem Volumenstrom an die Zylinderflächen angepasste Pumpen auf einer Motorwelle betrieben. Dabei sind die Fördermengen der Pumpen Q1 / Q2 an das Flächenverhältnis von Stangenquerschnitt / Ringfläche anzupassen. Da dies nicht immer situationsunabhängig perfekt möglich ist können sich komplexe Druckverteilungen ergeben.

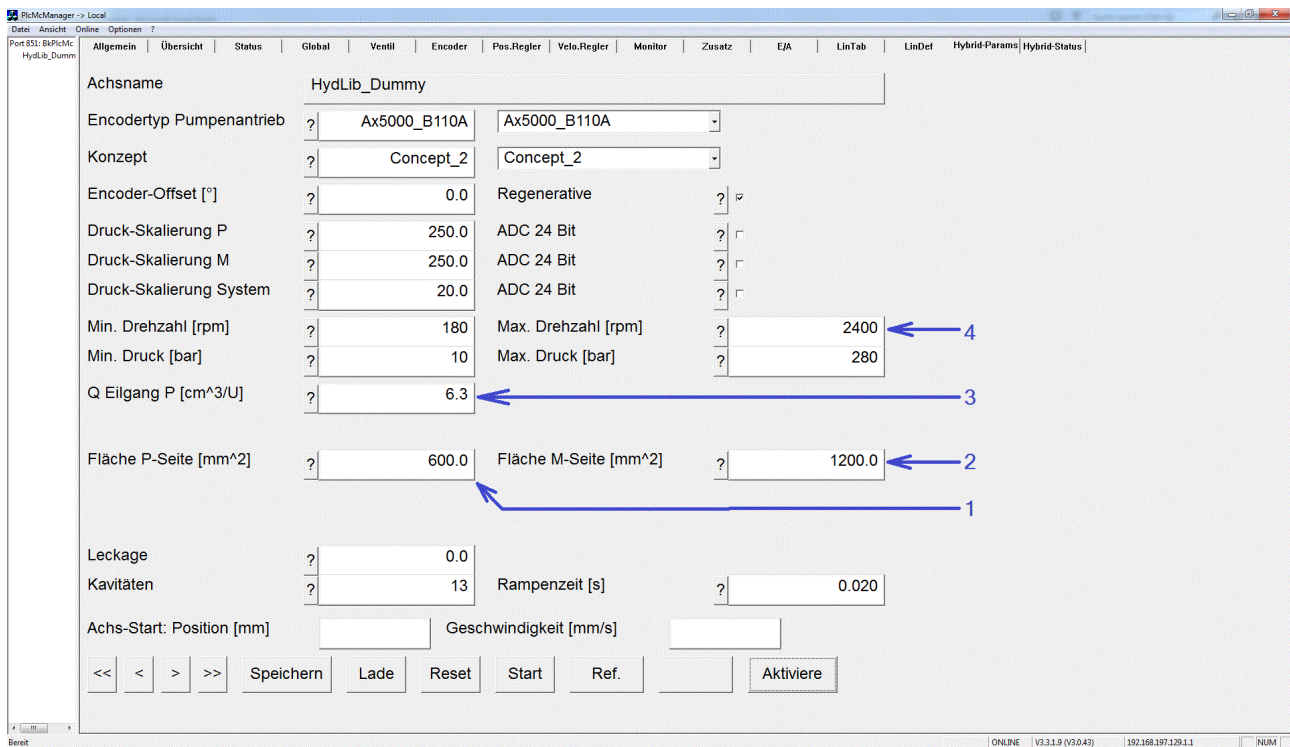
Für die Steuerung ergibt sich das Verhalten eines Gleichgang-Zylinders mit richtungsunabhängiger Vorschub-Konstante.

Hydraulisch liegt ein Differential-Zylinder vor und es ist ein Austauschvolumen zu speichern.

Es ist keine Getriebschaltung verfügbar. Sie kann durch synchron mengenumschaltbare Pumpen realisiert werden. Dies ergibt ein anderes Konzept.



Das Ölvolumen in Q1 sowie das Volumen der nur von ihrem Förderstrom durchströmten Leitungen muss kleiner sein als das im Betrieb ausgetauschte Öl für den Querschnitt der Kolbenstange. Andernfalls ist kein sicherer Ölaustausch gegeben.



Benötigte Parameter: Wirkungsfläche in positiver Richtung (1), Wirkungsfläche in negativer Richtung (2), Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung (3), maximale Pumpendrehzahl (4), maximale Pumpendrehzahl (5).

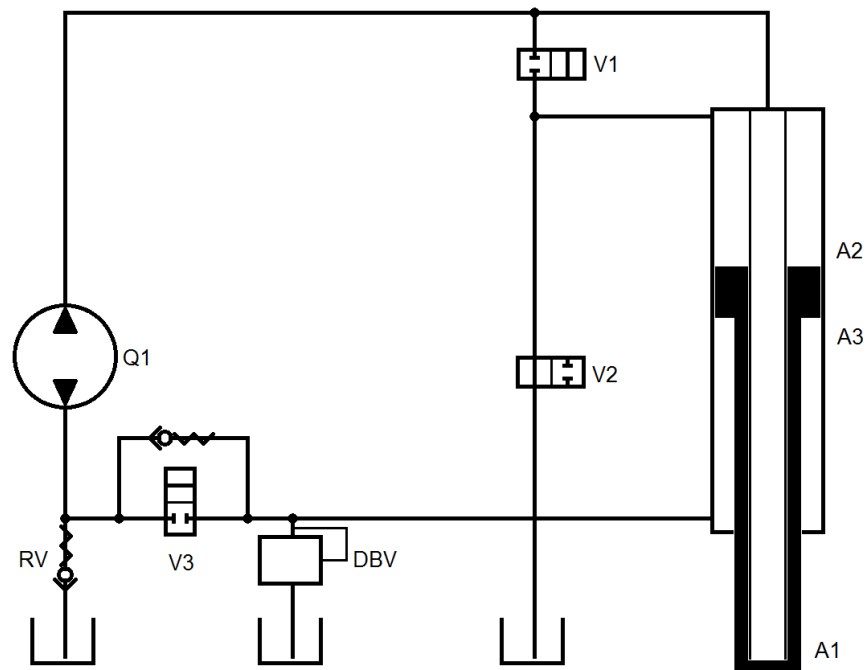
Automatisch berechnete Parameter: Volumen pro Umdrehung an der Wirkungsfläche in negativer Richtung. Dabei wird das Verhältnis der Wirkungsflächen verwendet.

Automatisch festgelegte Parameter: Die zuschaltbaren Flächen sind 0, die Umdrehungs-Volumina für den Kraftgang sind gleich den Werten für den Eilgang. Die Markierung für den Regenerativ-Betrieb ist gesetzt.

### 3: Getriebebeschaltung durch eine Umschaltung von Wirkungsflächen

Hier werden durch Schaltventile Wirkungsflächen eines Zylinders wirksam oder unwirksam gemacht oder variabel zusammengeschaltet. In einigen Fällen entstehen dabei ‚virtuelle‘ Flächen, die bei der Ölmenge zu berücksichtigen sind, aber nicht zum Kraftaufbau beitragen.

## 3a: Einsatz eines Zylinders mit 2+1 Wirkungsflächen



Hersteller / Produkt: EH-D.

Eilgang-Schaltung: V1 und V2 (Achtung: Ventil-Symbol beachten) sind ausgeschaltet. Zum Ausfahren des Zylinders muss V3 geschaltet werden. Wenn A1 und A3 gleich sind wird kein Öl mit dem Tank ausgetauscht. Andernfalls muss richtungsabhängig Differenz-Öl über das RV nachgesaugt oder über das DBV verdrängt werden.

Kraftgang-Schaltung: V1 und V2 (Achtung: Ventil-Symbol beachten) sind eingeschaltet. Beim Ausfahren muss V3 geschaltet werden. Die Ölmenge aus A3 wird für A1/A2 über das RV ergänzt. Ein Einfahren ist in dieser Schaltung nur möglich, indem ein erhebliches Volumen unter Wärmeentwicklung über das DBV verdrängt wird. Diese Kombination aus Ventil-Schaltung und Drehrichtung der Pumpe ist für den Druckabbau sinnvoll, sollte jedoch nicht für eine aktive Bewegung verwendet werden.

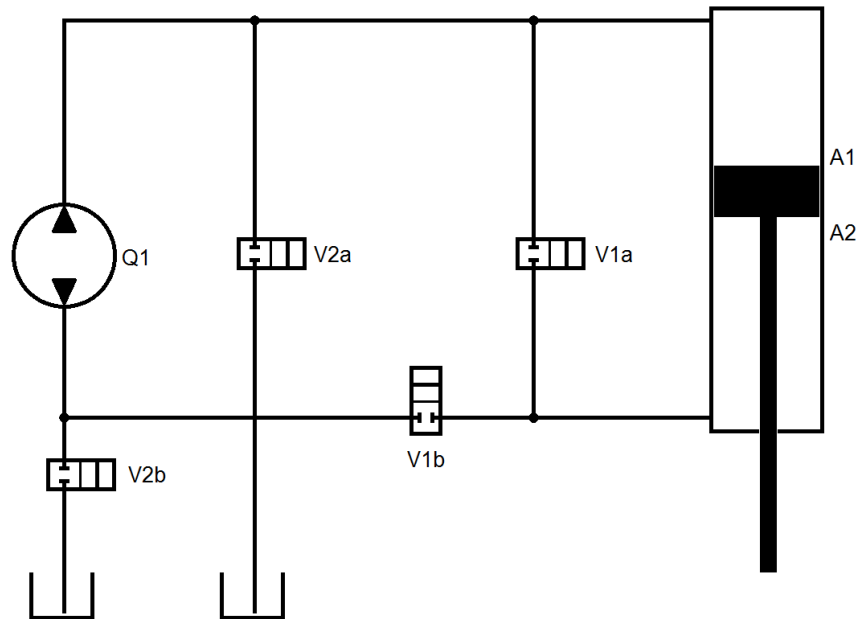
Benötigte Parameter: Wirkungsfläche in Ausfahr-Richtung = A1 (1), addierte Wirkungsfläche in Ausfahr-Richtung = A2 (2), Wirkungsfläche in Einfahr-Richtung = A3 (3), addierte Wirkungsfläche in Einfahr-Richtung = 0 (4), Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung (5), maximale Pumpendrehzahl (6).

Automatisch berechnete Parameter: Das Volumen pro Umdrehung an der Wirkungsfläche in negativer Richtung wird gleich dem Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung gesetzt.

Automatisch festgelegte Parameter: Die Umdrehungs-Volumina für den Kraftgang sind gleich den Werten für den Eilgang.



3b: Erzeugung einer virtuellen Flächenumschaltung



Hersteller / Produkt: Voith Turbo / CLSP.

Hier ergibt sich durch Ventile eine Getriebeschaltung.

Eilgang-Schaltung:

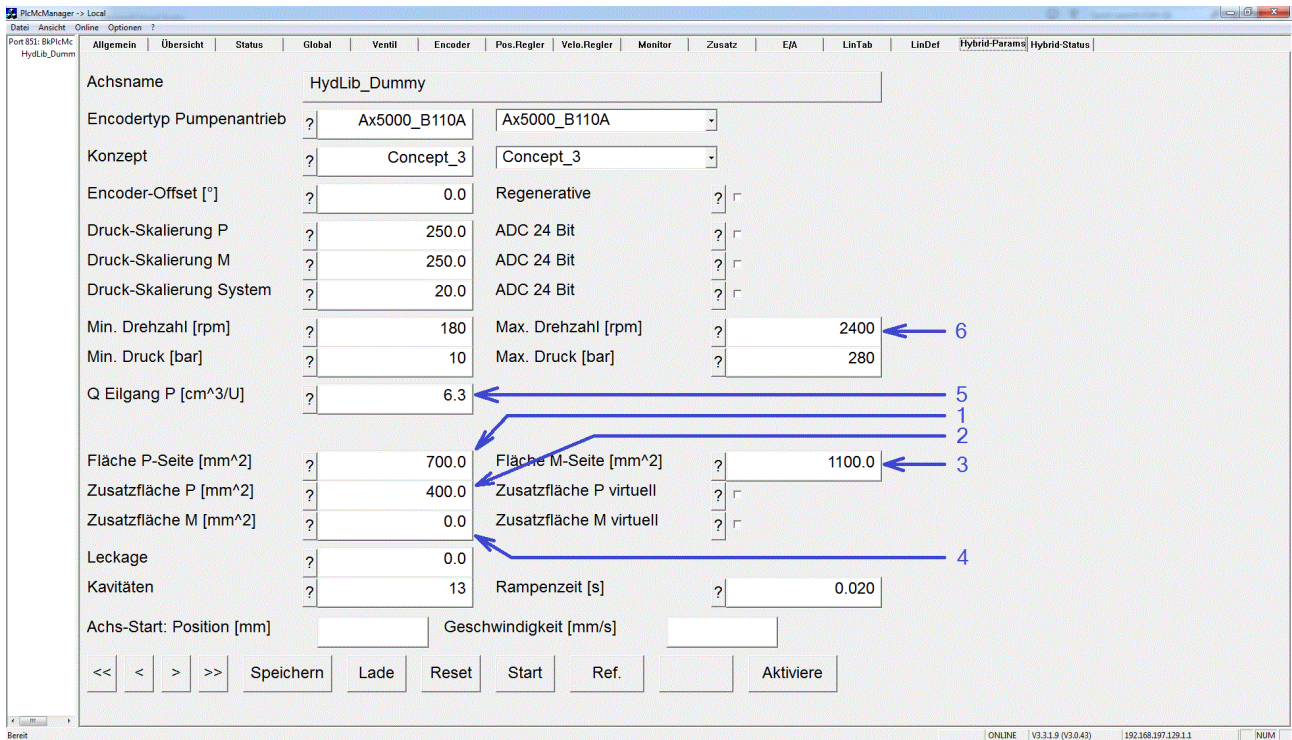
Beim Ausfahren sind V1a und V2b geschaltet. Die Ölmenge für die Ringfläche wird via V1a zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via V2b durch die Pumpe aus dem Tank ergänzt. Schaltungsbedingt ist der Stangenquerschnitt hydraulisch abgestützt. Der Zylinder weist eine niedrige Eigenfrequenz auf und sollte mit angepasster Dynamik betrieben werden.

Beim Einfahren sind V1b und V2a geschaltet. Die Ölmenge für die Ringfläche wird via V1b durch die Pumpe zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via V2a zum Tank abgeleitet. Schaltungsbedingt ist die Fläche A2 nur mit dem Vorspanndruck des Tanks beaufschlagt. Der Zylinder ist nur bedingt in der Lage, durch eigene Kraft zu Bremsen. Er sollte mit angepasster Dynamik betrieben werden.

Kraftgang-Schaltung:

Beim Ausfahren sind V1b und V2b geschaltet. Die Ölmenge für die Ringfläche wird via V1b durch die Pumpe zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via V2b durch die Pumpe aus dem Tank ergänzt. Schaltungsbedingt ist die Fläche A2 nur mit dem Vorspanndruck des Tanks beaufschlagt. Der Zylinder ist nur bedingt in der Lage, durch eigene Kraft zu Bremsen. Er sollte durch den Prozess abgestützt und gebremst werden.

Beim Einfahren sind V1b und V2b geschaltet. Die Ölmenge für die Ringfläche wird via V1b durch die Pumpe zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via V2b durch die Pumpe zum Tank abgeleitet. Schaltungsbedingt ist die Fläche A2 nur mit dem Vorspanndruck des Tanks beaufschlagt. Der Zylinder ist nur bedingt in der Lage, durch eigene Kraft zu Fahren. Es ist zu überprüfen, ob er in der Lage ist, die durch Gravitation und Reibung erzeugten Kräfte zu überwinden. Diese Schaltung sollte nur zum Abbau von in Ausfahrrichtung erzeugten Kräften verwendet werden.

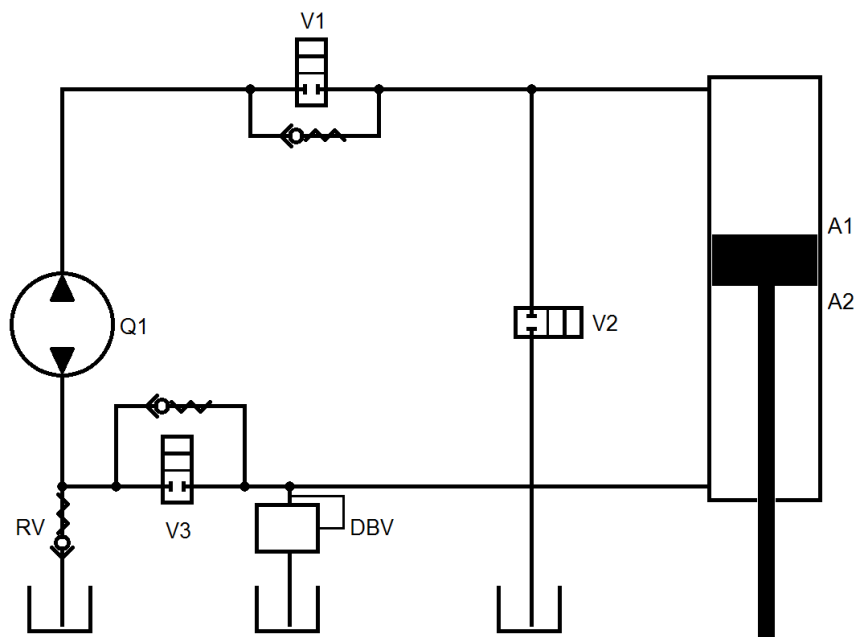


Benötigte Parameter: Wirkungsfläche in Ausfahr-Richtung = Stangenquerschnitt (1), addierte Wirkungsfläche in Ausfahr-Richtung = Ringfläche (2), Wirkungsfläche in Einfahr-Richtung = Ringfläche (3), addierte Wirkungsfläche in Einfahr-Richtung = 0 (4), Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung (5), maximale Pumpendrehzahl (6).

Automatisch berechnete Parameter: Das Volumen pro Umdrehung an der Wirkungsfläche in negativer Richtung wird gleich dem Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung gesetzt.

Automatisch festgelegte Parameter: Die Umdrehungsvolumina für den Kraftgang sind gleich den Werten für den Eilgang.

**3c: Erzeugung einer virtuellen Flächenumschaltung**



Hersteller / Produkt: EH-D / 18-0129-001-HY-K.

Hier ergibt sich eine Getriebebeschaltung durch Ventile.

Eilgang-Schaltung:

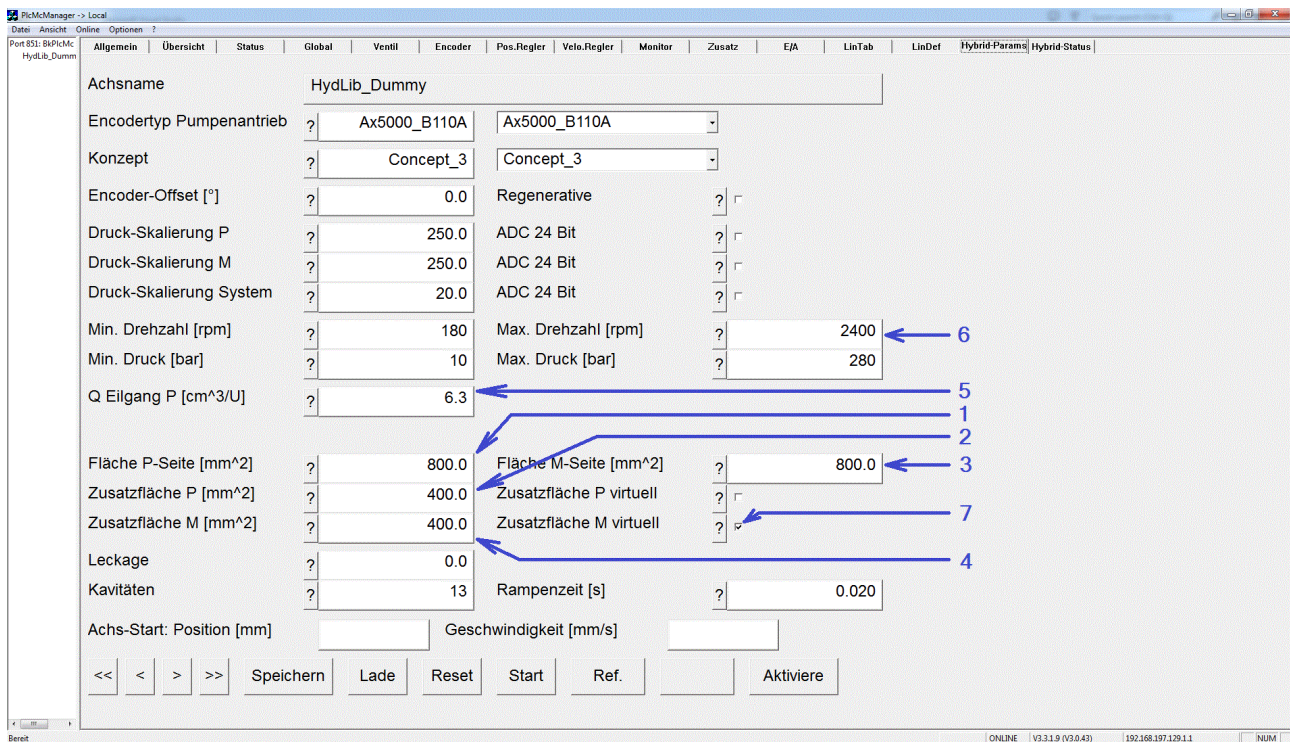
Beim Ausfahren sind V1 optional, V2 und V3 zwingend geschaltet. Die Ölmenge für die Ringfläche wird via V3 / V1 zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via V2 aus dem Tank ergänzt. Schaltungsbedingt ist die Fläche A1 nur mit dem Vorspanndruck des Tanks beaufschlagt. Schaltungsbedingt ist die Fläche A2 nur mit dem Vorspanndruck des Tanks beaufschlagt. Der Zylinder ist nur bedingt in der Lage, durch eigene Kraft zu fahren. Es ist zu überprüfen, ob er in der Lage ist, die durch Gravitation und Reibung erzeugten Kräfte zu überwinden. Diese Schaltung sollte nur zum Abbau von in Ausfahr-Richtung erzeugten Kräften verwendet werden.

Beim Einfahren sind V1 zwingend, V2 zwingend und V3 optional geschaltet. Die Ölmenge für die Ringfläche wird via V1 / V3 durch die Pumpe zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via V2 zum Tank abgeleitet. Schaltungsbedingt ist die Fläche A2 nur mit dem Vorspanndruck des Tanks beaufschlagt. Der Zylinder ist nur bedingt in der Lage, durch eigene Kraft zu Bremsen. Er sollte mit angepasster Dynamik betrieben werden.

Kraftgang-Schaltung:

Beim Ausfahren sind V1 optional und V3 zwingend geschaltet, V2 zwingend nicht geschaltet. Die Ölmenge für die Ringfläche wird via V3 / V1 durch die Pumpe zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via RV durch die Pumpe aus dem Tank ergänzt. Schaltungsbedingt ist die Fläche A2 nur mit dem Vorspanndruck des Tanks beaufschlagt. Der Zylinder ist nur bedingt in der Lage, durch eigene Kraft zu Bremsen. Er sollte durch den Prozess abgestützt und gebremst werden.

Beim Einfahren sind V1 zwingend und V3 optional geschaltet, V2 zwingend nicht geschaltet. Die Ölmenge für die Ringfläche wird via V1 / V3 durch die Pumpe zwischen den Flächen ausgetauscht. Die Ölmenge für den Stangenquerschnitt wird via DBV zum Tank abgeleitet. Schaltungsbedingt ist die Fläche A2 nur mit dem Begrenzungsdruck des DBV beaufschlagt. Der Zylinder ist nur bedingt in der Lage, durch eigene Kraft zu fahren. Es ist zu überprüfen, ob er in der Lage ist, die durch Gravitation und Reibung erzeugten Kräfte zu überwinden.

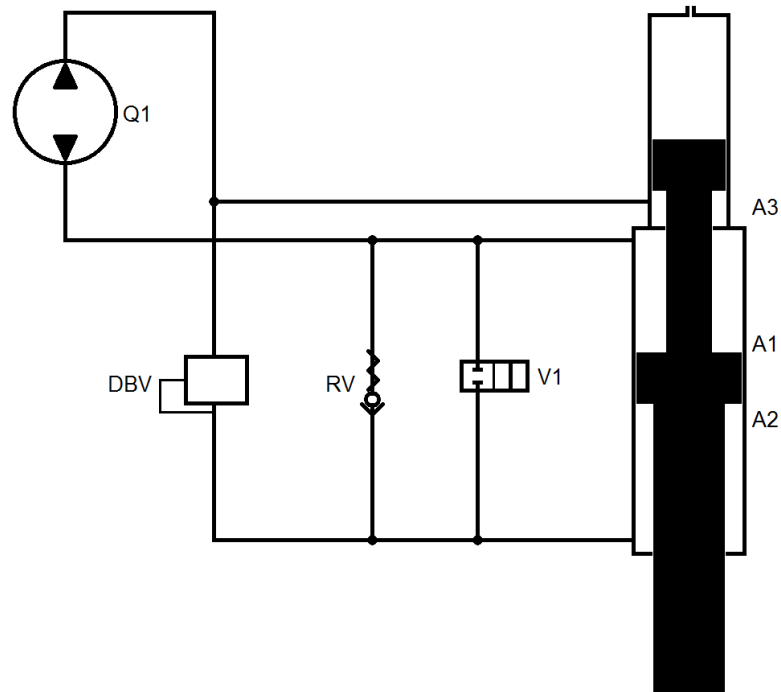


Benötigte Parameter: Wirkungsfläche in Einfahr- und Ausfahr-Richtung = Ringfläche (1, 3), addierte Wirkungsfläche in Einfahr- und Ausfahr-Richtung = Stangenquerschnitt (2, 4), Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung (5). Die addierte Wirkungsfläche in Ausfahr-Richtung ist als ,virtuell' zu kennzeichnen (7), da sie bei der Berechnung der benötigten Drehzahl berücksichtigt werden muss, aber nicht zum Kraftaufbau beiträgt, maximale Pumpendrehzahl (6).

Automatisch berechnete Parameter: Das Volumen pro Umdrehung an der Wirkungsfläche in negativer Richtung wird gleich dem Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung gesetzt.

Automatisch festgelegte Parameter: Die Umdrehungs-Volumina für den Kraftgang sind gleich den Werten für den Eilgang.

### 3d: Erzeugung einer virtuellen Flächenumschaltung



Hersteller / Produkt: Voith Turbo / PDSC.

Hier ergibt sich eine Getriebeschaltung durch Ventile.

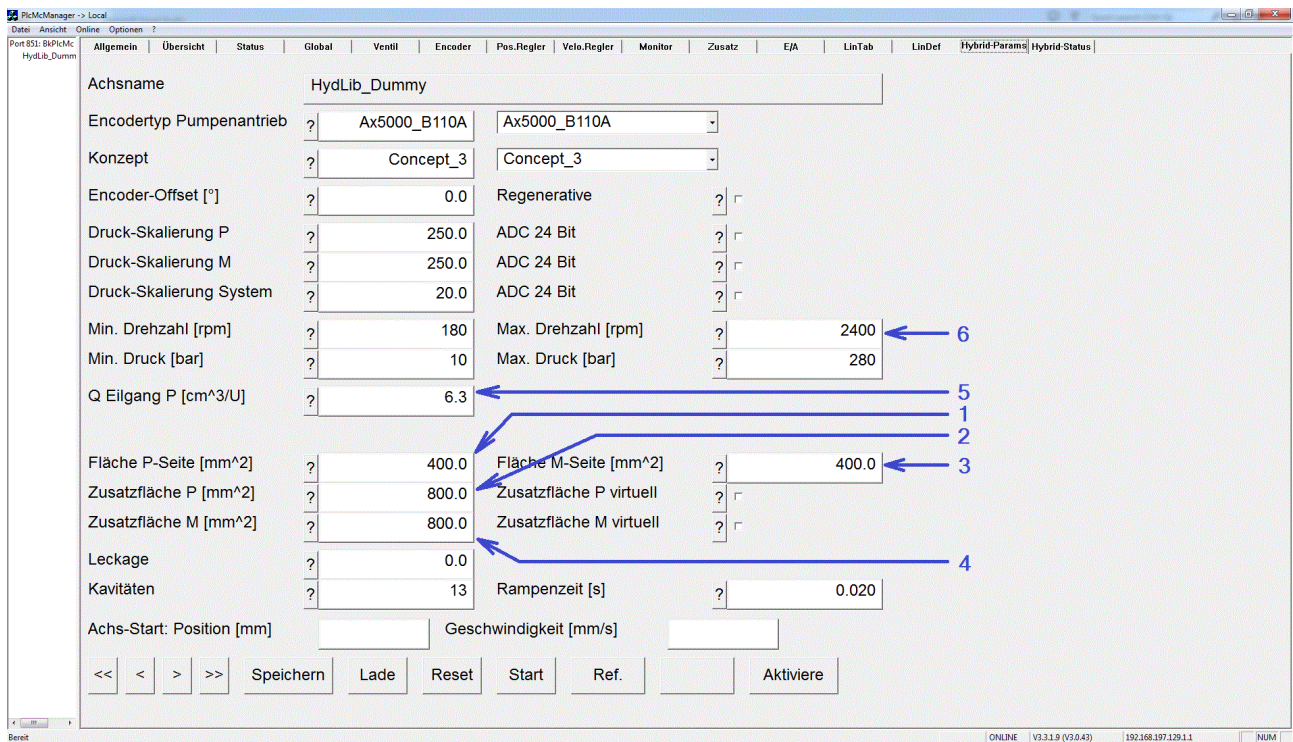
Eilgang-Schaltung:

Beim Ein- und Ausfahren ist V1 zwingend geschaltet. Ein Teil der Ölmenge aus A2 wird regenerativ für A1 genutzt, die Restmenge via Pumpe mit A3 ausgetauscht. Da  $A3=A1-A2$  gilt liegt sowohl hydraulisch als auch für die Steuerung ein Gleichgang-Verhalten vor.

Kraftgang-Schaltung:

Beim Ausfahren ist V1 zwingend nicht geschaltet. Da der Druck in A1 durch die Wirkung der Pumpe höher als die Drücke in A2 und A3 ist sperrt das RV und das DBV verbindet A2 und A3. Da  $A1=A2+A3$  gilt liegt sowohl hydraulisch als auch für die Steuerung ein Gleichgang-Verhalten vor.

Ein Einfahren im Kraftgang ist nicht möglich, da das RV hydraulisch gesteuert die Eilgang-Schaltung erzeugt.

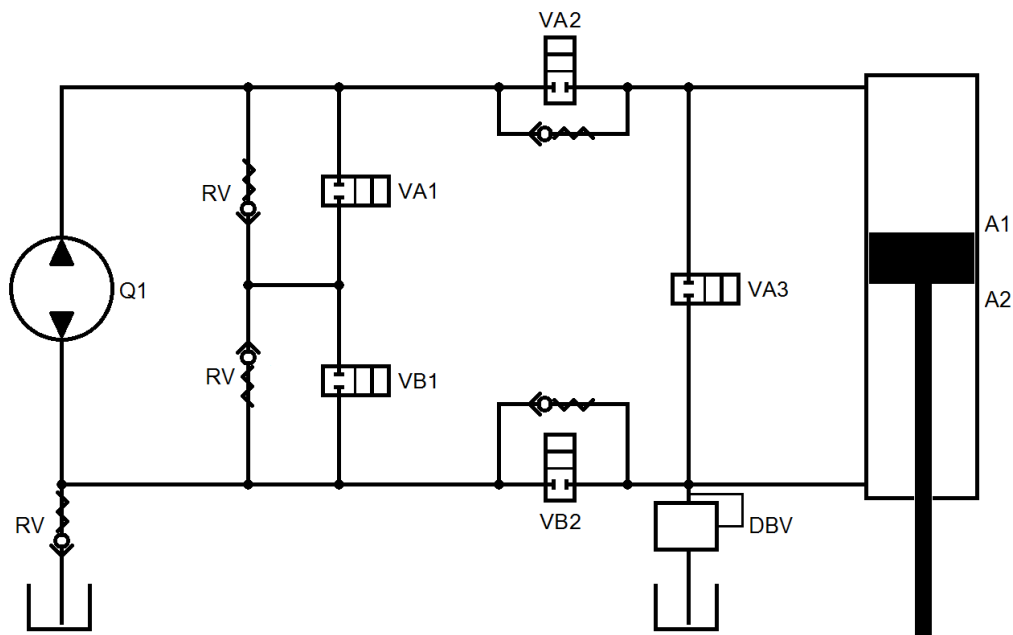


Benötigte Parameter: Wirkungsfläche in Einfahr- und Ausfahr-Richtung = A3 (1, 3), addierte Wirkungsfläche in Einfahr- und Ausfahr-Richtung = A2 (2, 4), Volumen pro Umdrehung an A1 (5), maximale Pumpendrehzahl (6).

Automatisch berechnete Parameter: Das Volumen pro Umdrehung an A2+A3 wird gleich dem Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung gesetzt.

Automatisch festgelegte Parameter: Die Umdrehungs-Volumina für den Kraftgang sind gleich den Werten für den Eilgang.

### 3e: Erzeugung einer virtuellen Flächenumschaltung



Hersteller / Produkt: EH-D / ECO.

Hier ergibt sich eine Getriebebeschaltung durch Ventile.

Eilgang-Schaltung:

Beim Einfahren sind VA1, VA2 zwingend und VB2 optional geschaltet. VB1 und VA3 sind zwingend nicht geschaltet. Die Pumpe fördert Öl von A1 nach A2, wobei der von A2 stammende Anteil für den Stangenquerschnitt via VA1 zum Tank abgeleitet wird.

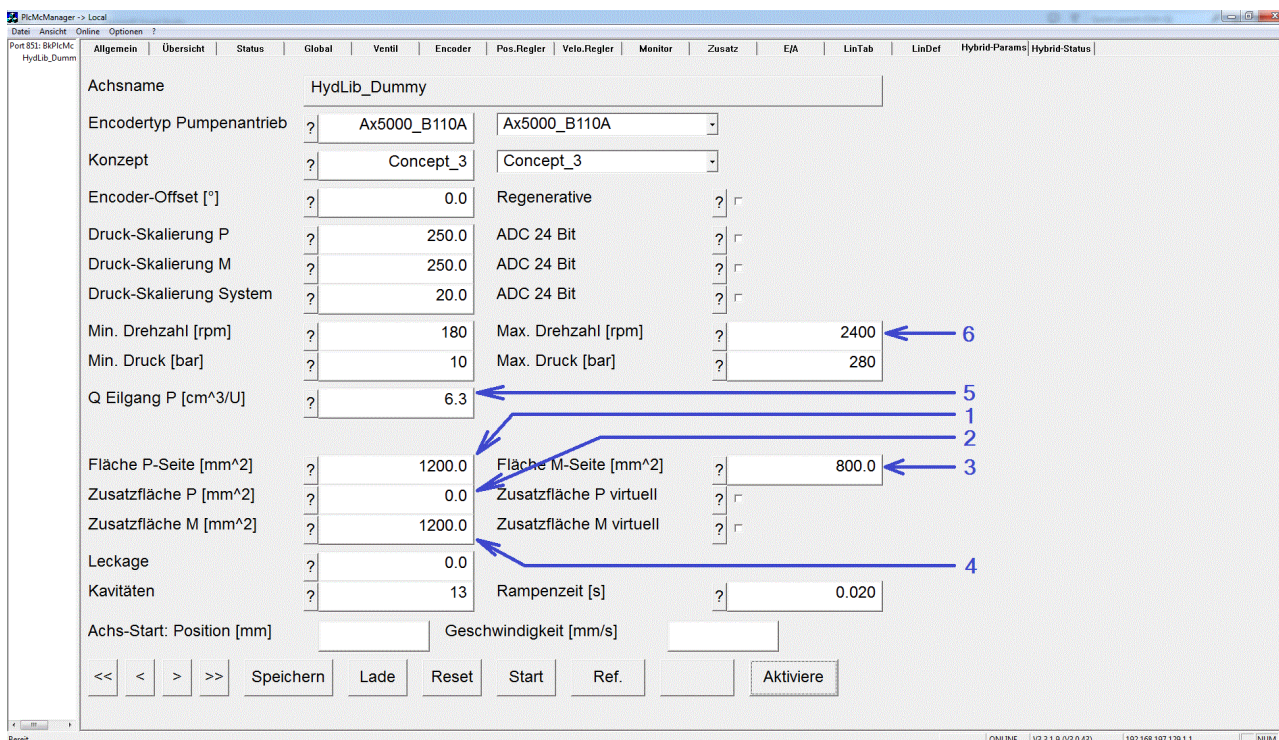
Beim Ausfahren sind VA1 und VB2 zwingend nicht geschaltet. VB1 und VA2 sind optional geschaltet. VA3 ist zwingend geschaltet. Die Ölmenge aus A2 wird via VA3 regenerativ für A1 genutzt, die Restmenge via Pumpe und VB1 vom Tank ergänzt.

Kraftgang-Schaltung:

Beim Ausfahren sind VA1 und VA3 zwingend nicht geschaltet. VB1 und VA2 sind optional geschaltet. VB2 ist zwingend geschaltet. Die Ölmenge aus A2 wird via VB2 regenerativ für A1 genutzt, die Restmenge via Pumpe und VB1 vom Tank ergänzt.

Ein Einfahren im Kraftgang ist nicht möglich, da hierfür das DBV die Ölmenge für den Stangenquerschnitt bei hohem Druck ableiten müsste.

Für eine Dekompression der Fläche A1 kann in der Kraftgangschaltung bei dann zwingend geschalteten VB1 und VA2 der Druck abgebaut werden.



Benötigte Parameter: Wirkungsfläche in Einfahr-Richtung = Ringfläche (1), addierte Wirkungsfläche in Einfahr-Richtung = 0 (2), Wirkungsfläche in Ausfahr-Richtung = Stangenquerschnitt (3), addierte Wirkungsfläche in Ausfahr-Richtung = Ringfläche (4), Volumen pro Umdrehung an A1 (5), maximale Pumpendrehzahl (6).

Automatisch berechnete Parameter: Das Volumen pro Umdrehung an A2 wird gleich dem Volumen pro Umdrehung an der Wirkungsfläche in positiver Richtung gesetzt.

Automatisch festgelegte Parameter: Die Umdrehungs-Volumina für den Kraftgang sind gleich den Werten für den Eilgang.

## 5.5 Bau einer Achse

Anders als bei der Beckhoff NC wird in der Hydraulik-Bibliothek die Achse von der Applikation selbst gebaut. Das heißt, die Bausteine zum Betrieb einer Achse (Istwert lesen, Sollwerte generieren, Lageregeln, Linearisierung und Ausgabe erzeugen) müssen einzeln aufgerufen werden.

Alle Bausteine arbeiten auf einer gemeinsamen Achsreferenz, welche global anzulegen ist. Gibt es mehr als eine Achse, so sind die Achsreferenzen als Array anzulegen.

Neben der Achsreferenz ([AXIS\\_REF BkPlcMc](#) [▶ 84]) müssen pro Achse die E/A Strukturen [ST\\_TcPlcDeviceInput](#) [▶ 138] und [ST\\_TcPlcDeviceOutput](#) [▶ 140] deklariert werden. Abhängig vom Anwendungsfall kommen weitere optionale Elemente hinzu.

Um Meldungen einsehen zu können sollte ein [ST\\_TcPlcMcLogBuffer](#) [▶ 142] deklariert werden. Dieser Puffer wird von allen Achsen gemeinsam benutzt.

Sind in der Applikation neben der reinen Positionserfassung weitere Sensoren wie Druck- oder Kraftmessdosen vorhanden, so muss der E/A-Wert in der Applikation angelegt werden. Die Parametrierung der Skalierung kann im Bereich `fCustomerData` der Achse verwaltet werden. In jeder Achse sind dort 20 kundenspezifische Daten bereitgestellt. Diese Daten werden über die Achse gespeichert, geladen und im `PLcMcManager` angezeigt. Für die Anzeige im `PLcMcManager` kann die Beschriftung über die Deklaration der Struktur [ST\\_TcMcAuxDataLabels](#) [▶ 137] verändert werden.

### VAR\_GLOBAL

```

AxisRef:           AXIS_REF_BkPlcMc;           (* mandatory *)
DevIn AT %I*:      ST_TcPlcDeviceInput;        (* mandatory *)
DevOut AT %Q*:     ST_TcPlcDeviceOutput;       (* mandatory *)

stLogBuffer:      ST_TcPlcMcLogBuffer;         (* strongly recommended *)

AuxAxLabel:       ST_TcMcAuxDataLabels;        (* optional *)

AutoIdent:        ST_TcMcAutoIdent;           (* if required *)

CmdBuffer:        ST_TcPlcCmdBuffer_BkPlcMc;  (* if required *)

ForceInput AT %I*: INT;                       (* if required *)

```

### END\_VAR

Beispiel für die Daten einer Achse

### Generelle Einstellungen

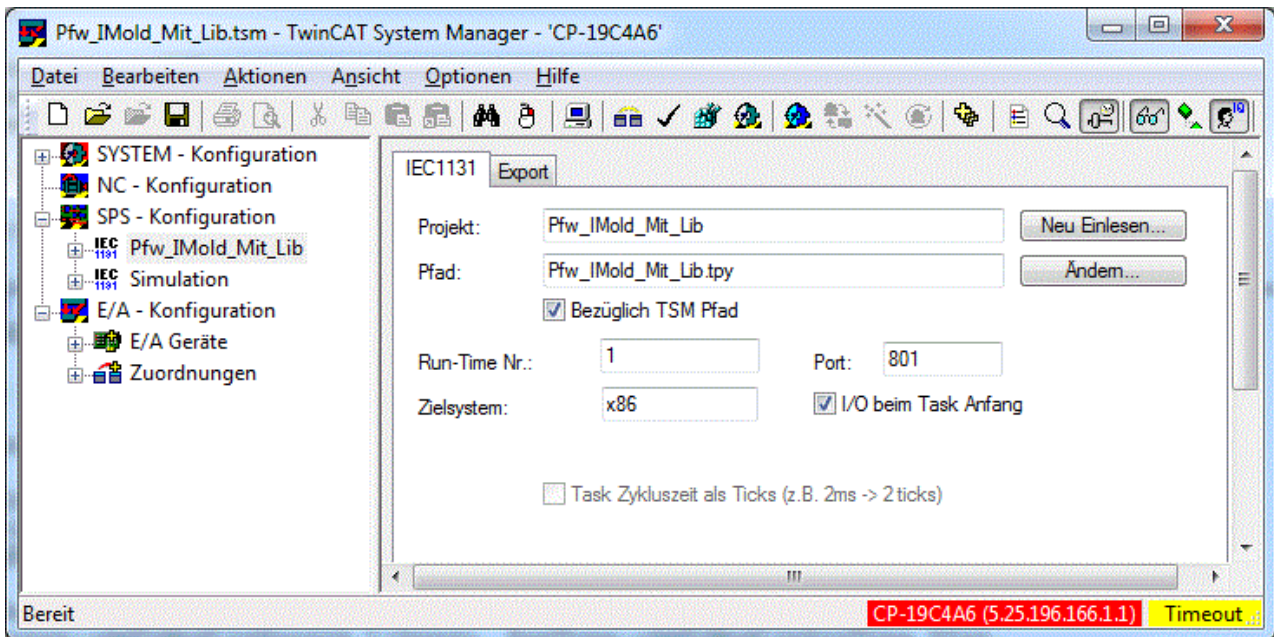
Damit das E/A immer unabhängig vom Zeitbedarf des Programms mit konstantem Zeitabstand eingelesen wird, muss in TwinCAT 3 im Programm ein Attribut gesetzt werden.

```

1  {attribute 'TcCallAfterOutputUpdate'}
2  PROGRAM MAIN
3  VAR
4      fbAxis:      FB_Axis;
5  END_VAR
6

```

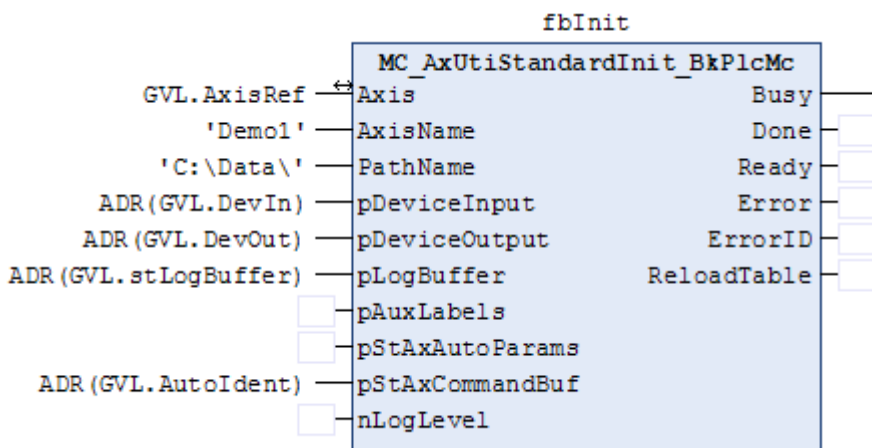
Unter TwinCAT 2 muss im System Manager unter **SPS - Konfiguration** das Flag **I/O am Task Anfang** gesetzt werden.



Anders als bei der NC wird die Hydraulikachse selbst (Sollwertgenerator, Regler usw. ) direkt in der PLC gerechnet. Damit ist zu empfehlen, die Zykluszeit der Task auf <10 ms zu stellen.

**Initialisierung**

Der PLCopen-Standard legt fest, dass alle Motion-Bausteine der Applikation mit einer Instanz des Typs `AXIS_REF_BkPlcMc` aufgerufen werden. Einige Achsbestandteile können aus technischen Gründen nicht in einer solchen Instanz enthalten sein, da sie in gesonderten Bereichen (z.B. Prozessabbilder) liegen müssen. Andere Elemente sind optional und werden nur bei Bedarf ergänzt. Um sie mit der Achs-Referenz zu verbinden werden sie einem Initialisierungsbaustein vom Typ `MC_AxUtiStandardInit_BkPlcMc` [ 238 ] übergeben.



Beim erstmaligen Aufruf verknüpft der Baustein die Eingangs- und Ausgangsstrukturen sowie alle optionalen Elemente mit der Achsreferenz. Variablen, die als Adresse übergeben werden müssen, sind mit dem Präfix „p“ gekennzeichnet. Der Baustein sollte weiter zyklisch aufgerufen werden, um die Pointeradressen zu kontrollieren.

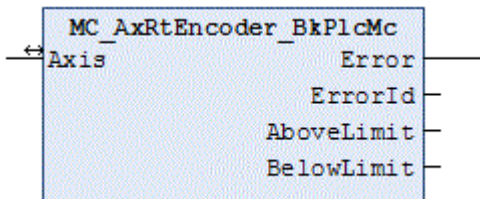
**i** Es ist nicht zulässig, eine Instanz von `ST_TcPlcDeviceInput`, `ST_TcPlcDeviceOutput` oder `ST_TcMcAutoUdent` an mehrere Achsen zu binden. Es ist nicht zulässig, mehr als eine Instanz von `ST_TcMcLogBuffer` mit Achsen zu verbinden.

Der Baustein lädt die Parameter aus dem übergebenen Dateipfad und übergibt diese an die Achsreferenz. Sämtliche Parameter sind binär in einer Datei *Achsname.dat* abgespeichert.



Wurde das Laden der Parameter erfolgreich abgeschlossen, wird das Flag bParamsEnable in der Achsreferenz TRUE. Erst jetzt ist die Verwendung noch nicht definierter Parameter ausgeschlossen und es dürfen alle weiteren achsbezogenen Bausteine aufgerufen.

**Istwert-Erfassung**



Der in der Parameter-Struktur der Achs-Referenz eingestellte Encodertyp legt fest, wie und aus welchen Variablen der Eingangsstruktur der [MC\\_AxRtEncoder\\_BkPlcMc](#) [▶ 183] Baustein den Istwert lesen und in eine Position [mm] und eine Geschwindigkeit [mm/s] umrechnen wird. Dabei wird bei Verwendung von EtherCAT-Komponenten die Verbindung überwacht.

Sollten die Aktualwerte stark verrauscht sein, besteht die Möglichkeit, diese über einen gleitenden Mittelwert ([MC\\_AxUtiSlidingAverage\\_BkPlcMc](#) [▶ 253]) oder ein Pt1-Glied ([MC\\_AxUtiPT1\\_BkPlcMc](#) [▶ 250]) zu filtern. Die Verwendung eigener Filter ist möglich.

Filterbausteine sind nach dem Encoder-Baustein aufzurufen. Ihrem Eingang muss die zu filternde Variable übergeben werden. Das Ergebnis kann auf die entsprechende Variable der Achsreferenz zurück geschrieben werden. Dadurch ist der alte verrauschte Wert durch einen neuen beruhigten Wert ersetzt.



Wird ein stark gefilterter Aktualwert für eine Regelung verwendet kann aufgrund der Filtersprungantwort die Dynamik und die Regelbarkeit beeinträchtigt sein.

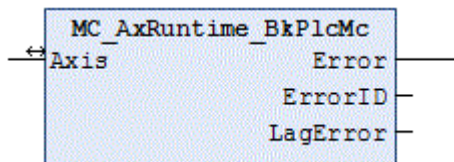
Zum Einlesen von Drücken und Kräften stehen weitere Bausteine zur Verfügung. In Abhängigkeit von der zu erfassenden Messgröße muss der entsprechende Baustein verwendet werden. Anders als bei der Positionsermittlung muss für die Kraft- und Druckermittlung die Mapping-Schnittstelle sowie die Klemmenüberwachung von der Applikation bereitgestellt werden.



Für die Ermittlung der wesentlichen Istwerte einer servoelektrisch-hydraulischen Hybrid-Achse steht mit [MC\\_AxRtHybridAxisActuals\\_BkPlcMc](#) [▶ 209] ein angepasster Baustein zur Verfügung.

**Sollwert-Generierung und Default-Lage-Regler**

Wird durch z.B. [MC\\_MoveAbsolute\\_BkPlcMc](#) [▶ 72] eine aktive Bewegung der Achse ausgelöst berechnet der Sollwert-Generator in jedem Zyklus die aktuellen Werte für Soll-Geschwindigkeit und Soll-Position. Dies kann zeit- oder weggesteuert erfolgen. Bei einer zeitgesteuerten Generierung ist eine permanente Lage-Regelung erforderlich, andernfalls wird dies nur im Stillstand benötigt. Es werden mehrere Profil-Varianten unterstützt. Näheres hierzu wird in der Dokumentation des Bausteins bereitgestellt.



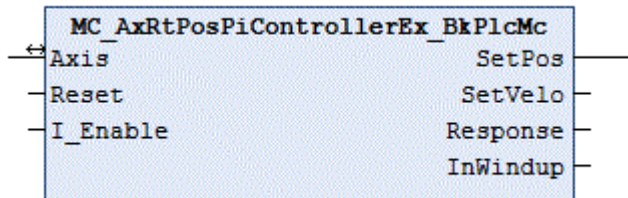
Wenn die Achse nicht mit einem Kommando-Puffer ausgerüstet ist wird ein Befehl direkt in den Laufzeitdaten der Achse eingetragen. Andernfalls werden Kommandos gepuffert, einer Wegplanung unterzogen und dann entsprechend den Blending-Regeln wirksam gemacht.

Bei Bedarf kann die Applikation die Sollwert-Generierung übernehmen. Dazu ist ein [MC\\_AxRtSetExtGenValues\\_BkPlcMc](#) [▶ 236] Baustein einzusetzen. Der weiter aufzurufende Bibliotheksbaustein wird bei aktiver externer Generierung in einen passiven Zustand geschaltet und anschließend reaktiviert. Auf diese Weise können applikationsspezifische Getriebe und andere nicht standardisierbare Mechanismen verwirklicht werden.

Der Sollwert-Generator und ein in den meisten Fällen ausreichender Default-Lage-Regler sind im MC\_AxRuntime\_BkPlcMc [► 221] Baustein integriert.

### Alternativer Lage-Regler

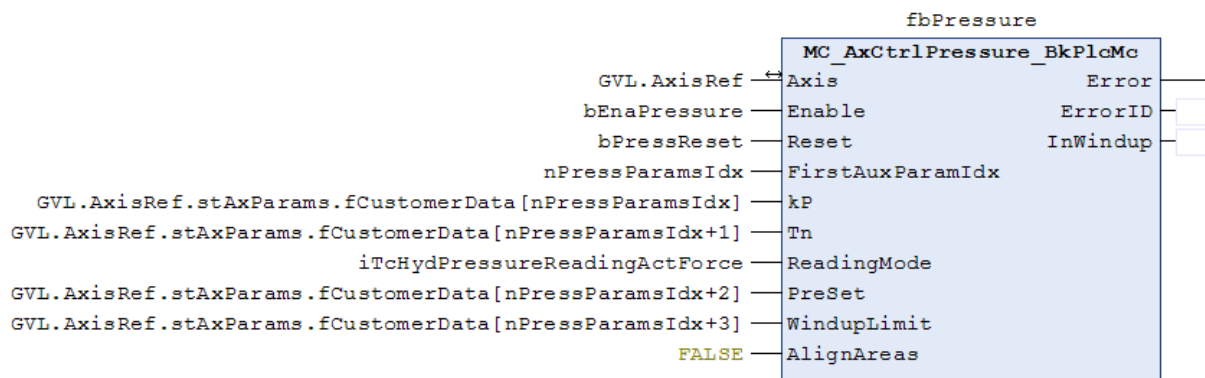
Wenn nach dem Default-Lage-Regler ein anderer Regler aufgerufen wird und fLagCtrlOutput in den Laufzeitdaten der Achsreferenz überschrieben wird kann dadurch ein anderer Lage-Regler wirksam gemacht werden. Dabei kann es sich um einen kundenspezifischen Regler oder einen anderen Regler der Bibliothek wie zum Beispiel den FB MC\_AxRtPosPiControllerEx\_BkPlcMc [► 169] handeln.



Bei diesem Bibliotheks-Regler handelt es sich um einen PID Regler mit optionalen Erweiterungen wie Zustandsrückführung und Beschleunigungsvorsteuerung.

### Weitere Regler

In vielen Anwendungen mit hydraulischen Achsen werden Druck- oder Kraftregler eingesetzt. Als Beispiel wird hier ein MC\_AxCtrlPressure\_BkPlcMc [► 156] Baustein gezeigt.



Im aktiven Zustand überschreibt der Baustein die Ausgabe des Sollwert-Generators. Damit die Reaktion des Reglers wirksam wird muss er vor der Linearisierung aufgerufen werden.



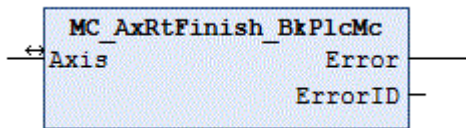
Bei der Aktivierung oder Deaktivierung kann es abhängig von Parameter-Werten zu Sprüngen in den Kontroll-Werten der Achse kommen.

### Abschließende Bearbeitung

An diesem Punkt liegen die Kontroll-Werte der Achse in einer Form vor, die ein lineares Verhalten der Achse und ihrer Komponenten voraussetzt. Dies ist in der Praxis so nur in den seltensten Fällen gegeben. Um dies zu berücksichtigen werden die Kontroll-Werte (Sollwerte, Reglerausgaben, Überdeckungskompensation) zu einem Ausgabewert zusammengefasst und einer Linearisierung unterzogen. Diese Anpassung kann abschnittsweise oder kennlinienbasiert erfolgen.

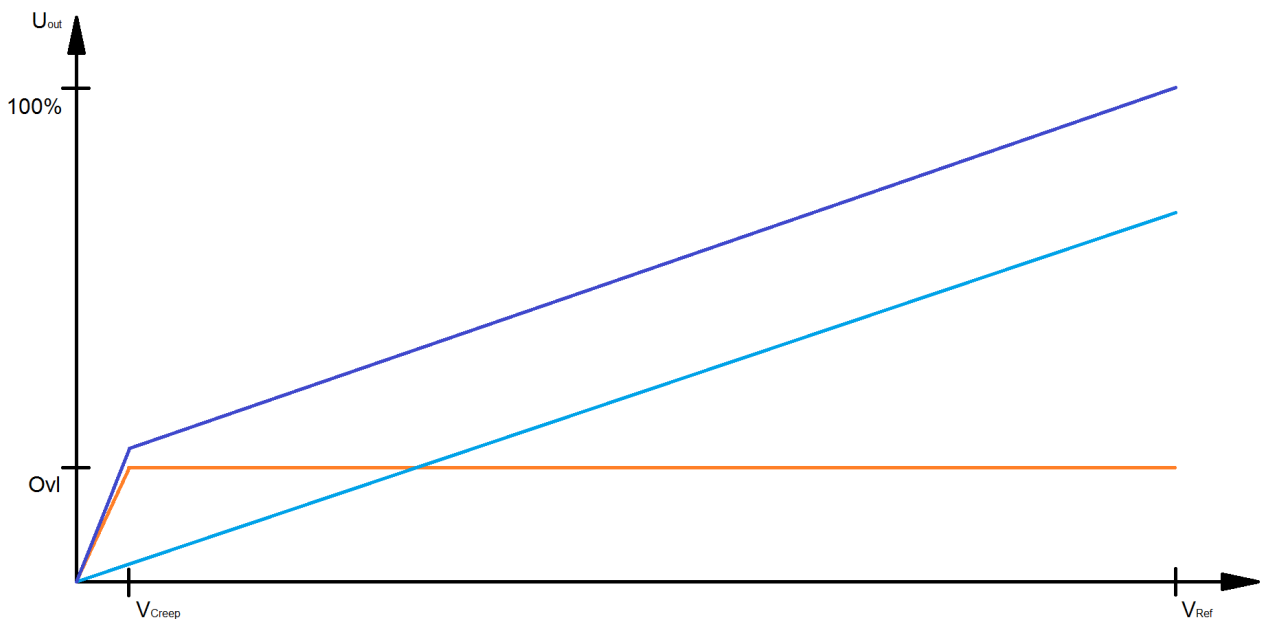
### Abschnittsweise Linearisierung

Für eine einfache Linearisierung stellt die Bibliothek den Baustein MC\_AxRtFinish\_BkPlcMc [► 230] bereit.



Die mit der Vorsteuerung gewichtete Sollgeschwindigkeit und die Reglerausgabe werden zur Ausgabegeschwindigkeit addiert.

Eine aktive Überdeckungskompensation wird so gewählt, dass sie zwischen 0 und  $V_{Creep}$  linear von Null auf die eingestellte Überdeckungskompensation  $Ovl$  geramppt wird. Für den übrigen Bereich ist sie voll wirksam.

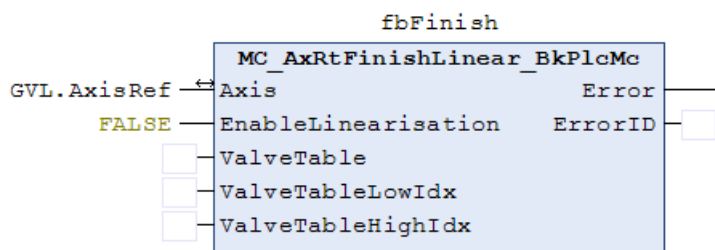


Die Richtungsabhängigkeit wird kompensiert. Dazu wird die Ausgabegeschwindigkeit mit  $fAreaRatio$  aus den Achsparametern multipliziert, wenn die Geschwindigkeit positiv und  $fAreaRatio \geq 1.0$  ist. Wenn die Geschwindigkeit negativ und  $fAreaRatio \leq 1.0$  ist wird sie dividiert.

Die Ausgabe wird durch Addition der gewichteten Sollgeschwindigkeit, der Reglerausgabe, der aktiven Überdeckungskompensation und der Offset-Korrektur gebildet.

### Kennlinienbasierte Linearisierung

Für diese höher auflösende Linearisierung stellt die Bibliothek den Baustein MC\_AxRtFinishLinear\_BkPlcMc [► 231] bereit.

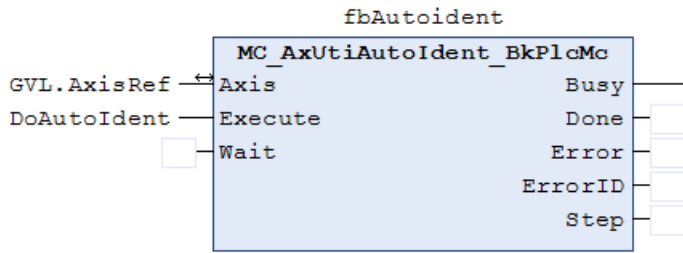


Sollte die Verwendung der Kennlinie nicht freigegeben oder nicht möglich sein wird ein interner Baustein für die abschnittsweise Linearisierung genutzt. Dies ist der Fall wenn mindestens einer der folgenden Gründe vorliegt:

- Am Enable-Eingang des Bausteins wird FALSE übergeben.
- Es ist keine Instanz vom Typ ST\_TcMcAutolident mit der Achsreferenz verbunden worden.
- $bLinTabAvailable$  in den Parametern der Achse ist FALSE: Die Kennlinie ist nicht gültig.

Andernfalls werden die mit der Vorsteuerung gewichtete Sollgeschwindigkeit und die Reglerausgabe zur Ausgabegeschwindigkeit addiert. Die zwei dem errechneten Wert am nächsten liegenden Kennlinienpunkte werden ermittelt und der Ausgabewert durch Zwischeninterpolation und Addition der Offset-Korrektur gebildet.

**Kennlinien-Vermessung**



Der Baustein [MC\\_AxUtiAutoIdent\\_BkPlcMc](#) [▶ 254] unterstützt die Vermessung einer Kennlinie durch einen standardisierten automatischen Ablauf. Die hierfür einzustellenden Parameter sind in der Struktur [ST\\_TcMcAutoIdent](#) [▶ 118] hinterlegt. Soll eine Kennlinien-Vermessung und eine kennlinienbasierte Linearisierung genutzt werden ist ein solches Element anzulegen und mit der Achsreferenz zu verbinden.

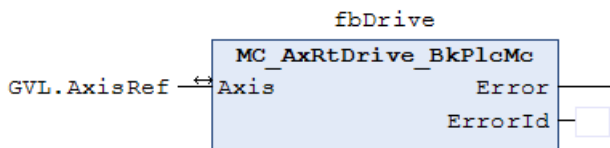
**i** Ein [MC\\_AxUtiAutoIdent\\_BkPlcMc](#) Baustein muss nach dem [MC\\_AxRtFinishLinear\\_BkPlcMc](#) [▶ 231] und vor dem [MC\\_AxRtDrive\\_BkPlcMc](#) [▶ 172] Baustein der Achse aufgerufen werden.

Die ermittelte Kennlinie vereint die Einflüsse einer Reihe von Quellen:

- Nichtlinearitäten des Ventils
- Asymmetrie des Zylinders
- Strömungseffekte bei höheren Geschwindigkeiten
- Mögliche Limitierungen durch eine Pumpe
- Lagebedingte Einwirkungen wie Gravitation
- Einflüsse weiterer Komponenten im Ölstrom

**i** Bei einer servoelektrisch-hydraulischen Hybridachse darf kein [MC\\_AxUtiAutoIdent\\_BkPlcMc](#) Baustein aktiviert werden.

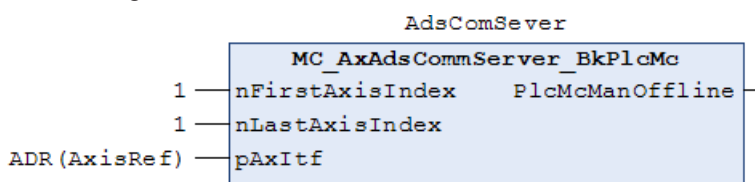
**Ausgabe-Anpassung**



An diesem Punkt liegen die Kontroll-Werte für die Achse als physikalische oder normierte Größen vor. Erst der [MC\\_AxRtDrive\\_BkPlcMc](#) [▶ 172] Baustein ermittelt eine Ausgabe-Größe, die diese Größen in einer Form darstellt, die vom eingesetzten Gerät in die gewünschte Reaktion umgesetzt wird. Die dabei verwendete Methode und ihre Parameter sind in der Parameter-Struktur der Achs-Referenz eingestellt.

**Anbindung des PlcMcManagers**

In Bearbeitung.

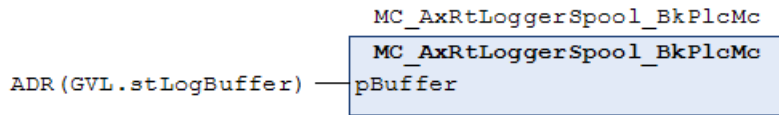


Die Anbindung des PlcMcManagers erfolgt über den ADS-Dienst von TwinCAT. Da dieser nur ein Port pro Applikation erlaubt müssen alle Achsen eine gemeinsame Anbindung nutzen. Mehrere Instanzen dieses Bausteins sind nicht zulässig.

Das gezeigte Beispiel gilt für eine Applikation mit nur einer Achse. Mehrachsige Projekte müssen die Achsreferenzen in einem Array zusammenfassen, dessen Adresse und erster und letzter Index übergeben werden.

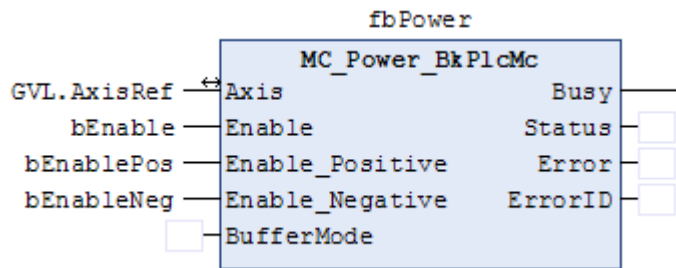
Der Aufruf dieses FBs hat unabhängig davon zu erfolgen, ob Achsen ihre Parameter laden können.

**Message-Logging**



Alle Achsen einer Applikation teilen sich einen Logging Buffer. Um die dort auflaufenden Meldungen in das Event Log des Betriebssystems und falls verfügbar in das Meldungsfenster der Entwicklungsumgebung zu senden ist pro Applikation eine Instanz des Bausteins MC\_AxRtLoggerSpool\_BkPlcMc [► 242] anzulegen. Der Aufruf des Bausteins ist unabhängig davon, ob Achsen ihre Parameter laden können.

**5.5.1 FB\_Power**



Der Baustein verwaltet die Freigaben der Achse. Dabei wird zwischen einer Reglerfreigabe (Enable) beziehungsweise der richtungsabhängigen Vorschubfreigabe in positiver und negativer Richtung unterschieden. Die Vorschubfreigabe ist eine interne Freigabe für den Sollwertgenerator, hingegen ist die Reglerfreigabe für den Lageregler, aber auch für die Endstufe von Antrieben.

**5.6 Der PlcMcManager**

Der PlcMcManager unterstützt die Inbetriebnahme und den Test von Achsen, die mit Hilfe der Hydraulik-Bibliothek automatisiert werden. Er visualisiert den Ist-Zustand, ermöglicht den Zugriff auf Parameter und das Auslösen von Kommandos.



Der PlcMcManager ist nicht für die Bedienung von Maschinen und Anlagen vorgesehen. Er ersetzt keine Bedienoberfläche.

**Sicherheitshinweis**

**HINWEIS**

**Unerwartetes Maschinenverhalten**

Der PlcMcManager kann durch von ihm ausgelöste Kommandos die automatisch ablaufenden Aktionen und Reaktionen der Steuerungs-Software behindern oder in eine unerwartete oder unerwünschte Richtung beeinflussen. Dabei kann es zu unerwarteten und gefährlichen Bewegungen kommen.

## Installation

**Für TC2:** Eine lizenzfreie Kopie des PlcMcManagers wird mit der Bibliothek bzw. der Dokumentation zur Verfügung gestellt. Wählen Sie einen geeigneten Pfad an und erstellen Sie anschließend auf dem Desktop des PCs eine Verknüpfung. Wird eine solche Verknüpfung nicht erstellt kann der PlcMcManager nur aus dem Explorer gestartet werden.

**Für TC3:** Beim Download der Library wird im Verzeichnis *C:\TwinCAT\Functions\TF5810-TC3\_Hydraulics-Positioning* eine lizenzfreie Kopie des PlcMcManagers angelegt. Sollte ihr TwinCAT nicht auf *C:* oder in einem anderen Verzeichnis installiert sein, ist der Pfad entsprechend anzupassen.

## Ausführen des PlcMcManagers

Wurde das Tool auf dem PC gespeichert kann es durch Doppelklicken gestartet werden.

## Offline-Anzeige einer Parameter-Datei

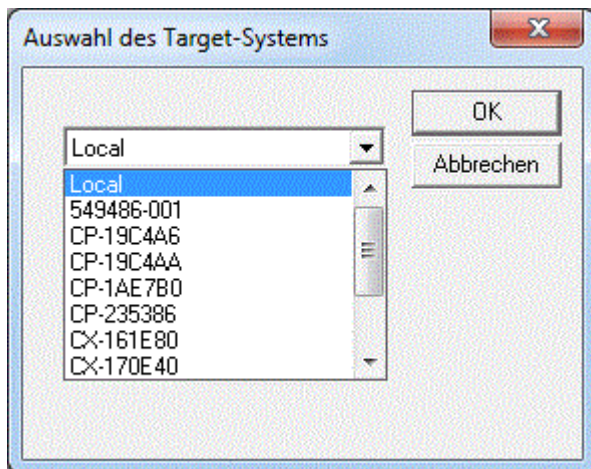
In der Menü-Leiste finden Sie unter **Online** den **Offline-Dateimodus**. Hier wird ein Dialog zur Auswahl einer Achsparameter-Datei vom Typ DAT angeboten. Wird eine Datei geöffnet werden die Parameter der Achse soweit möglich dargestellt wie im Online-Betrieb.



Es werden keine Istzustände von Achsen dargestellt und es können keinerlei Kommandos an Achsen ausgelöst werden. Dies gilt auch dann, wenn die angezeigten Parameter zu einer Achse gehören, auf die ein Zugriff möglich wäre.

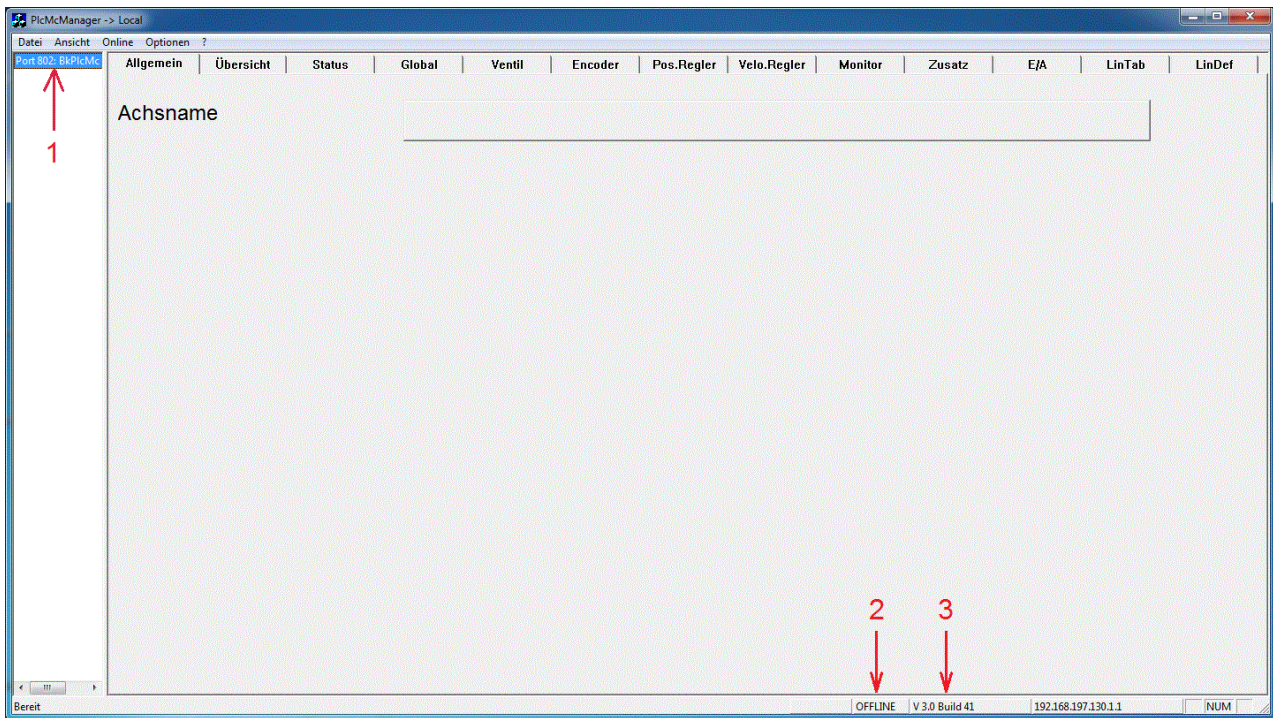
## Online-Betrieb

Befindet sich das Laufzeit-System mit den Bibliotheks-Bausteinen nicht auf dem PC, auf dem der PlcMcManager ausgeführt wird, ist zuerst das Zielsystem anzuwählen. In der Menü-Leiste finden Sie dafür unter **Online** den Dialog **Target**. Hier werden die Rechner aufgelistet, die im **TwinCAT System Service** auf dem Reiter **AMS Router** als **Remote Computer** eingetragen sind.



Durch die Auswahl eines **Remote Computer** wird die Kommunikation mit dem Laufzeitsystem automatisch aktiviert. Befindet sich das Laufzeit-System mit den Bibliotheks-Bausteinen auf dem PC, auf dem der PlcMcManager ausgeführt wird kann die Kommunikation mit dem Laufzeitsystem über die Menü-Leiste unter **Online** mit **Einloggen** aktiviert werden.

In den aktuellen Versionen ist der PlcMcManager für die Nutzung unter TC3 vorbereitet. Um die Verbindung zur Laufzeit herzustellen prüft er sowohl für TC2 als auch für TC3 die zu erwartenden ADS-Adressen. Dies kann vor allem bei Verwendung einer Netzwerkverbindung einige Sekunden dauern. Anschließend sollten die unten gezeigten Details erscheinen:



1. Hier werden der Port und der Server angezeigt, über den mit dem Laufzeitsystem kommuniziert wird.
2. Der Modus wird angezeigt. Da bis zu diesem Zeitpunkt noch keine Achse angewählt wurde ist der PlcMcManager noch im OFFLINE Modus.
3. Die Versions-Information der von der PLC-Applikation verwendeten Bibliothek wird angezeigt.

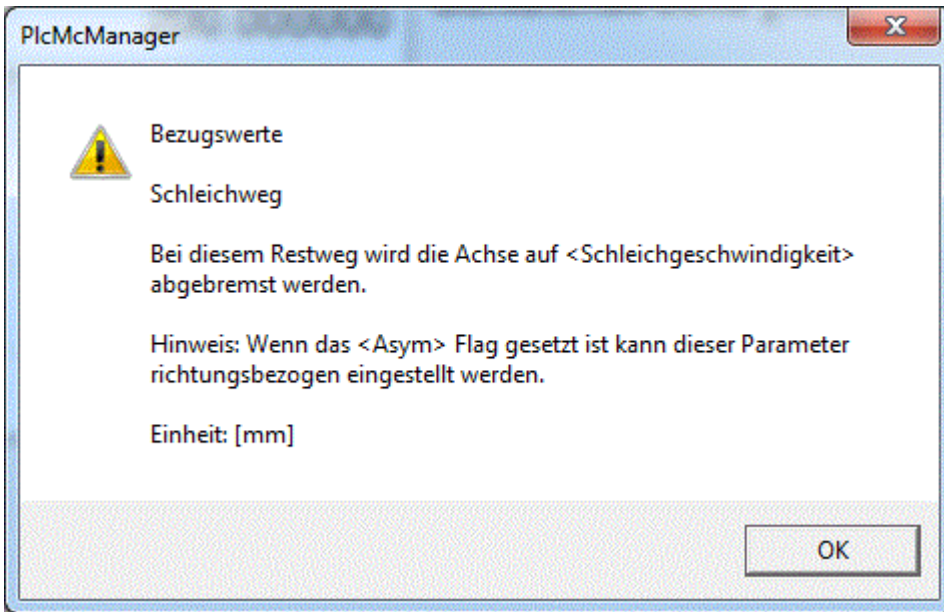
Sollten diese Details auch nach einigen Sekunden nicht erscheinen ist keine Verbindung zustande gekommen. Dies kann eine Reihe von Ursachen haben:

- Es wurde kein Ziel-System angewählt, obwohl die Applikation nicht auf demselben Rechner läuft, der den PlcMcManager ausführt.
- Die PLC-Applikation enthält keinen MC\_AxAdsCommServer\_BkPlcMc [▶ 263] Baustein oder ruft diesen nicht auf.
- Die Applikation läuft nicht auf dem angewählten Ziel-System.
- Es besteht keine Verbindung zum angewählten Ziel-System.
- Der PC auf dem der PlcMcManager ausgeführt wird hat keine Zugriffsrechte auf dem angewählten Ziel-System.
- Die PLC ist nicht gestartet.

Sollte zu diesem Zeitpunkt ein Dialog mit einer Fehlermeldung erscheinen ist die Verbindung zum Zielsystem gestört (Timeout / Zeitüberschreitung) oder der PlcMcManager und die in der Applikation verwendete Bibliothek sind nicht kompatibel. Letzteres ist in der Regel darauf zurück zu führen, dass eine neue Bibliotheks-Version verwendet wird ohne dass der PlcMcManager aktualisiert wurde.

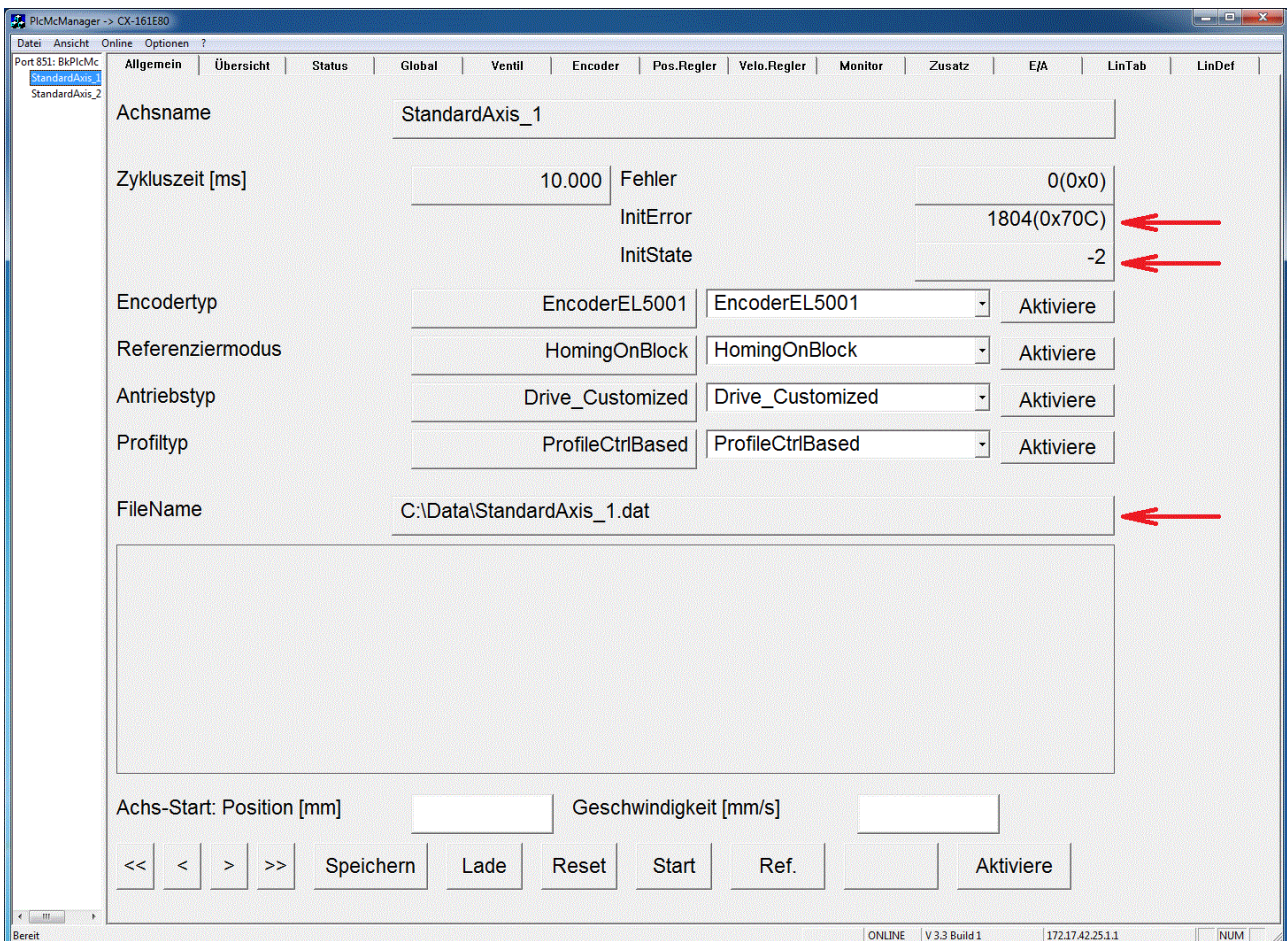
Viele Parameter-Eingabefelder besitzen auf der linken Seite ein „?“-Feld. Mit diesem kann eine kurze Erläuterung zum Parameter aufgerufen werden.

Beispiel: Erläuterung zum Parameter <Global.Schleichweg:



**Erste Schritte**

Durch Doppelklicken des auf der linken Seite angezeigten Servers erscheinen die Achsen der Applikation als Liste. Durch Anklicken einer Achse wird diese angewählt, ihr Status wird zyklisch aktualisiert und ihre Parameter sind zugreifbar. Sollte aus irgendeinem Grund die Kommunikation abbrechen kann durch Anklicken einer Achse ein Neustart der Kommunikation ausgelöst werden.



In diesem Beispiel wird der für diese Achse verwendete Datei-Pfad und -Name gezeigt. Allerdings wird ein **InitError 1804(0x70C)** und ein **InitState** von **-2** gemeldet. Der Error-Code signalisiert einen Datei-Fehler und der InitState ist „negativ beendet“. Hierfür sind mehrere Ursachen denkbar:



- Der Pfad existiert nicht auf dem Rechner, auch dem die PLC Applikation läuft. Hier kann es leicht zu Problemen kommen, wenn die Applikation erstmalig auf einem anderen System Online geht.
- Der Pfad ist vom Ort der PLC Laufzeit nicht erreichbar. Dies ist zum Beispiel möglich, wenn der Pfad in ein Netzwerk zeigt.
- Auf diesem Pfad ist ein Lesen und/oder Schreiben nicht zulässig.
- Pfad oder Dateiname sind nicht korrekt geschrieben. Möglicherweise fehlt der Backslash am Ende des Pfadnamens.
- Unter dem angegebenen Pfadnamen existiert keine entsprechende Datei.

Die zuletzt aufgeführte Ursache tritt stets dann auf, wenn die Inbetriebnahme einer PLC Applikation ohne vorhandene Datei begonnen wird. Um eine Datei mit Default-Parametern zu erzeugen ist durch die Taste **[Speichern]** ein Schreibvorgang mit den Initial-Werten der Parameter auszulösen. Durch die Taste **[Reset]** wird der Fehlerzustand gelöscht und in diesem Fall das Laden der Parameter aus der Datei wiederholt. Kann das Problem nicht durch diese Vorgehensweise gelöst werden ist es durch eine andere der aufgeführten Ursachen erzeugt.

## Daten und Kommandos

Der PlcMcManager stellt nur Variablen aus der PLC grafisch dar. Laufzeitwerte sind in der AxisRef in stRtData zu finden. Parameter, die über den PlcMcManager verändert werden, müssen über den Button **Aktiviere** aktiv in die Variablen der PLC geschrieben werden. Alle Werte, die permanent gespeichert werden müssen, liegen in der AxisRef unter stAxParams. Das Speichern dieser Parameter erfolgt durch die PLC und nicht durch den PlcMcManager.

Wird der Achse durch die PLC mit einem MC\_Power\_BkPlcMc-Baustein eine Regler- und Vorschubfreigabe erteilt, kann diese über die Jog Tasten (<, <<, >>, >) bewegt werden. Aktuell handelt es sich noch um eine simulierte Achse. Über die Felder **Position** und **Geschwindigkeit** kann die Achse auch kommandiert werden. Der Fahrbefehl wird über den Button **Start** ausgeführt.

## 5.7 Programm-Beispiele (ab V3.0)

### Aufbau der Applikation

Die Applikation wird weitgehend aus PLCopen Bausteinen aufgebaut. Es steht eine Auswahl an Funktionsbausteinen zur Verfügung, die mit einem durch die PLCopen definierten Interface ausgestattet sind. Nachstehend wird eine Reihe von Beispielen beschrieben, die eine gute Grundlage bei der Projekterstellung darstellen.

Jedes Beispiel beinhaltet die Projektdatei, die benötigten Achsparameterdateien und eine Scope-Konfiguration. Die Achsparameterdateien müssen in einem Ordner auf dem Zielsystem hinterlegt werden. Der Dateipfad muss in der globalen Konstanten „cnst\_ParamFilePath“ der Projektdatei angepasst werden.

### Beispiel 1: Einzelachse

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599853451.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937463179.zip>

Der [MC\\_AxUtiStandardInit\\_BkPlcMc \[► 238\]](#) lädt die Parameter und überwacht die Pointeradressen. Nachdem die Daten erfolgreich geladen wurden, wird „bParamsEnable“ TRUE und die eigentlichen Achsbausteine werden aufgerufen.

Der [MC\\_AxStandardBody\\_BkPlcMc \[► 237\]](#) ruft intern die erforderlichen Bausteine wie [MC\\_AxRtEncoder\\_BkPlcMc \[► 183\]](#), [MC\\_AxRuntime\\_BkPlcMc \[► 221\]](#), [MC\\_AxRtFinish\\_BkPlcMc \[► 230\]](#) und [MC\\_AxRtDrive\\_BkPlcMc \[► 172\]](#) auf. Wird jedoch ein Filter, ein Druckregler, eine Kennlinienvermessung oder ähnliches benötigt, müssen statt dem [MC\\_AxStandardBody\\_BkPlcMc \[► 237\]](#) die Einzelkomponenten aufgerufen werden.

Durch die Verwendung eines [MC\\_AxAdsCommServer\\_BkPlcMc \[► 263\]](#) kann die Achse über den PlcMcManager kommandiert werden. Der Baustein [MC\\_AxParamDelayedSave\\_BkPlcMc](#) speichert nach einer vorgegebenen Zeit (hier 10s) Änderungen, die vom PlcMcManager gemacht wurden. Über den PlcMcManager kann man sich auf dem Zielsystem einloggen und die Achse aktiv verfahren.

**Beispiel 2: Mehrachs-Applikation**

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599855627.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937465739.zip>

Das Beispiel zeigt die Arbeit mit Arrays von Bausteinen und Strukturen. Der Funktionsumfang entspricht Beispiel 1.

**Beispiel 3: Druckgeregeltes Abbremsen**

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599857803.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937468299.zip>

Das Beispiel zeigt, wie der Druckregler [MC\\_AxCtrlSlowDownOnPressure\\_BkPlcMc \[▶ 162\]](#) den Vorschub einer Achse abhängig vom Druck drosselt. Der Regler wird in diesem Beispiel aktiv, wenn der Istdruck größer als der Solldruck ist. Da das Ergebnis durch Applikationscode in „fLagCtrlOutput“ übergeben wird muss der Regler zwingend nach dem Sollwertgenerator aufgerufen werden. Andernfalls würde fLagCtrlOutput vom Lageregler in [MC\\_AxRuntime\\_BkPlcMc \[▶ 221\]](#) überschrieben werden.

Wenn im PlcMcManager ein Kommando mit einer Geschwindigkeit von zum Beispiel 100 mm/s und einer Position von zum Beispiel 500 mm gestartet wird kann im Scope verfolgt werden, dass bei steigender Position der Druck kontinuierlich ansteigt. Bei einer Position von 400 mm hat das System den Solldruck von 50 Bar erreicht und bleibt stehen.

**Beispiel 5: Move-Bausteine**

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599859979.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937547659.zip>

In diesem Beispiel werden verschiedene Funktionsblöcke für die Bewegungssteuerung eingesetzt. Wird die Variable bStart TRUE startet die Statemachine die Achse mit [MC\\_MoveAbsolute\\_BkPlcMc \[▶ 72\]](#) zur Position 500 mm. Wenn die Achse im Ziel angekommen ist und die Zielfensterbedingungen erfüllt hat (In PosRang, in TargetRange für TargetFilterTime und in BrakeDistance), startet automatisch ein [MC\\_MoveVelocity\\_BkPlcMc \[▶ 78\]](#) mit einer Geschwindigkeit von 400 mm/s. Diese Geschwindigkeit bleibt 5 s aktiv und wird dann durch ein [MC\\_Stop\\_BkPlcMc \[▶ 81\]](#) beendet, so dass die Achse zum Stillstand kommt. Anschließend folgt eine relative Bewegung von 100 mm mit [MC\\_MoveRelative\\_BkPlcMc \[▶ 76\]](#) um danach auf Position 0.0 mm zu fahren. In den verschiedenen Bewegungsprofilen werden unterschiedliche Beschleunigungs- und Verzögerungsrampen verwendet.

**Beispiel 6: Zeitrampen-Generator**

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599862155.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937550219.zip>

Eine Achse ohne Encoder kann nicht über den Standard Sollwertgenerator geführt werden. Für diese Art von Achsen steht mit [iTcMc\\_ProfileTimeRamp \[▶ 225\]](#) ein alternativer Sollwertgenerator zur Verfügung. Wird im Beispiel in den Globalen Variablen die Variable „bUp“ bzw. „bDown“ TRUE, so fährt die Achse mit der vorgegebenen Geschwindigkeit (hier 500 mm/s) bis zum ersten Endschalter (DigCamP – für Positiv/ DigCamM – für Negativ) und bremst dann auf die entsprechende Schleichgeschwindigkeit herunter. Nach Erreichen von DigCamPP – für Positiv/ DigCamMM – für negativ wird die Ausgabe abgelöscht.

**Beispiel 7: Override und Funktionsgenerator**

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599864331.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937552779.zip>

Demonstration des Bausteins [MC\\_SetOverride\\_BkPlcMc \[▶ 41\]](#). Über globale Variablen (bOverrideSinusoidal, fOverrideCycleTime, fOverrideMinValue, fOverrideMaxValue) kann der Verlauf, die Periodendauer und die Begrenzungen eines Signalerzeugers festgelegt werden, der den Override verändert. Für die Erzeugung des Overrides werden [MC\\_FunctionGeneratorFD\\_BkPlcMc \[▶ 211\]](#), [MC\\_FunctionGeneratorTB\\_BkPlcMc \[▶ 212\]](#) und [MC\\_FunctionGeneratorSetFrq\\_BkPlcMc \[▶ 212\]](#) Bausteine eingesetzt.

### Beispiel 8: Digitales Nockenschaltwerk

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599866507.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937555339.zip>

Das Beispiel zeigt, wie man durch eine Achse und [MC\\_DigitalCamSwitch\\_BkPlcMc \[► 53\]](#) digitale Nocken steuern kann. Im Beispiel sind zwei Nocken in [TRACK\\_REF\\_BkPlcMc \[► 116\]](#) aktiviert (Maximal 32). Die erste Nocke wird zu drei verschiedenen Bedingungen aktiv geschaltet:

1. Von Position -1000 mm bis 1000 mm und positive Richtung
2. Von Position 2000 mm bis 3000 mm und positive Richtung
3. Von Position 3000 mm bis 2500 mm und negative Richtung

Die Zweite Nocke hat nur eine Bedingung:

1. Ab Position 3000 mm für eine Zeit von 1.35 s in positiver und negativer Richtung aktiv zu sein.

Neben den Schaltbedingungen kann eine Nocke eine Einschalt- sowie Ausschaltverzögerung mitbringen.

Für Nocke 1 ist die Einschaltverzögerung 0.125 s und die Ausschaltverzögerung 0.250 s eingestellt. Die Bedingungen für das Schalten einer Nocke werden in [CAMSWITCH\\_REF\\_BkPlcMc \[► 114\]](#) festgelegt. Der

Ausgang einer Nocke ist in [OUTPUT\\_REF\\_BkPlcMc \[► 117\]](#) angegeben.

Die Achse ist über den PlcMcManger zu kommandieren (Position größer 3000 mm).

### Beispiel 9: Joystick

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599868683.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937557899.zip>

Das Beispiel demonstriert die Verwendung des Bausteins [MC\\_MoveJoySticked\\_BkPlcMc \[► 74\]](#). Bei diesem Baustein wird die Achse in eine endlose Bewegung mit einer durch JoyStick vorgegebenen Geschwindigkeit versetzt. Joystick ist ein normierter Wert zwischen +/-1.0, welcher multipliziert mit der kommandierten Geschwindigkeit die Sollgeschwindigkeit ergibt.

### Beispiel 10: Identifikation und Linearisierung

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599870859.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937611659.zip>

Das Beispiel beschreibt die automatische Kennlinienvermessung mit [MC\\_AxUtiAutoldent\\_BkPlcMc \[► 254\]](#) und die Verwendung der Kennlinie mit [MC\\_AxRtFinishLinear\\_BkPlcMc \[► 231\]](#). Die Einstellungen für die automatische Kennlinienvermessung sind im PlcMcManger unter dem Reiter **LinDef** zugänglich und finden sich in der Struktur [ST\\_TcMcAutoldent \[► 118\]](#) wieder.

Im Beispiel kann über die Globale Variable nTest zwischen drei verschiedenen Ventilsimulationen gewählt werden. Entsprechend der ausgewählten Simulation wird eine passende *.dat* Datei geladen. In der *.dat*-Datei sind die Parameter für die Kennlinienvermessung entsprechend voreingestellt. Achtung: Wird nTest während die PLC läuft umgeschaltet, muss der PlcMcManager neu verbunden werden. Folgende Szenarien können über nTest ausgewählt werden:

1. Vermisst lediglich die Überdeckung und das Geschwindigkeitsverhältnis
2. Vermisst eine Nullschnitt-Kennlinie mit Knick
3. Vermisst eine Kennlinie mit Überdeckung

Durch die Variable „bStartAuto“ kann [MC\\_AxUtiAutoldent\\_BkPlcMc \[► 254\]](#) gestartet werden. Während der Vermessung gibt der Baustein ein Busy zurück und auf dem Reiter **LinTab** wird die bereits vermessene Kennlinie angezeigt.

War die Vermessung erfolgreich, so kann die Kennlinie durch den Baustein [MC\\_AxRtFinishLinear\\_BkPlcMc \[► 231\]](#) verwendet werden. Die Kennlinie wird automatisch in der *.dat*-Datei der Achse mitgespeichert und mitgeladen. Soll die Kennlinie ASCII lesbar exportiert werden so steht hierfür der Baustein [MC\\_AxTableToAsciiFile\\_BkPlcMc \[► 218\]](#) zur Verfügung.

### Beispiel 11: Stop-Bausteine

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599873035.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937614219.zip>

Hier werden die verschiedenen Möglichkeiten gegenübergestellt, eine Achse anzuhalten. Das Beispiel kann gestartet werden, in dem die Variable bStart TRUE gesetzt wird.

MC\_Stop\_BkPlcMc [▶ 81]: Führt ein Stopp mit vorgegeben Verzögerungsparametern durch. Die Achse meldet sich fertig, wenn das berechnete Ziel inklusive Zieltoleranzen (in PosRange, In Target Rage für Target Filter time und in BrakeDistance) erreicht ist.

MC\_EmergencyStop\_BkPlcMc [▶ 56]: Bremst mit vorgegebener Rampe bis zum Stillstand.

MC\_ImediateStop\_BkPlcMc [▶ 71]: Setzt den Sollwert ohne Rampe auf Null.

### Beispiel 12: Buffering und Blending

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599875211.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937616779.zip>

Das grundlegende Vorgehen bei gepufferten Bewegungen ist in FAQ 20 [▶ 315] erklärt. Um das Beispiel zu starten, muss die Variable bStart TRUE werden. Im Scope View ist zu erkennen, dass es sich hier um 6 Bewegungen handelt, die aneinandergekoppelt abgearbeitet werden.

### Beispiel 13: Filter

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599877387.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937619339.zip>

Das Beispiel zeigt das Verhalten mehrerer Filtertypen und worauf beim Einsatz von Filtern zu achten ist. Schaltet man im Scope View alle Signale mit dem Namen „Noisy“ aus, so ist das Originalsignal und dazu verschoben die gefilterten Signale zu erkennen. Dabei bleibt die Form des Signals erhalten. Je stärker ein Signal gefiltert wird, desto stärker ist jedoch die Phasenverschiebung zwischen originalelem und gefiltertem Signal. Diese Phasenverschiebung hat direkten Einfluss auf die Regelbarkeit von Achsen und anderen Strecken.

Werden die verrauschten Signale im Scope sichtbar gemacht ist zu erkennen, dass der Rauschanteil im Signal sowohl durch ein MC\_AxUtiSlidingAverage\_BkPlcMc [▶ 253] als auch nach einem MC\_AxUtiPT1\_BkPlcMc [▶ 250] wesentlich geringer ist.

### Beispiel 14: Funktionsgenerator

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599879563.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937621899.zip>

In einigen Anwendungen wird ein Sollwertgenerator benötigt, der Sinus-, Trapez- oder Sägezahn-Signale erzeugt. Die mit MC\_FunctionGeneratorTB\_BkPlcMc [▶ 212] und MC\_FunctionGeneratorFD\_BkPlcMc [▶ 211] erzeugten Signale können zum Beispiel über MC\_AxRtSetExtGenValues\_BkPlcMc [▶ 236] an eine Achse übergeben werden.

### Beispiel 15: Druckregler

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599881739.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937624459.zip>

Das Beispiel zeigt das Einlesen und Skalieren eines Druckaktualwerts in der Applikation. Eine Druckregelung für eine Achse mit MC\_AxCtrlPressure\_BkPlcMc [▶ 156] wird demonstriert. Die Applikation fährt zunächst über eine schnelle Bewegung eine Position an, ab der ein Druckanstieg erwartet wird. Die Bewegung wird mit einer geringeren Geschwindigkeit fortgesetzt und nach Erreichen des Solldruckes der Regler aktiviert.

### Beispiel 16: Verteilte Achs-Referenzen

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599883915.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937627019.zip>

Das Beispiel zeigt die Verwendung einer Liste von POINTER TO Axis\_Ref\_BkPlcMc. Die Verwendung von [MC\\_AxAdsPtrArrCommServer\\_BkPlcMc \[▶ 265\]](#) an Stelle von [MC\\_AxAdsCommServer\\_BkPlcMc \[▶ 263\]](#) ermöglicht es, die Achs-Referenzen zu verteilen.

Die Liste muss in jedem Zyklus aktualisiert werden. Diese Aktualisierung muss vor dem Aufruf des [MC\\_AxAdsPtrArrCommServer\\_BkPlcMc \[▶ 265\]](#) durchgeführt sein.

### Beispiel 17: Externe Sollwert-Generierung

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/6407024139.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/9326778891.zip>

Das einfache Beispiel zeigt die grundsätzliche Verwendung eines Bausteins vom Typ [MC\\_AxRtSetGenValues \[▶ 236\]](#).

### Beispiel 18: PlcMcManager sperren

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599886091.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937629579.zip>

Es kann notwendig sein, Kommandos des PlcMcManagers wie Jog, MoveAbs oder Stopp zu sperren. Dies kann in der PLC mit [MC\\_AxRtCommandsLocked\\_BkPlcMc \[▶ 246\]](#) erfolgen.

### Beispiel 19: Externe Sollwert-Generierung

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/9326087819.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/9326781195.zip>

Hier werden mit einem umfangreicheren Projekt die Möglichkeiten der externen Sollwert-Generierung demonstriert.

### Beispiel 100: Elektronisches Getriebe

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599888267.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937663627.zip>

Das Beispiel zeigt, wie über eine Masterachse (Achse 3) zwei Slave Achsen durch ein elektronisches Getriebe verkoppelt werden können. Das Erzeugen und Lösen der Kopplung geschieht durch [MC\\_GearIn\\_BkPlcMc \[▶ 62\]](#) und [MC\\_GearOut\\_BkPlcMc \[▶ 66\]](#).

Es muss darauf geachtet werden, dass die Dynamikparameter vom Master und Slave zueinander passen, da sonst der Slave nicht dem Master folgen kann.

Zur Herstellung der Kopplung müssen Master und Slave im Ruhezustand sein. Das Lösen der Kopplung kann in der Bewegung passieren. Dabei fährt die Masterachse bis zum Ziel und die Slave Achse wird beim Lösen der Kopplung abgestoppt.

### Beispiel 101: Elektronische Kurvenscheibe

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599890443.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937666187.zip>

Die Achse 1 und 2 wird über eine Kurvenscheibe an die virtuelle Achse 3 gekoppelt. Die Kopplungsparameter für Achse 1 sind in diesem Beispiel in der Textdatei *TcPlcMcEx\_101\_2.txt* abgelegt. Für Achse 2 werden die Kopplungsparameter im Baustein „FB\_CalculateCamTable2“ berechnet. Über [MC\\_CamTableSelect\\_BkPlcMc \[▶ 52\]](#) werden die Master- und Slave-Achse und die Kurvenscheibentabelle festgelegt. Im Baustein [MC\\_CamIn\\_BkPlcMc \[▶ 48\]](#) wird die Kopplung erzeugt und die Sollwerte für den Slave berechnet. Wird die Masterachse über den PlcMcManager verfahren, folgt die Slave Achse nach der entsprechenden Kurvenscheibe. Mit [MC\\_CamOut\\_BkPlcMc \[▶ 50\]](#) wird die Kopplung aufgehoben.

### Beispiel 103: Fliegende Getriebekopplung

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599892619.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937668747.zip>

Demonstration einer fliegend aktivierten Getriebekopplung mit den Bausteinen [MC\\_GearInPos\\_BkPlcMc](#) [► 64] und [MC\\_GearOut\\_BkPlcMc](#) [► 66].

### Beispiel 104: Gleichlaufregelung

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599894795.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937671307.zip>

Demonstration einer Gleichlaufregelung für ein Zweiachs-Gantry unter Verwendung eines virtuellen Masters. Eine Gleichlaufregelung findet immer dort Anwendung, wo zwei oder mehrere Achsen balanciert geregelt werden müssen. Dabei wird eine virtuelle Masterachse für die Sollwertgenerierung verwendet. Die Sollwerte werden auf die Slaveachsen verteilt, welche ihren lokalen Lagereger addieren. Die aktuelle Position der virtuellen Masterachse wird zum Beispiel als Mittelwert über die Slaveachsen gebildet.

Für eine reibungslose Inbetriebnahme ist unbedingt darauf zu achten, dass bestimmte Parameter gleich gehalten werden. Dies gilt in einigen Fällen innerhalb der Gruppe der Slaveachsen, zum Teil auch für die Masterachse. In „FB\_Parameter“ wird dies durch zyklisches Kopieren erzwungen.

### Beispiel 105: Linearisierung für Gleichlaufregelung

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/1599896971.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/4937673867.zip>

Dieses Beispiel demonstriert die Kennlinienermittlung für ein Zweiachs-Gantry (siehe auch Beispiel 104) mit den Bausteinen [MC\\_AxUtiAutoldent\\_BkPlcMc](#) [► 254] und [MC\\_AxUtiAutoldentSlave\\_BkPlcMc](#).

### Beispiel 106: Fliegende Kopplung

Für TC2: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/9326092427.zip>

Für TC3: <https://infosys.beckhoff.com/content/1031/tcplclibhydraulics30/Resources/9326783499.zip>

Hier werden einige der Möglichkeiten dargestellt, die ein fliegendes Ein- und Auskoppeln eines elektronischen Getriebes bietet.

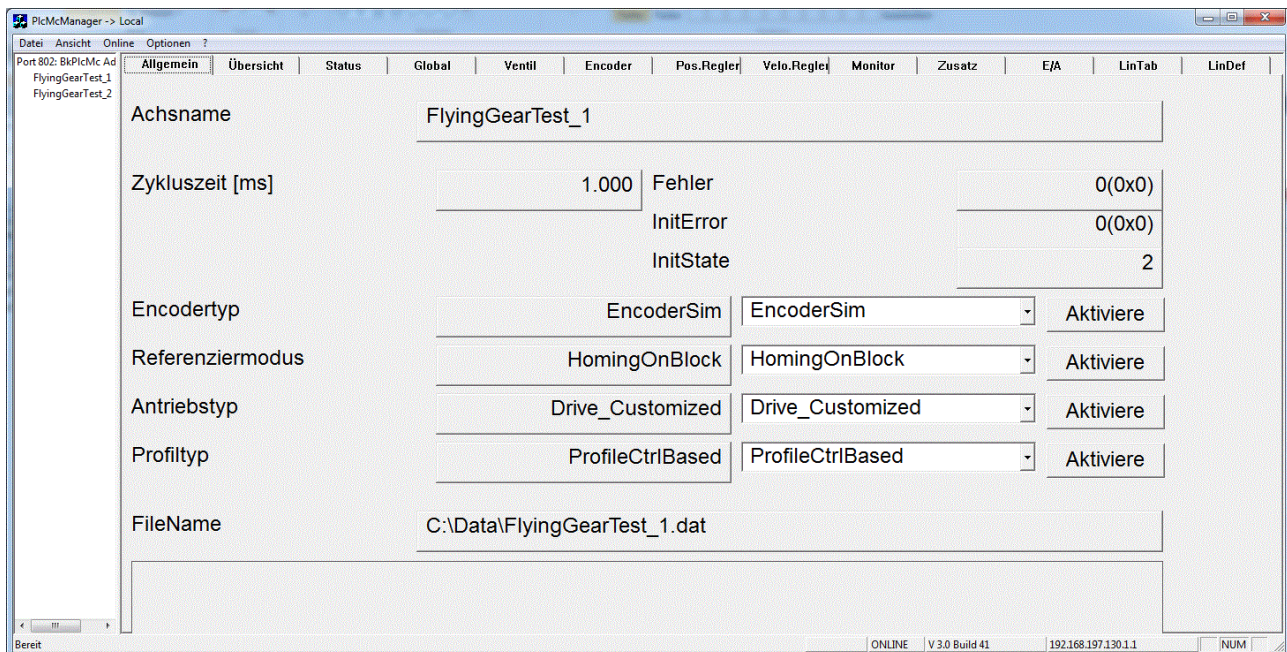
## 5.8 Inbetriebnahme

Die hier dargestellte Vorgehensweise beschreibt die Grundinbetriebnahme einer Achse, von der nichts bekannt ist. Bei baugleichen Achsen können verschiedene Punkte übersprungen werden.

### 5.8.1 Grundeinstellungen

Um die reale Achse in Betrieb zu nehmen, müssen verschiedene Voreinstellungen getätigt werden.

Auf dem Reiter **Allgemein** muss der entsprechende Encodertyp eingegeben werden. Hierzu muss über das Auswahlménü der entsprechende Encoder angewählt werden und über **Aktiviere** in die Laufzeit-Variablen geschrieben werden. Links vom Auswahlfenster wird der aktuell aktive Typ angezeigt.



Die Knowledge Base enthält eine [Tabelle \[► 306\]](#), die bei der Wahl des richtigen Encodertyps hilft und das Mappinginterface zum E/A erklärt.

Ist es aus technischen Gründen nicht möglich, die Istposition mit dem Standard-Encoderbaustein der Library zu ermitteln, kann diese Aufgabe auch durch Applikationsbausteine ausgeführt werden. Dann ist das Ergebnis in `fActPos` und `fActVelo` in `ST_TcHydAxRtData` einzutragen und in `fActPosDelta` die Positionsänderung im aktuellen Zyklus zu aktualisieren. In `bEncoderResponse` ist zu signalisieren, ob die Istwerte aktualisiert werden konnten. Um die Durchgängigkeit zu erhalten sollte auch hier wenn möglich auf die Parameter `fEnc_IncWeighting`, `fEnc_InclInterpolation` und `fEnc_ZeroShift` bzw. `fEnc_RefShift` zurückgegriffen werden.

Als Steller (Drivetype) kommen eine Reihe von Geräten und Einrichtungen in Frage, die abhängig von einer elektrischen Größe nach diversen physikalischen Prinzipien eine variable Geschwindigkeit kontrollieren. Abhängig von der entsprechenden E/A Komponente muss im Auswahlfenster der Drivetype eingestellt werden und die Variablen mit dem Feldgerät verbunden werden. Die Knowledge Base enthält eine [Tabelle \[► 309\]](#) die bei der Wahl des einzustellenden Typs unterstützt.

Handelt es sich bei dem Wegmesssystem um ein inkrementelles System, so muss auch die entsprechende Referenzierungsmethode [\[► 96\]](#) festgelegt werden.

Auf dem Reiter **Global** sollte für die Referenzgeschwindigkeit zunächst 100 eingetragen werden. Der Wert wird später korrigiert, aber auf diese Weise können Überdeckungen usw. direkt in % eingetragen werden.

Die Beschleunigung und Verzögerung sollte auf 100 mm/s<sup>2</sup> gestellt werden. Bei dieser Einstellung wird diese Achse in 1 s auf Referenzgeschwindigkeit beschleunigen. Die Jogparameter sollten auf 5 mm/s und 10 mm/s gestellt werden. Die Schleichgeschwindigkeit sollte auf 5 mm/s stehen, die Schleichstrecke auf 10 mm und die Bremsstrecke auf 2 mm.

Wenn es sich um ein überdecktes Ventil handelt und das Ventildatenblatt vorhanden ist, kann auf dem Reiter **Ventil** die Überdeckung vom Datenblatt eingetragen werden.

Auf dem Reiter **Encoder** ist in **Inc. Bewertung** die Auflösung pro Inkrement anzugeben. Alternativ kann auch in **Inc. Zwischenteilung** eine Inkrementanzahl angegeben werden und in **Inc. Bewertung** der entsprechende Fahrweg.

Im Reiter **Regler** muss der Schlepp- und Velo-Regler auf Null stehen.

Für die weitere Inbetriebnahme sollte ein Scope mit folgenden Variablen aufgenommen werden:

- SetVelo
- ActVelo
- SetPos

- ActPos
- fOutput
- fLagCtrlOutput

Wenn vorhanden sind Drücke, Kräfte und Ventilschieberposition aufzuzeichnen.

Werden die Reglerfreigabe und Vorschubfreigabe der Achse gesetzt, darf sich die Achse nicht bewegen. Sollte dies der Fall sein, muss ein temporärer Nullpunktgleich durchgeführt werden.

## 5.8.2 Temporärer Nullpunktgleich

Auf dem Reiter **Regler** ist der Parameter **Offset Compensation** angelegt. Für diesen muss, je nachdem in welche Richtung die Achse driftet, ein Wert zwischen -10...+10 Volt eingetragen werden. In der Regel sind Werte von +/-0,5 Volt zu erwarten.

## 5.8.3 Bewegungsrichtungen

Über die -Jog-Button sollte sich die Achse langsam verfahren lassen. Ist dies nicht der Fall, muss die Druckversorgung kontrolliert werden. Des Weiteren sind ggf. Schaltventile zu betätigen oder die Kompensation der Ventil-Überdeckung ist zu klein eingestellt.

Es ist zu empfehlen, eine positive Bewegungsrichtung für die Achse festzulegen, die der Arbeitsweise der Maschine entspricht. Bewegt sich die Achse in diese Richtung sollte die Istposition aufwärts zählen. Ist dies nicht der Fall, kann auf dem Reiter **Encoder** die Zählrichtung invertiert werden. Stimmt die Änderungsrichtung der angezeigten Position mit der mechanischen Bewegung überein, aber die Wirkungsrichtung der gegebenen Kommandos ist nicht wie gewünscht kann auf dem Reiter **Ventil** die Ausgabe invertiert werden.



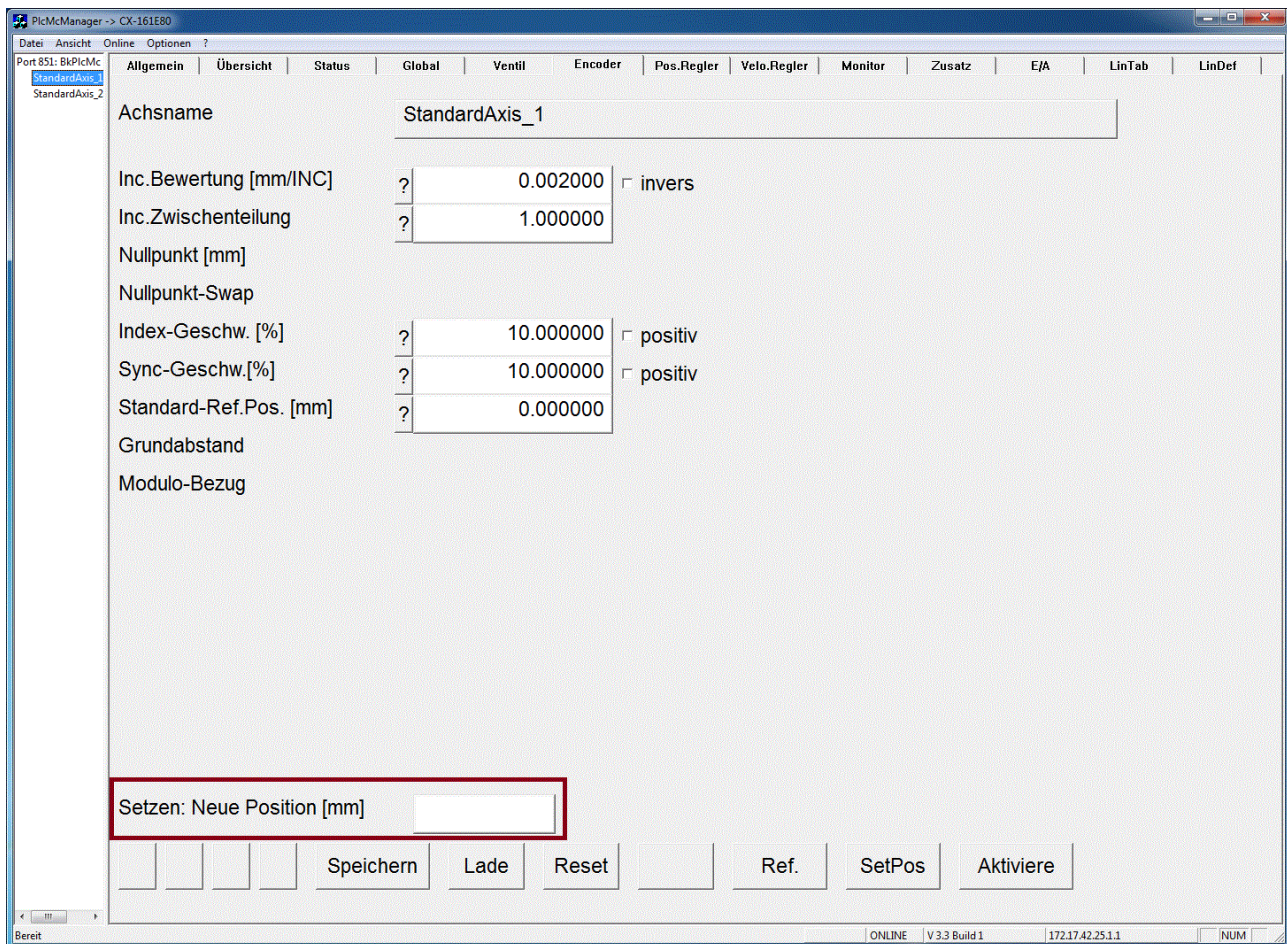
Wenn die Ventilausgabe invertiert wird, muss unbedingt die Offsetkompensation angepasst werden, da diese nicht mit invertiert wird und sich ihre Wirkung umkehrt.

---

## 5.8.4 Wegmesssystem

Sowohl bei einem absoluten wie auch einem inkrementellen Wegmesssystem sollte die Achse eine plausible Istposition zeigen. Der Nullpunkt des Encoders und der festgelegte Nullpunkt der Achse stimmen in der Regel nicht überein. Auf dem Reiter **Encoder** kann die gewünschte aktuelle Position eingegeben und über den Button **Set-Pos** in die Achse übernommen werden. Diese gesetzte Position muss zu diesem Zeitpunkt noch nicht zu 100 % exakt mit der realen Position übereinstimmen. Gerade bei inkrementellen Messsystemen wird später ein Homing durchgeführt.





Der PlcMcManager passt die Darstellung der Parameter soweit sinnvoll möglich an den eingestellten Encpdertyp an. Dadurch können bei verschiedenen Achsen unterschiedliche Parameter sichtbar werden.

Bei inkrementellen Encodertypen erscheint die oben gezeigte Darstellung. Die Sichtbarkeit der Parameter für das Homing hängt von der eingestellten Homing-Methode ab.

Um Kollisionen bei der Inbetriebnahme zu vermeiden sollten auf dem Reiter **Monitor** die Softwareendschalter aktiviert und passend eingestellt werden. Da die aktuelle Istposition sich von der realen leicht unterscheiden kann, ist es zu empfehlen, die Softwareendschalter etwas enger einzustellen.

## 5.8.5 Kennlinienvermessung

Über die Kennlinienvermessung ([MC\\_AxUtiAutoldent\\_BkPlcMc \[► 254\]](#)) wird nicht nur die Kennlinie selbst, sondern auch Bezugsgeschwindigkeit, Geschwindigkeitsverhältnis und optional Fahrwegsgrenzen ermittelt. Weitere Informationen zu den Einstelloptionen sind beim Baustein selbst erklärt.

Die Referenzgeschwindigkeit sollte auf einen annähernd plausiblen Wert voreingestellt werden. Eine Möglichkeit ist, die kleinere Zylinderfläche ( $A$  [mm<sup>2</sup>]) mit dem nominellen Volumenstrom ( $Q_n$  [l/min]) des Ventils zu verrechnen:

$$V_{ref} := Q_n \cdot 1.000.000 / 60 / A$$

Über den Reiter **LinDef** können verschiedene Einstellungen vorgenommen werden. Weitere Informationen sind hier zu finden.

Wenn dies aktiviert ist beginnt der Autoldent-Baustein als erstes damit, die Fahrwegsgrenzen zu ermitteln. Anschließend wird die Achse mind. 10 % von den Fahrwegsgrenzen entfernt positioniert, um die Überdeckung zu ermitteln. Wenn das erfolgreich durchgeführt wurde, fährt die Achse an das untere Ende und beginnt mit der Vermessung. Je nach dem zur Verfügung stehenden Fahrweg werden pro Richtung mehrere Messungen durchgeführt.

Nachdem die Kennlinie erfolgreich vermessen worden ist, kann sie im Reiter **LinTab** angesehen werden. Eine erfolgreich vermessene Kennlinie ist daran zu erkennen, dass `stParams.bLinTabAvaiable` TRUE ist.

Das Kapitel Überdeckung und Referenzgeschwindigkeit sind bei einer erfolgreich vermessenen Kennlinie zu überspringen.

## 5.8.6 Überdeckung

Um die Überdeckung zu ermitteln, muss die Sollgeschwindigkeit langsam erhöht werden, bis eine Reaktion an der Istgeschwindigkeit zu erkennen ist. Es ist möglich, dass die Sollgeschwindigkeit auf einen Wert von bis zu 30mm/s erhöht werden muss, bevor eine Reaktion an der Istgeschwindigkeit zu sehen ist. Beim Ausmessen der Überdeckung sollte die Überdeckung selbst immer wieder auf Null gesetzt werden.

Werden unterschiedliche Geschwindigkeitssollwerte benötigt, um die Achse in positiver bzw. negativer Richtung aus dem Stillstand in Bewegung zu versetzen, deutet dies auf ein nicht symmetrisches Ventil hin. In diesem Fall muss der Haken **Asym** auf dem Reiter **Global** gesetzt und aktiviert werden. Nun kann die Parametrierung für das Ventil in positive und negative Richtungen separat vorgenommen werden.

Die Sollgeschwindigkeit, bei der sich die Achse bewegt, muss auf dem „Ventil“-Reiter in Überdeckung eingetragen werden. Steht schon ein Wert in der Überdeckung, muss dieser Wert mitberücksichtigt werden. Bei einem nicht symmetrischen Ventil ist darauf zu achten, dass die Eintragung im richtigen Feld vorgenommen wird, wobei im oberen Feld die Überlappung für die positive Richtung und im unteren Feld für die negative Richtung erwartet wird.

Nach dieser Optimierung muss die Achse auch bei verschiedenen kleinen Geschwindigkeiten reagieren. Dabei ist es nicht wichtig, ob die Achse mit der richtigen Geschwindigkeit reagiert.

Wurde eine Überdeckung aus dem Datenblatt eingetragen und die Achse fährt immer zu schnell, so muss die Überdeckung reduziert werden.

## 5.8.7 Referenzgeschwindigkeit/Geschwindigkeitsverhältnis



Dieses Kapitel beschreibt eine manuelle Inbetriebnahme. Eine Kennlinienvermessung ermittelt auch die hier behandelten Parameter. Wird sie genutzt ist dieses Kapitel zu überspringen.

Nachdem die Achse mit kleiner Geschwindigkeit verfahren werden kann, muss nun die Referenzgeschwindigkeit eingestellt werden.

Um die Bezugsgeschwindigkeit zu ermitteln, wird die Sollgeschwindigkeit schrittweise erhöht und kontrolliert, ob die Achse mit annähernd der Sollgeschwindigkeit folgt.

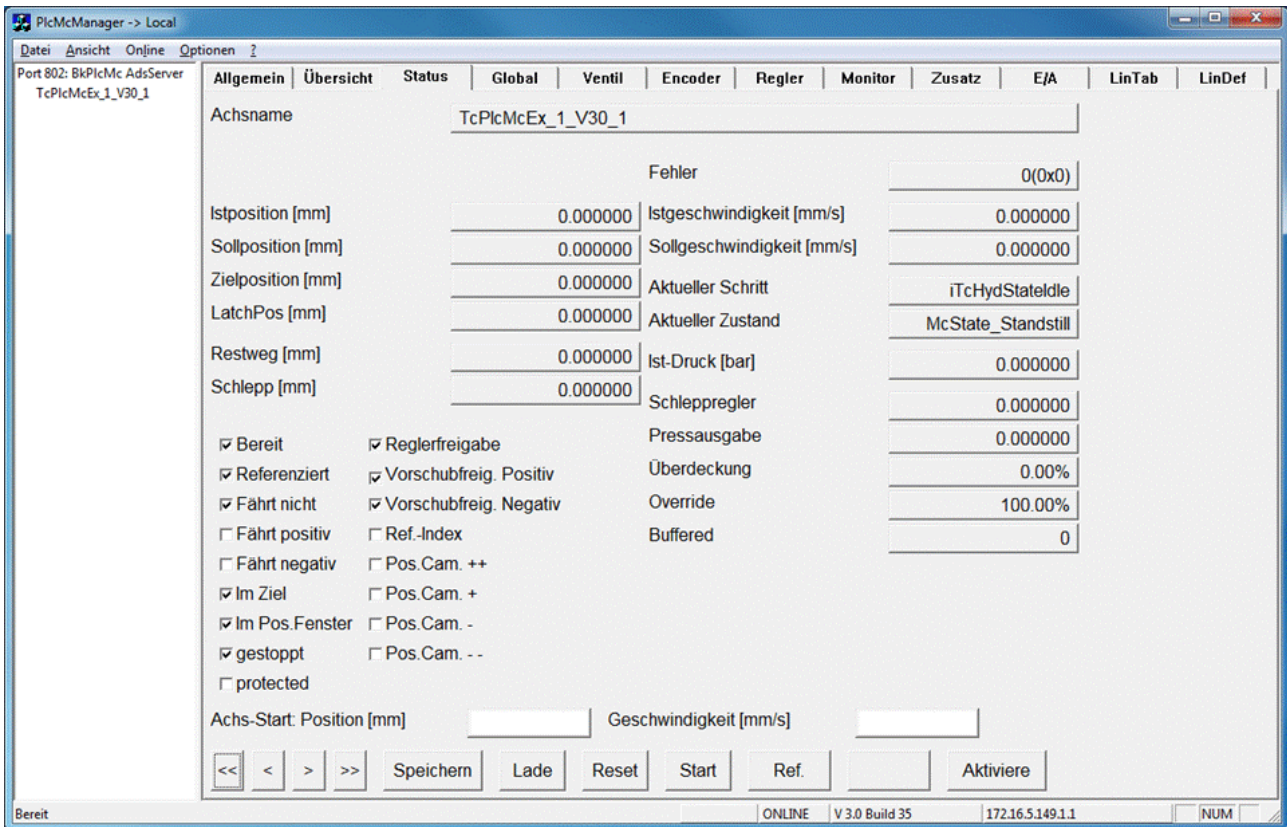


In diesem Schritt sind nur Bewegungen in der schnelleren Richtung auszuwerten. Dabei wird das Öl in die kleine Kolbenfläche transportiert! Die Richtungsabhängigkeit wird im nächsten Schritt behandelt.

Um die benötigten Bewegungen auszulösen kann im Reiter **Status** die Position und die Geschwindigkeit vorgegeben werden. Mit der Start-Taste wird die Bewegung ausgeführt. Hierbei sollte das vorher erzeugte Scope verwendet werden, um anschließend die Geschwindigkeiten analysieren zu können.



Die Softwareendschalter sollten aktiviert und so eingestellt werden, dass die Achse nicht gegen die mechanischen Endanschläge fährt.



Ist die Istgeschwindigkeit wesentlich kleiner als die Sollgeschwindigkeit, so muss die Bezugsgeschwindigkeit verkleinert werden.

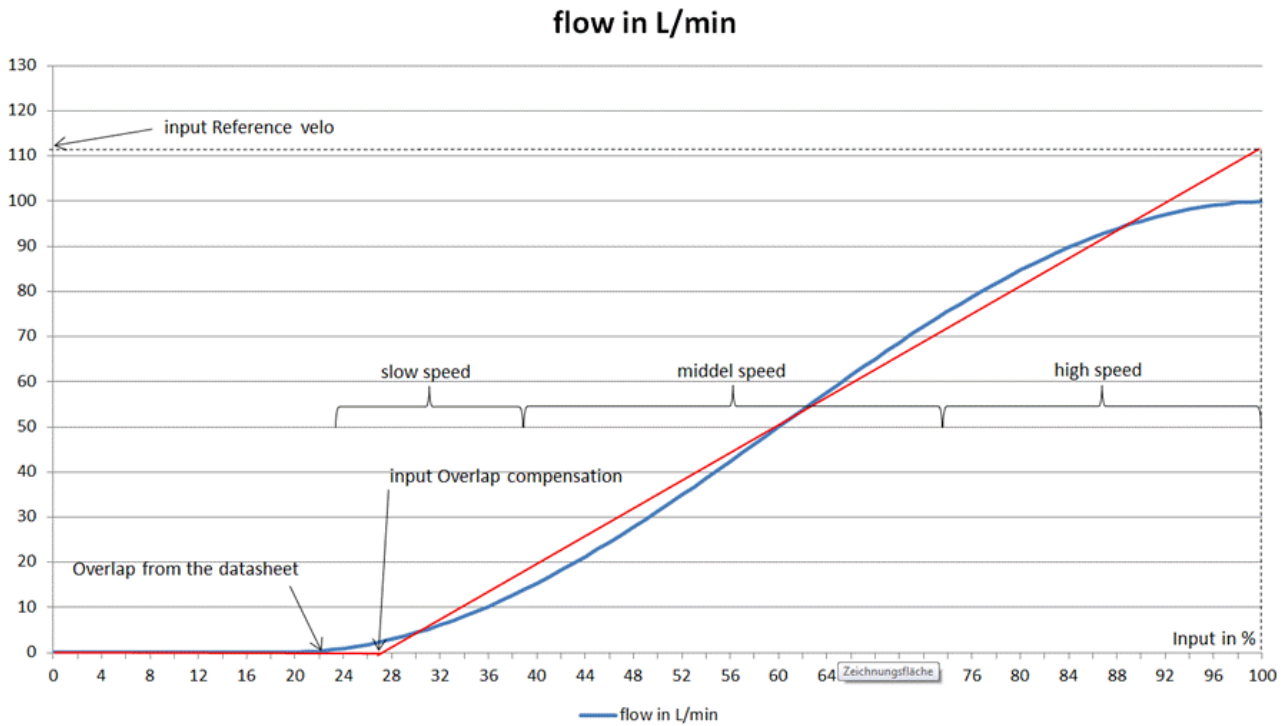
Ist die Istgeschwindigkeit wesentlich größer als die Sollgeschwindigkeit, so muss die Bezugsgeschwindigkeit vergrößert werden.

Stimmen die mittleren bis hohen Soll- und Istgeschwindigkeit annähert überein, so ist die passende Bezugsgeschwindigkeit gefunden.



Die Bezugsgeschwindigkeit muss nicht mit der realen oder berechneten maximalen Geschwindigkeit der Achse übereinstimmen.

Die nachfolgende Abbildung zeigt die abschnittsweise Linearisierung durch Überdeckung und Bezugsgeschwindigkeit bei einer nicht linearen Kennlinie. Dabei ist dem Anwender überlassen, wo die größte Abweichung zwischen Linearisierung und realer Kennlinie entstehen darf.



Die übliche Asymmetrie der Zylinder führt dazu, dass die Achse bei eingestellter Referenzgeschwindigkeit bei jeder kommandierten Geschwindigkeit in der langsameren Richtung zu langsam fährt. Dieses Verhalten kann auf dem Ventil-Reiter durch den Parameter Geschw.-Verhältnis kompensiert werden.

Bei einem symmetrischen Verhalten ist dieser Parameter auf 1.000 zu stellen. Ist die positive Fahrtrichtung die langsamere Richtung ist ein Wert >1.000 zu verwenden. Sollte die negative Fahrtrichtung die langsamere Richtung sein ist ein Wert <1.000 zu verwenden. Dadurch wird die Ausgabe in der langsameren Richtung erhöht und die Asymmetrie ausgeglichen.

**i** Durch diese Kompensation kann die Ausgabe nur bis zu ihrem Maximalwert erhöht werden. Die Parametrierung muss bei Geschwindigkeiten durchgeführt werden, die die Achse in beiden Richtungen erreichen kann.

**i** Wird der Parameter in die falsche Richtung verändert wird die Geschwindigkeit in der schnelleren Richtung verringert. In diesem Fall darf nicht die Referenzgeschwindigkeit korrigiert werden.

### 5.8.8 Referenzierungen

Bei inkrementellen Wegmeßsystemen: Spätestens jetzt sollte die Achse korrekt und vollständig referenziert werden. Unter dem Reiter **Encoder** muss die Index Geschwindigkeit, Index Richtung, Sync Geschwindigkeit, Sync Richtung sowie die Refpos eingegeben werden. Weitere Informationen finden Sie bei [MC Home BkPlcMc |> 68](#)].

**i** Es kann notwendig sein, die Fahrwegsgrenzen neu einzustellen.

### 5.8.9 Dynamik/Zielanfahrt

Die Achse ist zu diesem Zeitpunkt in der Lage, mit verschiedene Geschwindigkeiten und einer moderaten Dynamik zu positionieren.

Auf dem Reiter **Monitor** wird eingestellt, wann sich die Achse fertig melden soll. Eine Achse befindet sich im Ziel, wenn die Reststrecke kleiner als `PosRange` und `BrakeDistance` ist und für die `TargetFilterZeit` muss die Reststrecke kleiner als `Targetrange` sein. Diese drei Parameter müssen entsprechend der applikativen Anforderung passend eingestellt werden.

Im weiteren Verlauf muss entschieden werden, ob die Achse zeitgeführt oder weggeführt positioniert werden soll.

Die meisten hydraulischen Anwendungen können weggesteuert betrieben werden. Wenn dennoch die zeitgeführte Profilerzeugung notwendig ist, muss der Haken **TimeBased** gesetzt werden.

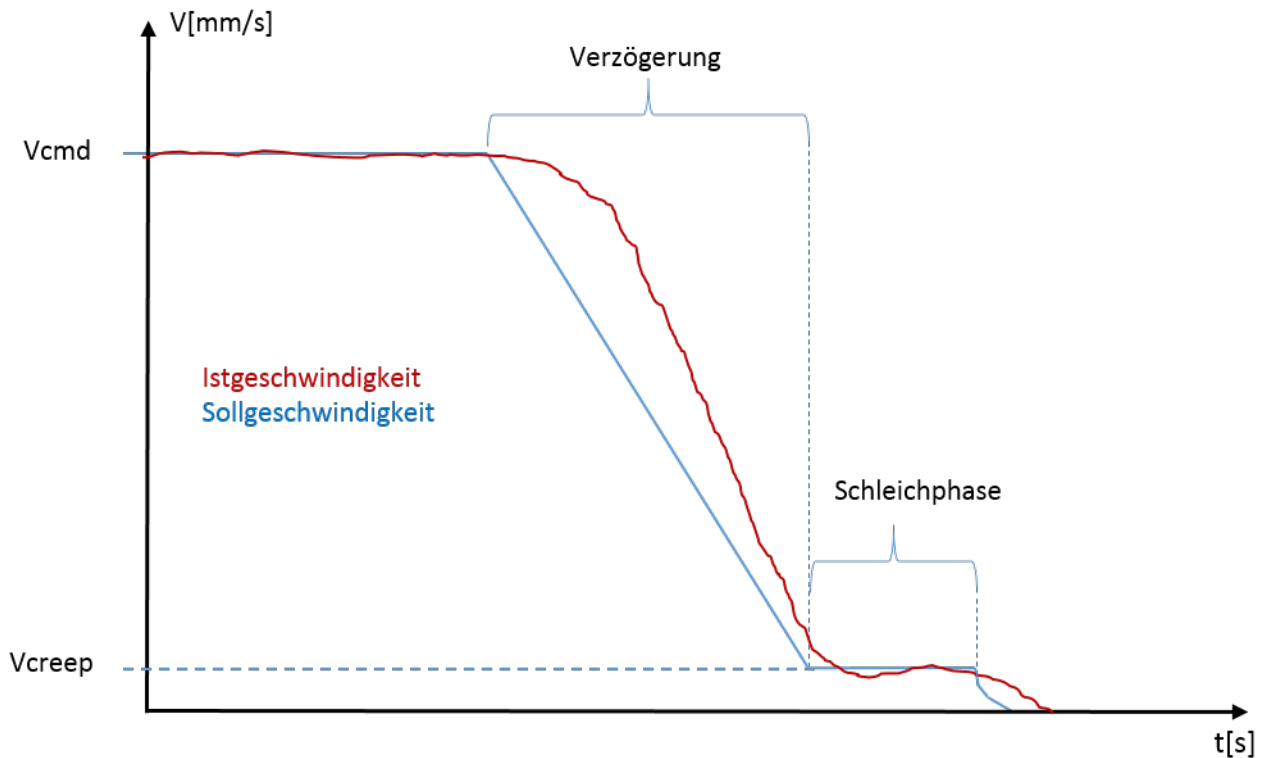
### 5.8.9.1 Weggeführte Achse

Der Lageregler ist nur für die Zielfahrt aktiv.

Die Beschleunigung kann so steil eingestellt werden, dass die Achse beim Losfahren ohne größere Schläge sanft beschleunigt.

Beim Abbremsen zum Ziel ist nicht nur die Verzögerung, sondern auch noch die Schleichstrecke, die Schleichgeschwindigkeit und die Bremsstrecke einzustellen. Alle drei Parameter hängen voneinander ab und beeinflussen die Zielfahrt. Ist die Achse innerhalb der Bremsstrecke, wird diese nur noch vom Lageregler kontrolliert. Schleichgeschwindigkeit und Schleichstrecke dienen dazu, die Achse nach dem Verzögern zu beruhigen, um diese anschließend über den Lageregler ins Ziel zu bringen.

Die Zielfahrt sollte wie folgt aussehen:



Häufig ist zu beobachten, dass eine Achse, die extrem stark verzögert wird, eine längere Schleichphase benötigt, um vergleichbar exakt zu positionieren wie eine Achse mit schwächerer Verzögerung.

### 5.8.9.2 Zeitgeführte Achse

Soll die Achse zeitgeführt werden ist der Lageregler während der gesamten Bewegung aktiv. Diese Option sollte nur bei Achsen mit hoher Eigenfrequenz und optimaler Weise einem Nullschnittventil genutzt werden.

Die Beschleunigung muss auf Werte begrenzt werden, denen die Achse ohne extreme Schwingneigung folgen kann. Hier sollte gerade auf das Anfahren ein spezielles Augenmerk gelegt werden.

Beim Bremsen ist die Verzögerung so einzustellen, dass die Achse der Sollwerttrampe folgen kann.

Schleichgeschwindigkeit, Schleichstrecke und Bremsstrecke können auf Null gestellt werden. Die Istposition muss zwingend der Sollposition hinterherlaufen um ein Überschwingen zu vermeiden. Ist dies nicht der Fall, muss die Vorsteuerung reduziert werden.

An diesem Punkt ist die Achse zur Positionierung vollständig in Betrieb genommen. Ist in der Applikation ein Druckregler, eine Kurvenscheibe oder eine Getriebekopplung, so müssen diese Elemente ebenfalls in Betrieb genommen werden.

## 6 Support und Service

Beckhoff und seine weltweiten Partnerfirmen bieten einen umfassenden Support und Service, der eine schnelle und kompetente Unterstützung bei allen Fragen zu Beckhoff Produkten und Systemlösungen zur Verfügung stellt.

### Downloadfinder

Unser [Downloadfinder](#) beinhaltet alle Dateien, die wir Ihnen zum Herunterladen anbieten. Sie finden dort Applikationsberichte, technische Dokumentationen, technische Zeichnungen, Konfigurationsdateien und vieles mehr.

Die Downloads sind in verschiedenen Formaten erhältlich.

### Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den [lokalen Support und Service](#) zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unserer Internetseite: [www.beckhoff.com](http://www.beckhoff.com)

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

### Beckhoff Support

Der Support bietet Ihnen einen umfangreichen technischen Support, der Sie nicht nur bei dem Einsatz einzelner Beckhoff Produkte, sondern auch bei weiteren umfassenden Dienstleistungen unterstützt:

- Support
- Planung, Programmierung und Inbetriebnahme komplexer Automatisierungssysteme
- umfangreiches Schulungsprogramm für Beckhoff Systemkomponenten

Hotline: +49 5246 963-157  
E-Mail: [support@beckhoff.com](mailto:support@beckhoff.com)

### Beckhoff Service

Das Beckhoff Service-Center unterstützt Sie rund um den After-Sales-Service:

- Vor-Ort-Service
- Reparaturservice
- Ersatzteilservice
- Hotline-Service

Hotline: +49 5246 963-460  
E-Mail: [service@beckhoff.com](mailto:service@beckhoff.com)

### Beckhoff Unternehmenszentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20  
33415 Verl  
Deutschland

Telefon: +49 5246 963-0  
E-Mail: [info@beckhoff.com](mailto:info@beckhoff.com)  
Internet: [www.beckhoff.com](http://www.beckhoff.com)





Mehr Informationen:  
**[www.beckhoff.de/ts5810](http://www.beckhoff.de/ts5810)**

Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Deutschland  
Telefon: +49 5246 9630  
[info@beckhoff.com](mailto:info@beckhoff.com)  
[www.beckhoff.com](http://www.beckhoff.com)

