

# YASKAWA Sigma-7 Library

YMC-LIB\_PN | YMC-LIB\_Sigma7-PN V2.0 | Handbuch

HB00 | YMC-LIB\_PN | YMC-LIB\_Sigma7-PN V2.0 | de | 19-40

Baustein Bibliothek - Sigma-7 - YASKAWA Motion Control PROFINET



YASKAWA Europe GmbH  
Hauptstr. 185  
65760 Eschborn  
Tel.: +49 6196 569-300  
Fax: +49 6196 569-398  
E-Mail: [info@yaskawa.eu.com](mailto:info@yaskawa.eu.com)  
Internet: [www.yaskawa.eu.com](http://www.yaskawa.eu.com)

## Inhaltsverzeichnis

<b>1</b>	<b>Allgemeines</b> .....	<b>5</b>
1.1	Copyright © YASKAWA Europe GmbH.....	5
1.2	Über dieses Handbuch.....	6
<b>2</b>	<b>Bibliothek einbinden</b> .....	<b>7</b>
2.1	Einbinden in Siemens SIMATIC Manager.....	8
2.2	Einbinden in Siemens TIA Portal.....	9
<b>3</b>	<b>Einsatz <i>Sigma-7</i> PROFINET</b> .....	<b>10</b>
3.1	Übersicht.....	10
3.2	Parameter am Antrieb einstellen.....	10
3.3	Einsatz im Siemens SIMATIC Manager.....	14
3.3.1	Voraussetzung.....	14
3.3.2	Hardware-Konfiguration CPU.....	14
3.3.3	Anwender-Programm.....	20
3.4	Einsatz im Siemens TIA Portal.....	24
3.4.1	Voraussetzung.....	24
3.4.2	Hardware-Konfiguration CPU.....	24
3.4.3	Anwender-Programm.....	32
<b>4</b>	<b>Bausteine zur Achskontrolle</b> .....	<b>37</b>
4.1	Übersicht.....	37
4.2	Antriebsspezifische Bausteine.....	38
4.2.1	UDT 862 - Y_SIG7PN_AXIS_CFG - Datenstruktur Achskonfiguration.....	38
4.2.2	UDT 850 ... UDT 859 - intern verwendete Datenstrukturen.....	38
4.2.3	FB 841 - Y_ServoFunction - Systemfunktionen.....	39
4.2.4	FB 847 - Y_ReadSafeState - Safety-Status lesen.....	40
4.2.5	FB 849 - Y_Init - Achskonfiguration.....	41
4.2.6	FB 862 - Y_SIG7PN_Kernel - Kernel.....	45
4.2.7	FB 863 - Y_SIG7PN_DeviceDriver - Diagnose intern.....	45
4.2.8	FB 865 - Y_SIG7PN_ServoInit - Initialisierung intern.....	45
4.2.9	FB 866 - Y_SIG7PN_ServoOrder - Auftragsinitialisierung intern.....	45
4.3	Komplexe Bewegungsaufgaben - PLCopen-Bausteine.....	46
4.3.1	UDT 860 - Y_SIG7PN_AXIS_REF - Datenstruktur Achsdaten.....	46
4.3.2	UDT 861 - MC_TRIGGER_REF - Datenstruktur Triggersignal.....	46
4.3.3	FB 800 - MC_Power - Achsenfreigabe.....	46
4.3.4	FB 801 - MC_Home - Achse referenzieren.....	48
4.3.5	FB 802 - MC_Stop - Achse stoppen.....	50
4.3.6	FB 803 - MC_Halt - Achse anhalten.....	53
4.3.7	FB 804 - MC_MoveRelative - Achse relativ verfahren.....	55
4.3.8	FB 805 - MC_MoveVelocity - Achse verfahren mit konstanter Geschwindigkeit.....	57
4.3.9	FB 808 - MC_MoveAbsolute - Achse auf absolute Position verfahren.....	59
4.3.10	FB 811 - MC_Reset - Reset Achse.....	61
4.3.11	FB 812 - MC_ReadStatus - Status Achse lesen.....	63
4.3.12	FB 813 - MC_ReadAxisError - Fehler von Achse lesen.....	65
4.3.13	FB 816 - MC_ReadActualPosition - Aktuelle Position der Achse lesen....	67
4.3.14	FB 817 - MC_ReadActualVelocity - Aktuelle Geschwindigkeit der Achse lesen.....	69
4.3.15	FB 818 - MC_ReadAxisInfo - Zusatzinformationen der Achse lesen.....	71
4.3.16	FB 819 - MC_ReadMotionState - Zustand Bewegungsauftrag lesen.....	73

4.3.17	FB 823 - MC_TouchProbe - Achsposition erfassen.....	75
4.3.18	FB 824 - MC_AbortTrigger - Achsposition erfassen abbrechen.....	77
4.3.19	FB 833 - Y_ReadParameter - Antriebsparameter lesen.....	78
4.3.20	FB 834 - Y_WriteParameter - Antriebsparameter schreiben.....	80
4.3.21	FB 835 - Y_HomeInit_LimitSwitch - Initialisierung Referenzfahrt auf Endschalter.....	82
4.3.22	FB 836 - Y_HomeInit_HomeSwitch - Initialisierung Referenzfahrt auf Referenzschalter.....	84
4.3.23	FB 837 - Y_HomeInit_ZeroPulse - Initialisierung Referenzfahrt auf Null Impuls.....	86
4.3.24	FB 838 - Y_HomeInit_SetPosition - Initialisierung Referenzfahrt auf aktuelle Position.....	88
4.3.25	FB 839 - MC_TorqueControl - Achse mit konstantem Drehmoment verfahren.....	89
4.3.26	FB 840 - MC_ReadActualTorque - Aktuelles Drehmoment lesen.....	91
<b>5</b>	<b>Zustände und Verhalten der Ausgänge.....</b>	<b>92</b>
5.1	PLCopen-States.....	92
5.2	Verhalten der Ein- und Ausgänge.....	94
<b>6</b>	<b>ErrorID - Zusätzliche Fehlerinformationen.....</b>	<b>96</b>
	<b>Anhang.....</b>	<b>100</b>
A	Änderungshistorie.....	102

# 1 Allgemeines

## 1.1 Copyright © YASKAWA Europe GmbH

### All Rights Reserved

Dieses Dokument enthält geschützte Informationen von YASKAWA und darf außer in Übereinstimmung mit anwendbaren Vereinbarungen weder offengelegt noch benutzt werden.

Dieses Material ist durch Urheberrechtsgesetze geschützt. Ohne schriftliches Einverständnis von YASKAWA und dem Besitzer dieses Materials darf dieses Material weder reproduziert, verteilt, noch in keiner Form von keiner Einheit (sowohl YASKAWA-intern als auch -extern) geändert werden, es sei denn in Übereinstimmung mit anwendbaren Vereinbarungen, Verträgen oder Lizenzen.

Zur Genehmigung von Vervielfältigung oder Verteilung wenden Sie sich bitte an:  
YASKAWA Europe GmbH, European Headquarters, Hauptstraße 185, 65760 Eschborn, Germany

Tel.: +49 6196 569 300

Fax.: +49 6196 569 398

E-Mail: [info@yaskawa.eu.com](mailto:info@yaskawa.eu.com)

Internet: [www.yaskawa.eu.com](http://www.yaskawa.eu.com)



*Es wurden alle Anstrengungen unternommen, um sicherzustellen, dass die in diesem Dokument enthaltenen Informationen zum Zeitpunkt der Veröffentlichung vollständig und richtig sind. Das Recht auf Änderungen der Informationen bleibt jedoch vorbehalten.*

*Die vorliegende Kundendokumentation beschreibt alle heute bekannten Hardware-Einheiten und Funktionen. Es ist möglich, dass Einheiten beschrieben sind, die beim Kunden nicht vorhanden sind. Der genaue Lieferumfang ist im jeweiligen Kaufvertrag beschrieben.*

### EG-Konformitätserklärung

Hiermit erklärt YASKAWA Europe GmbH, dass die Produkte und Systeme mit den grundlegenden Anforderungen und den anderen relevanten Vorschriften übereinstimmen. Die Übereinstimmung ist durch CE-Zeichen gekennzeichnet.

### Informationen zur Konformitätserklärung

Für weitere Informationen zur CE-Kennzeichnung und Konformitätserklärung wenden Sie sich bitte an Ihre Landesvertretung der YASKAWA Europe GmbH.

### Warenzeichen

VIPA, SLIO, System 100V, System 200V, System 300V, System 300S, System 400V, System 500S und Commander Compact sind eingetragene Warenzeichen der YASKAWA Europe GmbH.

SPEED7 ist ein eingetragenes Warenzeichen der YASKAWA Europe GmbH.

SIMATIC, STEP, SINEC, TIA Portal, S7-300, S7-400 und S7-1500 sind eingetragene Warenzeichen der Siemens AG.

Microsoft und Windows sind eingetragene Warenzeichen von Microsoft Inc., USA.

Portable Document Format (PDF) und Postscript sind eingetragene Warenzeichen von Adobe Systems, Inc.

Alle anderen erwähnten Firmennamen und Logos sowie Marken- oder Produktnamen sind Warenzeichen oder eingetragene Warenzeichen ihrer jeweiligen Eigentümer.

**Dokument-Support**

Wenden Sie sich an Ihre Landesvertretung der YASKAWA Europe GmbH, wenn Sie Fehler anzeigen oder inhaltliche Fragen zu diesem Dokument stellen möchten. Ist eine solche Stelle nicht erreichbar, können Sie YASKAWA Europe GmbH über folgenden Kontakt erreichen:

YASKAWA Europe GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Fax: +49 9132 744 29 1204

E-Mail: Documentation.HER@yaskawa.eu.com

**Technischer Support**

Wenden Sie sich an Ihre Landesvertretung der YASKAWA Europe GmbH, wenn Sie Probleme mit dem Produkt haben oder Fragen zum Produkt stellen möchten. Ist eine solche Stelle nicht erreichbar, können Sie den YASKAWA Kundenservice über folgenden Kontakt erreichen:

YASKAWA Europe GmbH,  
European Headquarters, Hauptstraße 185, 65760 Eschborn, Germany

Tel.: +49 6196 569 500 (Hotline)

E-Mail: support@yaskawa.eu.com

## 1.2 Über dieses Handbuch

**Zielsetzung und Inhalt**

Das Handbuch beschreibt die Baustein-Bibliothek *"Sigma-7 - YASKAWA Motion Control PROFINET"* von YASKAWA:

- Beschrieben wird Aufbau, Projektierung und Anwendung in verschiedenen Programmiersystemen.
- Das Handbuch ist geschrieben für Anwender mit Grundkenntnissen in der Automatisierungstechnik.
- Das Handbuch ist in elektronischer Form als PDF-Datei verfügbar. Hierzu ist der Adobe Acrobat Reader erforderlich.
- Das Handbuch ist in Kapitel gegliedert. Jedes Kapitel beschreibt eine abgeschlossene Thematik.
- Als Orientierungshilfe stehen im Handbuch zur Verfügung:
  - Gesamt-Inhaltsverzeichnis am Anfang des Handbuchs
  - Verweise mit Seitenangabe

**Piktogramme Signalwörter**

Besonders wichtige Textteile sind mit folgenden Piktogrammen und Signalworten ausgezeichnet:

**GEFAHR!**

Unmittelbar drohende oder mögliche Gefahr. Personenschäden sind möglich.

**VORSICHT!**

Bei Nichtbefolgen sind Sachschäden möglich.



*Zusätzliche Informationen und nützliche Tipps.*

## 2 Bibliothek einbinden

### Baustein-Bibliothek

Die Baustein-Bibliothek finden Sie im "Service/Support"-Bereich auf der entsprechenden Webseite zum Download. Die Bibliothek liegt als gepackte zip-Dateien vor. Sobald Sie die Bausteine verwenden möchten, müssen Sie diese in Ihr Projekt importieren.

Über folgende Webseiten haben Sie Zugriff auf die Bibliotheken:

- <https://www.yaskawa.eu.com/en/service/drives-motion-software-download>
- <http://www.vipa.com/de/service-support/downloads/yaskawa-lib>



*Bitte verwenden Sie immer das zu Ihrer Bibliothek zugehörige Handbuch. Solange es keine beschreibungsrelevante Änderungen gibt, können im Handbuch die Versionsangaben der Bibliothek und der zugehörigen Dateien von denen der Bibliothek abweichen.*




### Folgende Bausteinbibliotheken stehen zur Verfügung

Datei	Beschreibung
YMC-LIB_Sigma7-PN_S7_V0001.zip	<ul style="list-style-type: none"> <li>■ Bausteinbibliothek für Siemens SIMATIC Manager.</li> <li>■ Für den Einsatz in VIPA-CPU's bzw. S7-300 CPU's von Siemens.</li> </ul>
YMC-LIB_Sigma7-PN_TIA_V0001.zip	<ul style="list-style-type: none"> <li>■ Bausteinbibliothek für Siemens TIA Portal V15.</li> <li>■ Für den Einsatz in VIPA-CPU's bzw. S7-300 CPU's von Siemens.</li> </ul>


## 2.1 Einbinden in Siemens SIMATIC Manager

### Übersicht






Die Einbindung in den Siemens SIMATIC Manager erfolgt nach folgenden Schritten:

1.  ZIP-Datei laden
2.  Bibliothek "dearchivieren"
3.  Bibliothek öffnen und Bausteine in Projekt übertragen



### ZIP-Datei laden

-  Navigieren Sie auf der Webseite zu der gewünschten ZIP-Datei, laden und speichern Sie diese in Ihrem Arbeitsverzeichnis.

### Bibliothek dearchivieren

1.  Starten Sie den Siemens SIMATIC Manager mit Ihrem Projekt.
2.  Öffnen Sie mit "*Datei* → *Dearchivieren*" das Dialogfenster zur Auswahl der ZIP-Datei.
3.  Wählen Sie die entsprechende ZIP-Datei an und klicken Sie auf [Öffnen].
4.  Geben Sie ein Zielverzeichnis an, in dem die Bausteine abzulegen sind.
5.  Starten Sie den Entpackvorgang mit [OK].

### Bibliothek öffnen und Bausteine in Projekt übertragen

1.  Öffnen Sie die Bibliothek nach dem Entpackvorgang.
2.  Öffnen Sie Ihr Projekt und kopieren Sie die erforderlichen Bausteine aus der Bibliothek in das Verzeichnis "Bausteine" Ihres Projekts.  
⇒ Nun haben Sie in Ihrem Anwenderprogramm Zugriff auf die Bausteine.



*Werden anstelle der SFCs FCs verwendet, so werden diese von den System 300S VIPA-CPU's ab Firmware 3.6.0 unterstützt.*



## 2.2 Einbinden in Siemens TIA Portal

### Übersicht

Die Einbindung in das Siemens TIA Portal erfolgt nach folgenden Schritten:

1. ➤ ZIP-Datei laden
2. ➤ ZIP-Datei entpacken
3. ➤ Bibliothek "dearchivieren"
4. ➤ Bibliothek öffnen und Bausteine in Projekt übertragen

### ZIP-Datei laden

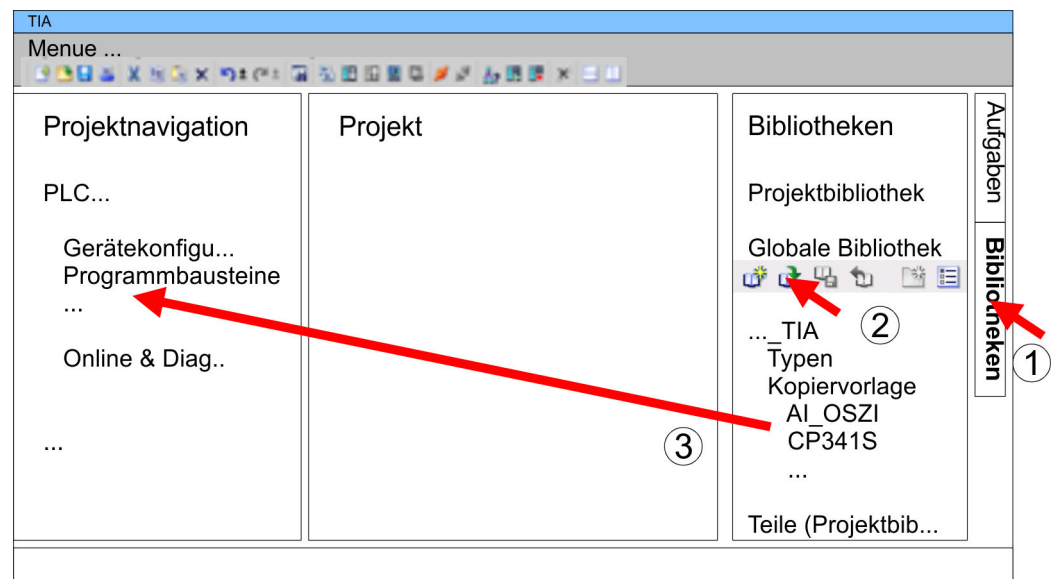
1. ➤ Navigieren Sie auf der Webseite zu der ZIP-Datei, welche zu Ihrer Programmversion passt.
2. ➤ Laden und speichern Sie diese in Ihrem Arbeitsverzeichnis.

### ZIP-Datei entpacken

- Entpacken Sie die ZIP-Datei mit Ihrem Entpackprogramm in ein Arbeitsverzeichnis für das Siemens TIA Portal.

### Bibliothek öffnen und Bausteine in Projekt übertragen

1. ➤ Starten Sie das Siemens TIA Portal mit Ihrem Projekt.
2. ➤ Wechseln sie in die *Projektansicht*.
3. ➤ Wählen Sie auf der rechten Seite die Task-Card "Bibliotheken".
4. ➤ Klicken Sie auf "Globale Bibliothek".
5. ➤ Klicken Sie auf "Globale Bibliothek öffnen".
6. ➤ Navigieren Sie zu ihrem Arbeitsverzeichnis und laden Sie die Datei ...\_TIA.al1x.



7. ➤ Kopieren Sie die erforderlichen Bausteine aus der Bibliothek in das Verzeichnis "Programmbausteine" in der *Projektnavigation* Ihres Projekts. Nun haben Sie in Ihrem Anwenderprogramm Zugriff auf die Bausteine.

## 3 Einsatz *Sigma-7* PROFINET

### 3.1 Übersicht

#### Voraussetzung

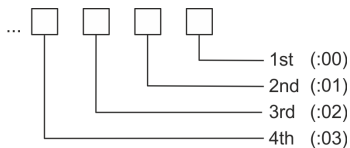
- Siemens SIMATIC Manager ab V 5.5 SP2 & *YMC-LIB\_Sigma7-PN Library* oder Siemens TIA Portal V 15.1 & *YMC-LIB\_Sigma7-PN Library*.
- CPU mit PROFINET-IO-Controller wie z.B. CPU 017-CEFPR00 mit Erweiterung des Arbeitsspeichers auf 2MB.
- *Sigma-7*-Antrieb mit PROFINET-Anbindung.

#### Schritte der Projektierung

1. ➤ Parameter am Antrieb einstellen ☞ 10
  - Die Einstellung der Parameter hat mit dem Softwaretool *SigmaWin+* bzw. FB 834 - *Y\_WriteParameter* ☞ 80 zu erfolgen.
2. ➤ Hardwarekonfiguration im Siemens SIMATIC Manager ☞ 14 oder Hardwarekonfiguration im Siemens TIA Portal ☞ 24
  - Projektierung einer CPU mit PROFINET-IO-Controller-Funktionalität.
  - Projektierung eines *Sigma-7* PROFINET-Antriebs.
  - Projektierung der PROFINET-Anbindung über die Hardware-Konfiguration.
3. ➤ Programmierung im Siemens SIMATIC Manager ☞ 20 oder Programmierung im Siemens TIA Portal ☞ 32
  - *Init*-Baustein zur Konfiguration der Achse beschalten.
  - *Kernel*-Baustein zur Kommunikation mit der Achse beschalten.
  - Bausteine für die Bewegungsabläufe beschalten.

### 3.2 Parameter am Antrieb einstellen

#### Parameter-Digits



#### VORSICHT!

Vor der Erstinbetriebnahme müssen Sie Ihren Antrieb an Ihre Applikation anpassen! Näheres hierzu finden Sie im Handbuch zu ihrem Antrieb.

Folgende Parameter sind mittels *SigmaWin+* bzw. FB 834 - *Y\_WriteParameter* ☞ 80 entsprechend einzustellen:

## Sigma-7

SERVOPACK Parameter	Parameter.Digit	Beschreibung	Default Wert
Pn001	Pn001.0	Servo OFF or alarm group 1 stopping method <ul style="list-style-type: none"> <li>■ 0: Dynamic break               <ul style="list-style-type: none"> <li>– stops the motor by applying the dynamic brake</li> </ul> </li> <li>■ 1: Dynamic break and release               <ul style="list-style-type: none"> <li>– coast the motor to a stop without the dynamic brake</li> </ul> </li> <li>■ 2: No dynamic break</li> </ul>	0
Pn002	Pn002.2	Encoder Usage <ul style="list-style-type: none"> <li>■ 0: Use the encoder according to the encoder specifications.</li> <li>■ 1: Use the encoder as an incremental encoder (currently not supported).</li> <li>■ 2: Use the encoder as a single-turn absolute encoder (currently not supported).</li> </ul>	0
Pn00B	Pn00B.2	Power input selection for three-phase SERVOPACK (Sigma-7 200V) <ul style="list-style-type: none"> <li>■ 0: Use a 3-phase power supply input</li> <li>■ 1: Use a 3-phase power supply input as single-phase power supply input</li> </ul> <p>This parameter is relevant for SGD7S-xxxAC0 SERVOPACK only. Do not change in case of SGD7S-xxxDC0 SERVOPACK is used.</p>	0
Pn50A	Pn50A.3	P-OT (forward drive prohibit) signal allocation <ul style="list-style-type: none"> <li>■ 0: Enable forward drive when CN1...13 input signal ON (closed)</li> <li>■ 1: Enable forward drive when CN1...7 input signal ON (closed)</li> <li>■ 2: Enable forward drive when CN1...8 input signal ON (closed)</li> <li>■ 3: Enable forward drive when CN1...9 input signal ON (closed)</li> <li>■ 4: Enable forward drive when CN1...10 input signal ON (closed)</li> <li>■ 5: Enable forward drive when CN1...11 input signal ON (closed)</li> <li>■ 6: Enable forward drive when CN1...12 input signal ON (closed)</li> <li>■ 7: Set the signal to always prohibit forward drive</li> <li>■ 8: Set the signal to always enable forward drive.</li> <li>■ 9: Enable forward drive when CN1...13 input signal OFF (open)</li> <li>■ A: Enable forward drive when CN1...7 input signal OFF (open)</li> <li>■ B: Enable forward drive when CN1...8 input signal OFF (open)</li> <li>■ C: Enable forward drive when CN1...9 input signal OFF (open)</li> <li>■ D: Enable forward drive when CN1...10 input signal OFF (open)</li> <li>■ E: Enable forward drive when CN1...11 input signal OFF (open)</li> <li>■ F: Enable forward drive when CN1...12 input signal OFF (open)</li> </ul>	1

## Parameter am Antrieb einstellen

SERVOPACK Parameter	Parameter.Digit	Beschreibung	Default Wert
Pn50B	Pn50B.0	<p>N-OT (reverse drive prohibit) signal allocation</p> <ul style="list-style-type: none"> <li>■ 0: Enable reverse drive when CN1...13 input signal ON (closed)</li> <li>■ 1: Enable reverse drive when CN1...7 input signal ON (closed)</li> <li>■ 2: Enable reverse drive when CN1...8 input signal ON (closed)</li> <li>■ 3: Enable reverse drive when CN1...9 input signal ON (closed)</li> <li>■ 4: Enable reverse drive when CN1...10 input signal ON (closed)</li> <li>■ 5: Enable reverse drive when CN1...11 input signal ON (closed)</li> <li>■ 6: Enable reverse drive when CN1...12 input signal ON (closed)</li> <li>■ 7: Set the signal to always prohibit reverse drive</li> <li>■ 8: Set the signal to always prohibit reverse drive</li> <li>■ 9: Enable reverse drive when CN1...13 input signal OFF (open)</li> <li>■ A: Enable reverse drive when CN1...7 input signal OFF (open)</li> <li>■ B: Enable reverse drive when CN1...8 input signal OFF (open)</li> <li>■ C: Enable reverse drive when CN1...9 input signal OFF (open)</li> <li>■ D: Enable reverse drive when CN1...10 input signal OFF (open)</li> <li>■ E: Enable reverse drive when CN1...11 input signal OFF (open)</li> <li>■ F: Enable reverse drive when CN1...12 input signal OFF (open)</li> </ul>	2
Pn511	Pn511.0	<p>DEC (home switch input) signal allocation</p> <ul style="list-style-type: none"> <li>■ 0: Active when CN1...13 input signal ON (closed)</li> <li>■ 1: Active when CN1...7 input signal ON (closed)</li> <li>■ 2: Active when CN1...8 input signal ON (closed)</li> <li>■ 3: Active when CN1...9 input signal ON (closed)</li> <li>■ 4: Active when CN1...10 input signal ON (closed)</li> <li>■ 5: Active when CN1...11 input signal ON (closed)</li> <li>■ 6: Active when CN1...12 input signal ON (closed)</li> <li>■ 7: The signal is always active</li> <li>■ 8: The signal is always inactive</li> <li>■ 9: Active when CN1...13 input signal OFF (open)</li> <li>■ A: Active when CN1...7 input signal OFF (open)</li> <li>■ B: Active when CN1...8 input signal OFF (open)</li> <li>■ C: Active when CN1...9 input signal OFF (open)</li> <li>■ D: Active when CN1...10 input signal OFF (open)</li> <li>■ E: Active when CN1...11 input signal OFF (open)</li> <li>■ F: Active when CN1...12 input signal OFF (open)</li> </ul>	3

SERVOPACK Parameter	Parameter.Digit	Beschreibung	Default Wert
Pn511	Pn511.1	EXT1 (probe 1 latch input) signal allocation <ul style="list-style-type: none"> <li>■ 0 ... 3: The signal is always inactive</li> <li>■ 4: Active when CN1...10 input signal ON (closed)</li> <li>■ 5: Active when CN1...11 input signal ON (closed)</li> <li>■ 6: Active when CN1...12 input signal ON (closed)</li> <li>■ 7 ... C: The signal is always inactive</li> <li>■ D: Active when CN1...10 input signal OFF (open)</li> <li>■ E: Active when CN1...11 input signal OFF (open)</li> <li>■ F: Active when CN1...12 input signal OFF (open)</li> </ul>	4
Pn511	Pn511.2	EXT2 (probe 2 latch input) signal allocation <ul style="list-style-type: none"> <li>■ 0 ... 3: The signal is always inactive</li> <li>■ 4: Active when CN1...10 input signal ON (closed)</li> <li>■ 5: Active when CN1...11 input signal ON (closed)</li> <li>■ 6: Active when CN1...12 input signal ON (closed)</li> <li>■ 7 ... C: The signal is always inactive</li> <li>■ D: Active when CN1...10 input signal OFF (open)</li> <li>■ E: Active when CN1...11 input signal OFF (open)</li> <li>■ F: Active when CN1...12 input signal OFF (open)</li> </ul>	5



### **Bitte folgende Parameter nicht ändern**

Beim Aufruf des Init-Bausteines Y\_SIG7PN\_Servolnit werden folgende Parameter angepasst. Diese sollten Sie nicht ändern:

- PnC00 ... PnC0F - Setpoint telegram: PZD 1 ... 16
- PnC10 ... PnC1F - Actual value telegram: PZD 1 ... 16
- PnC20 - Telegram selection
- PnB02 - Position user unit: Numerator
- PnB04 - Position user unit: Denominator
- PnB06 - Velocity user unit: Numerator
- PnB08 - Velocity user unit: Denominator
- PnB0A - Acceleration user unit: Numerator
- PnB42 - Position range limit (min.)
- PnB44 - Position range limit (max.)
- PnB48 - Software position limit (min.)
- PnB4A - Software position limit (max.)
- PnB0C - Acceleration user unit: Denominator
- PnB4C (607Fh): Max. profile velocity (Default: Bezug auf PnBF0 (2312h): Max. motor speed)
- PnB7C (60C5h): Max. acceleration (Default: Bezug auf PnBF2 (2313h): Max. motor acceleration)
- PnB7E (60C6h): Max. deceleration (Default: Bezug auf PnBF2 (2313h): Max. motor acceleration)
- Pn205 - Multiturn limit

### 3.3 Einsatz im Siemens SIMATIC Manager

#### 3.3.1 Voraussetzung

##### Übersicht

- Bitte verwenden Sie für die Projektierung den Siemens SIMATIC Manager ab V 5.5 SP2.
- Die Bausteine können Sie bei folgenden CPUs einsetzen:
  - System SLIO CPU 017-CEFPR00 mit Erweiterung des Arbeitsspeichers auf 2MB
  - System 300S CPU 315-4PN43 mit Erweiterung des Arbeitsspeichers auf 1MB
  - System 300S CPU 315-4PN23
  - System 300S CPU 317-4PN23
- Die Projektierung der System SLIO CPU erfolgt im Siemens SIMATIC Manager in Form des virtuellen PROFINET IO Devices "*VIPA SLIO CPU*". Das "*VIPA SLIO System*" ist mittels GSDML im Hardware-Katalog zu installieren.
- Die Projektierung der System 300S CPUs 315-4PNxx erfolgt im Siemens SIMATIC Manager als Siemens CPU 315-2 PN/DP (6ES7 315-2EH14-0AB0 V3.2).
- Die Projektierung der System 300S CPU 317-4PN23 erfolgt im Siemens SIMATIC Manager als Siemens CPU 317-2 PN/DP (6ES7 317-2EK14-0AB0 V3.2).
- Die Projektierung des PROFINET-IO-Controller für die Antriebe erfolgt im Siemens SIMATIC Manager.

#### 3.3.2 Hardware-Konfiguration CPU

##### 3.3.2.1 Hardware-Konfiguration System SLIO CPU 017PN

###### IO Device "*VIPA SLIO System*" installieren

Die Projektierung der System SLIO CPU erfolgt in Form des virtuellen PROFINET IO Devices "*VIPA SLIO CPU*". Die Installation des PROFINET IO Devices "*VIPA SLIO CPU*" im Hardware-Katalog erfolgt nach folgender Vorgehensweise:

1. ➤ Gehen Sie in den Service-Bereich von [www.vipa.com](http://www.vipa.com).
2. ➤ Laden Sie aus dem Downloadbereich unter "*Config Dateien* ➔ *PROFINET*" die Konfigurationsdatei für Ihre CPU.
3. ➤ Extrahieren Sie die Datei in Ihr Arbeitsverzeichnis.
4. ➤ Starten Sie den Hardware-Konfigurator von Siemens.
5. ➤ Schließen Sie alle Projekte.
6. ➤ Gehen Sie auf "*Extras* ➔ *GSD-Dateien installieren*".
7. ➤ Navigieren Sie in Ihr Arbeitsverzeichnis und installieren Sie die entsprechende GSDML-Datei.

⇒ Nach der Installation finden Sie das entsprechende PROFINET IO Device unter "*PROFINET IO* ➔ *Weitere Feldgeräte* ➔ *I/O* ➔ *VIPA SLIO System*".

###### GSDML für YASKAWA *Sigma-7* Servo-Antrieb installieren

Im Lieferumfang des Beispielsprojekts befindet sich die GSDML-Datei für den Sigma-7 Servo-Antrieb.

Die Installation der GSDML für den *Sigma-7* Servo-Antrieb im Hardware-Katalog erfolgt nach folgender Vorgehensweise:

1. ➤ Extrahieren Sie aus Ihrem Beispielprojekt die GSDML-Datei in Ihr Arbeitsverzeichnis.
2. ➤ Starten Sie den Hardware-Konfigurator von Siemens.
3. ➤ Schließen Sie alle Projekte.
4. ➤ Gehen Sie auf "*Extras* ➔ *GSD-Dateien installieren*".

5. Navigieren Sie in Ihr Arbeitsverzeichnis und installieren Sie die entsprechende GSDML-Datei.
  - ⇒ Nach der Installation finden Sie das PROFINET-Device "SGD7..." unter "PROFINET IO → Weitere Feldgeräte → Drives → YASKAWA Drives".

### CPU im Projekt anlegen

Steckp..	Baugruppe
1	
<b>2</b>	<b>CPU 317-2PN/DP</b>
X1	MPI/DP
X2	PN-IO
X2...	Port 1
X2...	Port 2
3	

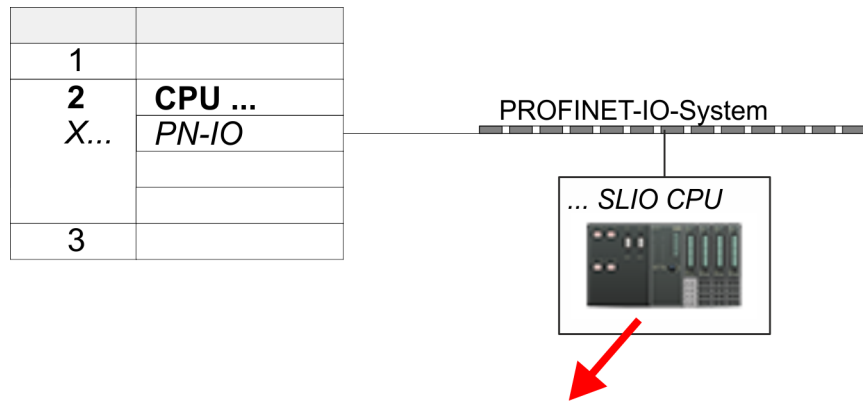
Um kompatibel mit dem Siemens SIMATIC Manager zu sein, sind folgende Schritte durchzuführen:

1. Starten Sie den Hardware-Konfigurator von Siemens mit einem neuen Projekt.
2. Fügen Sie aus dem Hardware-Katalog eine Profilschiene ein.
3. Platzieren Sie auf "Slot"-Nummer 2 die CPU 317-2PN/DP (6ES7 317-2EK14-0AB0 V3.2).
4. Klicken Sie auf das Submodul "PN-IO" der CPU.

Steckpl.	Baugruppe
1	
<b>2</b>	<b>CPU ...</b>
X...	PN-IO
3	

PROFINET-IO-System

5. Legen Sie mit [Neu] ein neues Subnetz an und vergeben Sie gültige IP-Adress-Daten für Ihr PROFINET-System.
6. Klicken Sie auf das Submodul "PN-IO" der CPU und öffnen Sie mit "Kontextmenü → Objekteigenschaften" den Eigenschafts-Dialog.
7. Geben Sie unter "Allgemein" einen "Gerätenamen" an. Der Geräte name muss eindeutig am Ethernet-Subnetz sein.



0	... SLIO CPU ...	...	
X2	...		
1			
2			
3			
...			


8. Navigieren Sie im Hardware-Katalog in das Verzeichnis "PROFINET IO → Weitere Feldgeräte → I/O → VIPA SLIO System" und binden das Ihrer CPU entsprechende IO-Device "017-CEFPR00 → FW V2.4" an Ihr PROFINET-System an.
  - ⇒ In der Steckplatzübersicht des PROFINET IO Device "VIPA SLIO CPU" ist auf Steckplatz 0 die entsprechende CPU bereits vorplatziert. Ab Steckplatz 1 können Sie Ihre System SLIO Module platzieren.

**Ethernet-PG/OP-Kanal parametrieren**

Die CPU hat einen Ethernet-PG/OP-Kanal integriert. Über diesen Kanal können Sie Ihre CPU programmieren und fernwarten.

Steckpl.	Modul
1	
2	CPU ...
X...	PN-IO
3	
4	343-1EX30
5	
...	

1. Platzieren Sie für den Ethernet-PG/OP-Kanal auf Steckplatz 4 den Siemens CP 343-1 (SIMATIC 300 \ CP 300 \ Industrial Ethernet \ CP 343-1 \ 6GK7 343-1EX30 0XE0 V3.0).
2. Öffnen Sie durch Doppelklick auf den CP 343-1EX30 den Eigenschaften-Dialog und geben Sie für den CP unter "Eigenschaften" IP-Adress-Daten an. Gültige IP-Adress-Parameter erhalten Sie von Ihrem Systemadministrator.
3. Ordnen Sie den CP einem "Subnetz" zu. Ohne Zuordnung werden die IP-Adress-Daten nicht übernommen!

 Näheres zum Einsatz des Ethernet-PG/OP-Kanals finden Sie im Handbuch zu Ihrer CPU.



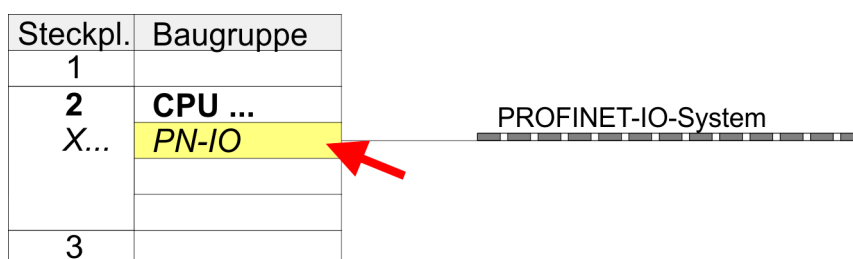
### 3.3.2.2 Hardware-Konfiguration System 300S CPU 315PN ... 317PN

#### CPU im Projekt anlegen

Steckp..	Baugruppe
1	
<b>2</b>	<b>CPU 315-2 PN/DP</b>
X1	MPI/DP
X2	PN-IO
X2...	Port 1
X2...	Port 2
...	...
3	

Um kompatibel mit dem Siemens SIMATIC Manager zu sein, sind folgende Schritte durchzuführen:

1. ➤ Starten Sie den Hardware-Konfigurator von Siemens mit einem neuen Projekt.
2. ➤ Fügen Sie aus dem Hardware-Katalog eine Profilschiene ein.
3. ➤ Platzieren Sie auf "Slot"-Nummer 2 für die CPU 315PN die Siemens CPU 315-2 PN/DP (6ES7 315-2EH14-0AB0 V3.2) und für die CPU 317PN die Siemens CPU 317-2 PN/DP (6ES7 317-2EK14-0AB0 V3.2).
4. ➤ Klicken Sie auf das Submodul "PN-IO" der CPU.



5. ➤ Legen Sie mit [Neu] ein neues Subnetz an und vergeben Sie gültige IP-Adress-Daten für Ihr PROFINET-System.
6. ➤ Klicken Sie auf das Submodul "PN-IO" der CPU und öffnen Sie mit "Kontextmenü" → "Objekteigenschaften" den Eigenschafts-Dialog.
7. ➤ Geben Sie unter "Allgemein" einen "Gerätenamen" an. Der Gerätename muss eindeutig am Ethernet-Subnetz sein.

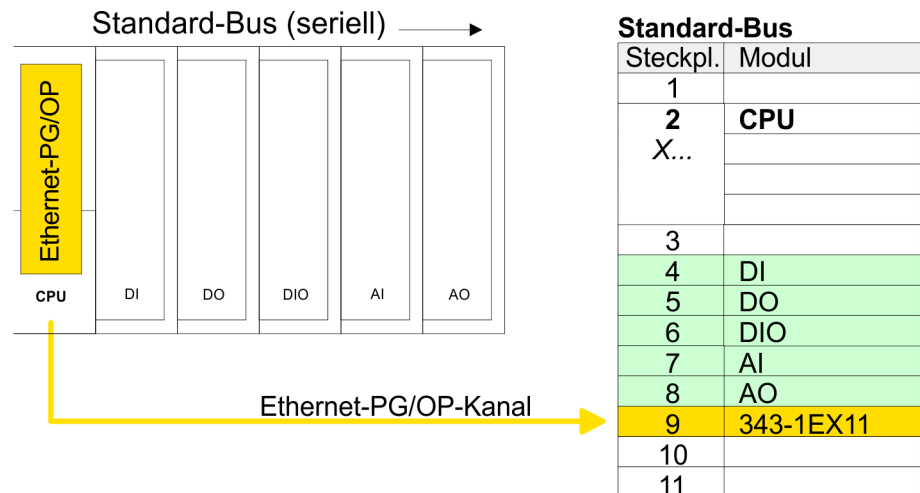
#### Ethernet-PG/OP-Kanal parametrieren

Die CPU hat einen Ethernet-PG/OP-Kanal integriert. Über diesen Kanal können Sie Ihre CPU programmieren und fernwarten.

1. ➤ Projektieren Sie die Module am Standard-Bus.
2. ➤ Für den Ethernet-PG/OP-Kanal ist immer unterhalb der reell gesteckten Module ein Siemens CP 343-1 (SIMATIC 300 \ CP 300 \ Industrial Ethernet \ CP 343-1 \ 6GK7 343-1EX11 0XE0) zu platzieren.
3. ➤ Öffnen Sie durch Doppelklick auf den CP 343-1EX11 den Eigenschaften-Dialog und geben Sie für den CP unter "Eigenschaften" die IP-Adress-Daten aus der Urtaufe an.
4. ➤ Ordnen Sie den CP einem "Subnetz" zu. Ohne Zuordnung werden die IP-Adress-Daten nicht übernommen!

**5.** Übertragen Sie Ihr Projekt in Ihre CPU

⇒ Die IP-Adress-Daten werden in Ihr Projekt übernommen.



Näheres zur Urtaufe und zum Einsatz des Ethernet-PG/OP-Kanals finden Sie im Handbuch zu Ihrer CPU.

**3.3.2.3** *Sigma-7* PROFINET Antrieb einfügen und konfigurieren***Sigma-7* PROFINET Antrieb einfügen und konfigurieren**

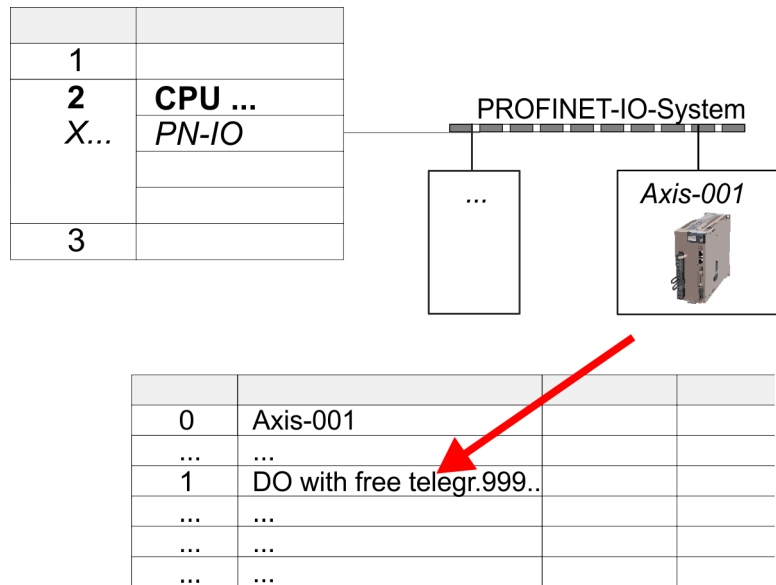
Die Konfiguration des Antriebs erfolgt im Siemens SIMATIC Manager. Hierbei ist für jede Achse ein *Sigma-7* PROFINET-Device zu projektieren.

- 1.** Wählen Sie aus dem Hardware-Katalog Ihren *Sigma-7* PROFINET Antrieb "*SGD7S-xxx...*" an und ziehen Sie diesen auf das "*PROFINET-IO-System*".
  - ⇒ Der *Sigma-7* PROFINET Antrieb wird an den IO-Controller angebunden und kann nun konfiguriert werden.
- 2.** Öffnen Sie die Objekteigenschaften des *Sigma-7* PROFINET Antriebs und vergeben Sie einen passenden "*Gerätenamen*" wie z.B. Axis-001.
- 3.** Wählen Sie aus dem Hardware-Katalog Ihren *Sigma-7* PROFINET Antrieb "*SGD7S-xxx...*" an und ziehen Sie das Element "*DO with free telegr.999...*" auf den Steckplatz 1 der Steckplatzübersicht des *Sigma-7* PROFINET Antriebs.

4. ➔ Notieren Sie sich die Diagnoseadresse von "DO with free telegr.999..." in der Steckplatzübersicht.

**i Bitte folgendes beachten!**

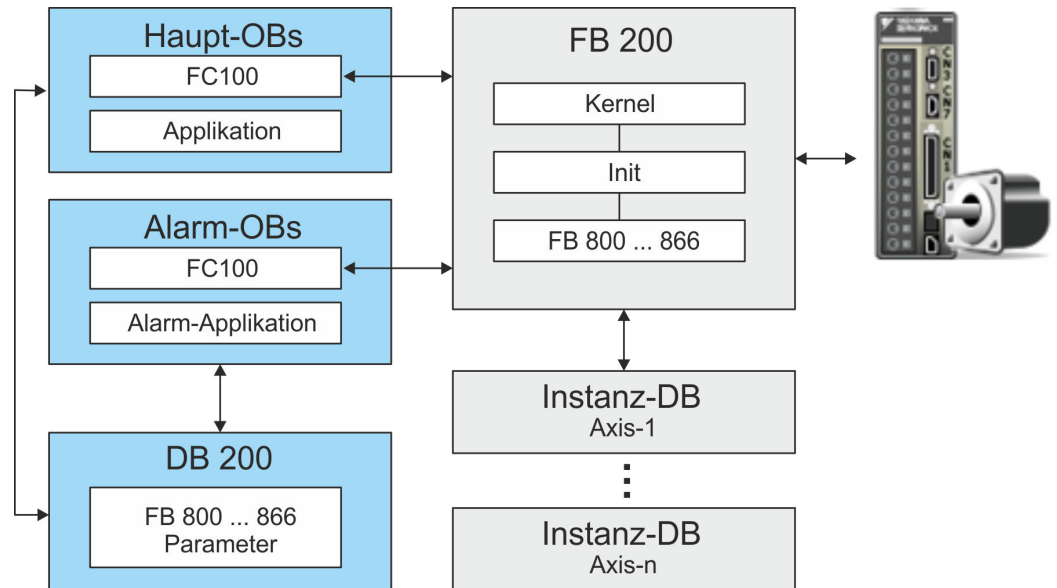
- Die Verknüpfung zwischen den Achsen in der Hardware-Konfiguration und ihrem Anwenderprogramm erfolgt über die jeweilige Diagnoseadresse des Submoduls "DO with free telegr.999..."
- Die Anfangsadressen von "E-Adresse" und "A-Adresse" des Submoduls "Free telegram PZD-16/16" müssen identisch sein.



5. ➔ Speichern, übersetzen Sie Ihre Konfiguration und übertragen Sie diese in die CPU.
6. ➔ Für die Namenszuweisung wählen Sie "Zielsystem ➔ Ethernet ➔ Ethernetteilnehmer bearbeiten". Klicken Sie hier auf "Durchsuchen".  
 ⇒ Die Ethernet-Teilnehmer werden aufgelistet.
7. ➔ Suchen Sie den Sigma7 PROFINET Antrieb "SGM7xxxACxx" und weisen Sie diesem den Namen aus der Hardware-Konfiguration zu.
8. ➔ Speichern, übersetzen Sie Ihre Konfiguration und übertragen Sie diese in die CPU.

### 3.3.3 Anwender-Programm

#### 3.3.3.1 Programmstruktur



- **Init**
  - Die *Init*-Bausteine dienen zur Konfiguration der Achsen.
  - Die *Init*-Bausteine werden innerhalb des FB 200 zyklisch aufgerufen und bei Bedarf abgearbeitet.
  - Spezifischer Baustein für *Sigma-7* PROFINET.
  - Die Konfiguration für die Initialisierung erfolgt über den entsprechenden Instanz-DB.
- **Kernel**
  - Die *Kernel*-Bausteine werden innerhalb des FB 200 zyklisch aufgerufen.
  - Die *Kernel*-Bausteine kommunizieren mit dem Antrieb über PROFINET, verarbeiten die Benutzeraufträge und liefern Statusmeldungen zurück.
  - Der Austausch der Daten erfolgt mittels des entsprechenden Instanz-DB.
- **FB 200**
  - Innerhalb des FB 200 werden alle Funktionsbausteine der Bibliothek zyklisch aufgerufen und bei Bedarf abgearbeitet.
  - Als Instanz-DB dient der beim Aufruf anzugebende DB der entsprechenden Achse.
  - Als Referenz zur Achse dient die Diagnoseadresse, welche in der Hardware-Konfiguration für die entsprechende Achse vorzugeben ist.
- **Instanz-DB - Achse-DB**
  - Für den FB 200 ist für jede Achse ein Instanz-DB anzulegen.
  - Im Instanz-DB handhabt der FB 200 Parameter-, Status- und Konfigurationsdaten für die entsprechende Achse.
  - Im Beispiel ist DB 101 für Achse 1, DB 102 für Achse 2 usw.
- **FB 800 ... FB 866 - PLCopen**
  - Die PLCopen-Bausteine dienen zur Programmierung von Bewegungsabläufen und Statusabfragen.
  - Alle PLCopen-Bausteine werden innerhalb des FB 200 zyklisch aufgerufen und bei Bedarf abgearbeitet.
  - In der Applikation erfolgt der Einsatz dieser Bausteine mittels des DB 200.

- DB 200
  - Im DB 200 sind alle Parameter aller Bausteine verschaltet.
  - Durch Zugriff aus Ihrer Applikation auf den DB 200 können Sie Ihr Antriebssystem ansteuern.
- FC 100
  - Im FC 100 sind die FB200-Aufrufe für jede Achse zu programmieren.
  - Für den Einsatz von Simple Motion Control ist der FC 100 im Anwenderprogramm zyklisch aufzurufen.
  - Damit die Variablen aus dem Simple Motion Control im entsprechenden Alarm-OB zur Verfügung stehen, müssen Sie bei jedem Alarm-OB einen FC100-Aufruf programmieren.



*Bitte beachten Sie, dass Sie nach maximal 2 unterschiedlichen Bewegungs-Aufgaben einen STOP/Halt-Befehl ausführen. Ansonsten werden nachfolgende Bewegungsaufgaben ignoriert!*

### 3.3.3.2 Programmierung

#### Bibliothek einbinden

1. ➔ Gehen Sie in den Service-Bereich der entsprechenden Webseite. Über folgende Webseiten haben Sie Zugriff auf die Bibliotheken:
  - <https://www.yaskawa.eu.com/en/service/drives-motion-software-download>
  - <http://www.vipa.com/de/service-support/downloads/yaskawa-lib>
2. ➔ Laden Sie aus dem Downloadbereich die Baustein Bibliothek *Sigma-7 - YASKAWA Motion Control PROFINET*.
3. ➔ Öffnen Sie mit "Datei ➔ Dearchivieren" das Dialogfenster zur Auswahl der ZIP-Datei.
4. ➔ Wählen Sie die entsprechende ZIP-Datei an und klicken Sie auf [Öffnen].
5. ➔ Geben Sie ein Zielverzeichnis an, in dem die Bausteine abzulegen sind und starten Sie den Entpackvorgang mit [OK].

#### Bausteine in Projekt kopieren

- ➔ Öffnen Sie die Bibliothek nach dem Entpackvorgang und ziehen Sie per Drag&Drop folgende Bausteine in "Bausteine" Ihres Projekts:
  - *Sigma-7* PROFINET:
    - FB 200 - USER\_IF
    - DB 200 - DB\_USER\_IF
    - FC 100 - USER\_CallSMC
    - alle UDTs
    - FB 849 - Y\_Init
    - FB 862 - Y\_SIG7PN\_Kernel
    - FB 863 - Y\_SIG7PN\_DeviceDriver
    - FB 865 - Y\_SIG7PN\_ServoInit
    - FB 866 - Y\_SIG7PN\_ServoOrder
    - FC 1 - Y\_GCD
    - FC 260 - Y\_CheckREAL
  - Axis Control
    - FB 800 ... FB 847: Bausteine für die gewünschten Bewegungsabläufe

### Instanz-DB für Achsen anlegen

Für jede Achse ist ein Instanz-DB anzulegen.

- ➔ Klicken Sie in Ihrem Projekt auf "Bausteine" und wählen Sie "Kontextmenü ➔ Neues Objekt einfügen ➔ Datenbaustein".

Geben Sie folgende Parameter an:

- Name und Typ
  - Die DB-Nr. als "Name" können Sie frei wählen wie z.B. DB101 für Achse 1
  - Stellen Sie "Instanz-DB" von "FB 200" als "Typ" ein.
- Symbolischer Name
  - Geben Sie "FB200\_Axis01\_DB" an.

Bestätigen Sie Ihre Eingaben mit [OK].

⇒ Der Baustein wird angelegt.

### FC 100 - USER\_CallSMC

#### Simple-Motion-Aufrufstruktur

- ➔ Öffnen Sie den FC 100 und programmieren Sie für jede Achse einen FB200-Aufruf nach folgender Struktur:

```
CALL FB 200, DB n
  DiagAddress :=DW#16# [Diagnoseadresse Achse n]
  Axis_IF     := "DB_USER_IF".Axis_IF[n]
```

DiagAddress - Diagnoseadresse aus der Hardware-Konfiguration der Achse n

Axis\_IF - Achsdaten von Achse n innerhalb des DB 200 - DB\_USER\_IF

#### Beispiel-Aufruf für 3 Achsen im FC 100

```
CALL FB 200, DB 101 //Instanz-DB Achse 1
  DiagAddress :=DW#16#7DC //aus HW-Konfig Achse 1
  Axis_IF     := "DB_USER_IF".Axis_IF[1] //Daten Achse 1 DB 200

CALL FB 200, DB 102 //Instanz-DB Achse 2
  DiagAddress :=DW#16#7DD //aus HW-Konfig Achse 2
  Axis_IF     := "DB_USER_IF".Axis_IF[2] //Daten Achse 2 DB 200

CALL FB 200, DB 103 //Instanz-DB Achse 3
  DiagAddress :=DW#16#7DE //aus HW-Konfig Achse 3
  Axis_IF     := "DB_USER_IF".Axis_IF[3] //Daten Achse 3 DB 200
```



Für die Bewegungsaufgaben müssen Sie die PLCopen-Bausteine in Ihr Projekt kopieren. Innerhalb des FB 200 werden diese zyklisch aufgerufen. Mittels des DB 200 - DB\_USER\_IF können Sie Bausteine mit Parameter versorgen und deren Aufruf aktivieren. Näheres hierzu finden Sie auch im Beispielprojekt.

### OB 1

#### Anwenderprogramm

1. ➔ Öffnen Sie den OB 1 und programmieren Sie einen FC100-Aufruf:  
CALL FC 100
2. ➔ Programmieren Sie Ihre Anwenderapplikation.

#### Alarm-OBs anlegen

1. ➔ Klicken Sie in Ihrem Projekt auf "Bausteine" und wählen Sie "Kontextmenü ➔ Neues Objekt einfügen ➔ Organisationsbaustein".  
⇒ Das Dialogfenster "Eigenschaften Organisationsbaustein" öffnet sich.

2. ➤ Fügen Sie nacheinander OB 57, OB 82 und OB 86 Ihrem Projekt hinzu.



*Damit die Variablen aus dem Simple Motion Control im entsprechenden Alarm-OB zur Verfügung stehen, müssen Sie bei jedem Alarm-OB einen FC100-Aufruf programmieren:*

`CALL FC 100`

### 3.3.3.3 Zeitlicher Ablauf

#### Zeitlicher Ablauf

1. ➤ Wechseln Sie in den Siemens SIMATIC Manager und übertragen Sie Ihr Projekt in die CPU.  
⇒ Sie können jetzt Ihre Applikation in Betrieb nehmen.



#### VORSICHT!

Bitte beachten Sie immer die Sicherheitshinweise zu ihrem Antrieb, insbesondere bei der Inbetriebnahme!

2. ➤ Bevor eine Achse gesteuert werden kann, muss diese initialisiert werden. Setzen Sie hierzu im DB 200 den Wert "Axis\_IF[x].Init.Execute" auf TRUE.  
⇒
  - Der Ausgang "Axis\_IF[x].Init.Busy" meldet TRUE zurück.
  - Bei fehlerfreien Durchlauf meldet der Ausgang "Axis\_IF[x].Init.Done" TRUE zurück.
  - Bei einem Fehler meldet der Ausgang "Axis\_IF[x].Init.Error" TRUE zurück. Hierbei erhalten Sie den entsprechenden Fehlercode über den Ausgang "Axis\_IF[x].Init.ErrorID".
  - Bei Änderung von Initialisierungsparametern im DB 200 ist der Wert "Axis\_IF[x].Init.Execute" erneut auf TRUE zu setzen.



*Fahren Sie erst fort, wenn der Init-Baustein keinen Fehler meldet!*

3. ➤ Programmieren Sie Ihre Applikation mit den entsprechenden FB 200 Aufruf-Abfolgen.

## 3.4 Einsatz im Siemens TIA Portal

### 3.4.1 Voraussetzung

#### Übersicht

- Bitte verwenden Sie für die Projektierung das Siemens TIA Portal ab V15.1
- Die Bausteine können Sie bei folgenden CPUs einsetzen:
  - System SLIO CPU 017-CEFPR00 mit Erweiterung des Arbeitsspeichers auf 2MB
  - System 300S CPU 315-4PN43 mit Erweiterung des Arbeitsspeichers auf 1MB
  - System 300S CPU 315-4PN23
  - System 300S CPU 317-4PN23
- Die Projektierung der System SLIO CPU erfolgt im Siemens TIA Portal in Form des virtuellen PROFINET IO Devices "*VIPA SLIO CPU*". Das "*VIPA SLIO System*" ist mittels GSDML im Hardware-Katalog zu installieren.
- Die Projektierung der System 300S CPUs 315-4PNxx erfolgt im Siemens TIA Portal als Siemens CPU 315-2 PN/DP (6ES7 315-2EH14-0AB0 V3.2).
- Die Projektierung der System 300S CPU 317-4PN23 erfolgt im Siemens TIA Portal als Siemens CPU 317-2 PN/DP (6ES7 317-2EK14-0AB0 V3.2).
- Die Projektierung des PROFINET-Masters für die Antriebe erfolgt im Siemens TIA Portal.

### 3.4.2 Hardware-Konfiguration CPU

#### 3.4.2.1 Hardware-Konfiguration System SLIO CPU 017PN

##### IO Device "*VIPA SLIO System*" installieren

Die Projektierung der System SLIO CPU erfolgt in Form des virtuellen PROFINET IO Devices "*VIPA SLIO CPU*". Die Installation des PROFINET IO Devices "*VIPA SLIO CPU*" im Hardware-Katalog erfolgt nach folgender Vorgehensweise:

1. ➤ Gehen Sie in den Service-Bereich von [www.vipa.com](http://www.vipa.com).
2. ➤ Laden Sie aus dem Downloadbereich unter "*Config Dateien* ➔ *PROFINET*" die Konfigurationsdatei für Ihre CPU.
3. ➤ Extrahieren Sie die Datei in Ihr Arbeitsverzeichnis.
4. ➤ Starten das Siemens TIA Portal.
5. ➤ Schließen Sie alle Projekte.
6. ➤ Wechseln Sie in die *Projektansicht*.
7. ➤ Gehen Sie auf "*Extras* ➔ *Gerätebeschreibungsdatei (GSD) installieren*".
8. ➤ Navigieren Sie in Ihr Arbeitsverzeichnis und installieren Sie die entsprechende GSDML-Datei.
  - ⇒ Nach der Installation wird der Hardware-Katalog aktualisiert und das Siemens TIA Portal beendet.

Nach einem Neustart des Siemens TIA Portals finden Sie das entsprechende PROFINET-IO-Device unter *Weitere Feldgeräte* > *PROFINET* > *IO* > *VIPA GmbH* > ....



Damit die spezifischen Komponenten im Hardware-Katalog angezeigt werden können, müssen Sie im Hardware-Katalog bei "*Filter*" den Haken entfernen.

#### GSDML für YASKAWA *Sigma-7* Servo-Antrieb installieren

Im Lieferumfang des Beispielsprojekts befindet sich die GSDML-Datei für den Sigma-7 Servo-Antrieb.



Die Installation der GSDML für den *Sigma-7* Servo-Antrieb im Hardware-Katalog erfolgt nach folgender Vorgehensweise:

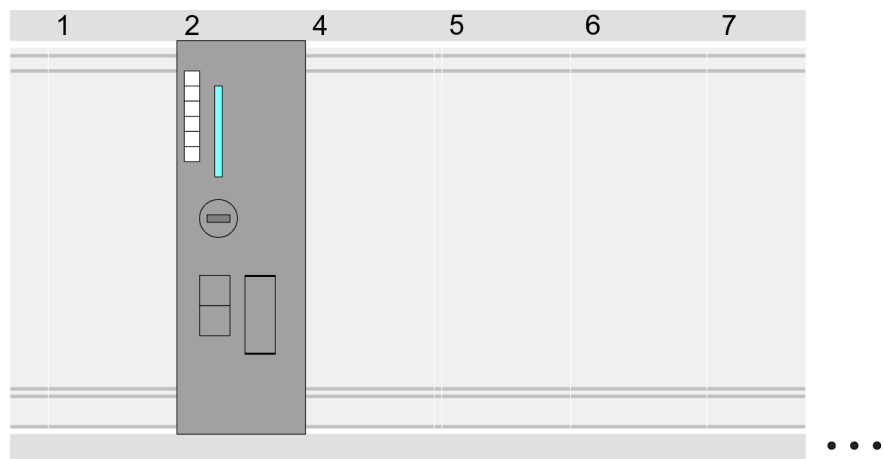
1. ➤ Extrahieren Sie aus Ihrem Beispielprojekt die GSDML-Datei in Ihr Arbeitsverzeichnis.
2. ➤ Starten das Siemens TIA Portal.
3. ➤ Schließen Sie alle Projekte.
4. ➤ Wechseln Sie in die *Projektansicht*.
5. ➤ Gehen Sie auf "*Extras* ➔ *Gerätebeschreibungsdatei (GSD) installieren*".
6. ➤ Navigieren Sie in Ihr Arbeitsverzeichnis und installieren Sie die entsprechende GSDML-Datei.
  - ⇒ Nach der Installation wird der Hardware-Katalog aktualisiert und das Siemens TIA Portal beendet.

Nach einem Neustart des Siemens TIA Portals finden Sie das PROFINET device "*SGD7...*" unter "*PROFINET IO* ➔ *Weitere Feldgeräte* ➔ *Drives* ➔ *YASKAWA Drives*".

**CPU im Projekt anlegen**

Um kompatibel mit dem Siemens TIA Portal zu sein, sind folgende Schritte durchzuführen:

1. ➤ Starten Sie das Siemens TIA Portal mit einem neuen Projekt.
2. ➤ Wechseln Sie in die *Projektansicht*.
3. ➤ Klicken Sie in der *Projektnavigation* auf "*Neues Gerät hinzufügen*".
4. ➤ Projektieren Sie für die CPU 017-CEFPR00 die Siemens CPU 317-2PN/DP (6ES7 317-2EK14-0AB0 V3.2).
  - ⇒ Die CPU wird mit einer Profilschiene eingefügt.

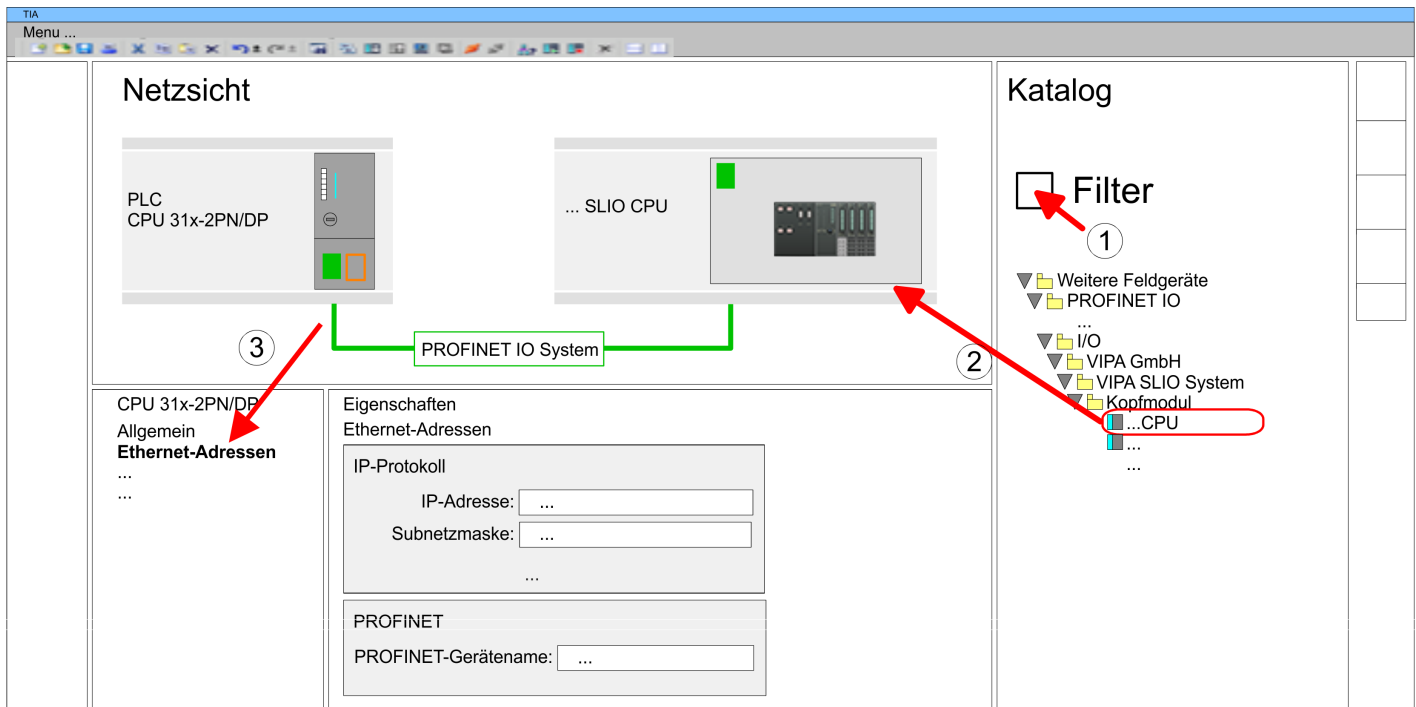


**Geräteübersicht:**

Baugruppe	...	Steckplatz	...	Typ	...
PLC...		2		CPU 317-2PN/DP	
MPI-Schnittstelle...		2 X1		MPI/DP-Schnittstelle	
PROFINET-Schnitt...		2 X2		PROFINET-Schnittstelle	
...					

### Anbindung CPU als PROFINET-IO-Device

1. ➔ Wechseln Sie im *Projektbereich* in die *"Netzschritt"*.
2. ➔ Navigieren Sie im Hardware-Katalog zu *"Weitere Feldgeräte → PROFINET IO → I/O → VIPA GmbH → VIPA SLIO System"* und binden Sie das Slave-System an die CPU an, indem Sie dies aus dem Hardware-Katalog in die *Netzschritt* ziehen und dieses über PROFINET an die CPU anbinden.
3. ➔ Klicken Sie in der *Netzschritt* auf den PROFINET-Teil der Siemens CPU und geben Sie in *"Eigenschaften"* unter *"Ethernet-Adressen"* im Bereich *"IP-Protokoll"* gültige IP-Adressdaten an.
4. ➔ Geben Sie unter *"PROFINET"* einen *"PROFINET Gerätenamen"* an. Der Geräte-name muss eindeutig am Ethernet-Subnetz sein.

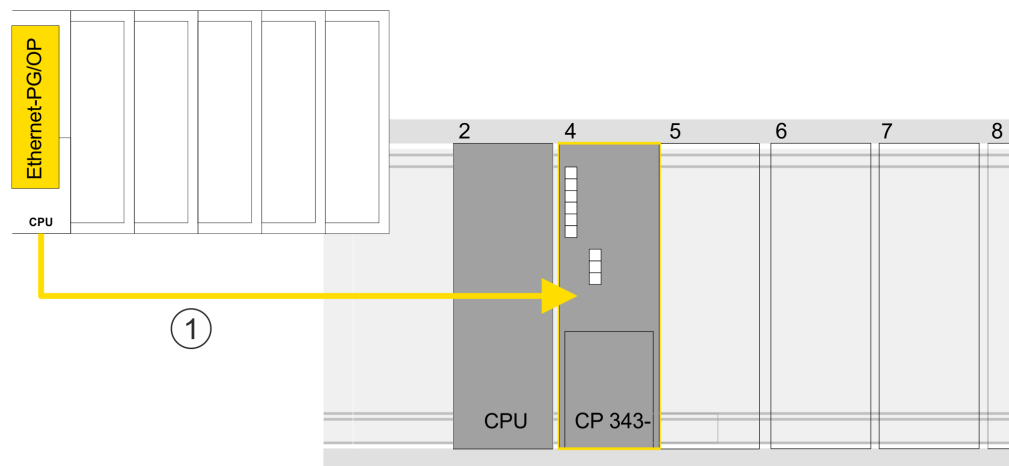


5. ➔ Wählen Sie in der *Netzschritt* das IO-Device *"VIPA SLIO CPU 017-CEFPR00"* an und wechseln Sie in die *Geräteübersicht*.
  - ⇒ In der *Geräteübersicht* des PROFINET-IO-Device *"VIPA SLIO CPU 017-CEFPR00"* ist auf Steckplatz 0 die CPU 017-CEFPR00 bereits vorplatziert. Ab Steckplatz 1 können Sie Ihre System SLIO Module platzieren.

**Ethernet-PG/OP-Kanal parametrieren**

Die CPU hat einen Ethernet-PG/OP-Kanal integriert. Über diesen Kanal können Sie Ihre CPU programmieren und fernwarten.

1. ➤ Platzieren Sie für den Ethernet-PG/OP-Kanal auf Steckplatz 4 den Siemens CP 343-1 (6GK7 343-1EX30 0XE0 V3.0).
2. ➤ Öffnen Sie durch Klick auf den CP 343-1EX30 den "Eigenschaften"-Dialog und geben Sie für den CP in den "Eigenschaften" unter "Ethernet-Adresse" die zuvor zugewiesenen IP-Adress-Daten an.
3. ➤ Übertragen Sie Ihr Projekt.

**Geräteübersicht**

Baugruppe	...	Steckplatz	...	Typ	...
PLC ...		2		CPU ...	
MPI/DP-Schnittstelle		2 X1		MPI/DP-Schnittstelle	
PROFINET-Schnittstelle		2 X2		PROFINET-Schnittstelle	
...		...		...	
CP 343-1		4		CP 343-1	
...		...		...	



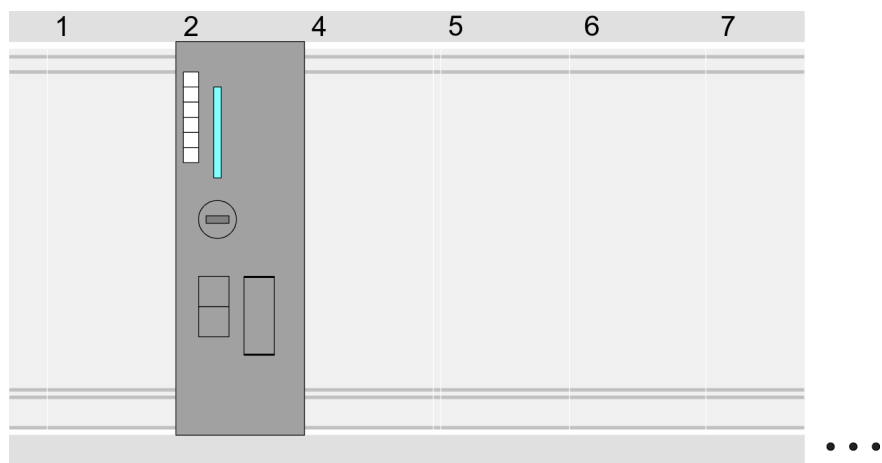
Näheres zum Einsatz des Ethernet-PG/OP-Kanals finden Sie im Handbuch zu Ihrer CPU.

## 3.4.2.2 Hardware-Konfiguration System 300S CPU 315PN ... 317PN

**Projektierung Siemens CPU**

Um kompatibel mit dem Siemens TIA Portal zu sein, sind folgende Schritte durchzuführen:

1. Starten Sie das Siemens TIA Portal.
2. Erstellen sie in der *Portalansicht* mit "*Neues Projekt erstellen*" ein neues Projekt.
3. Wechseln Sie in die *Projektansicht*.
4. Klicken Sie in der *Projektnavigation* auf "*Neues Gerät hinzufügen*".
5. Wählen Sie für die CPU 315PN die Siemens CPU 315-2 PN/DP (6ES7 315-2EH14-0AB0 V3.2) und für die CPU 317PN die Siemens CPU 317-2 PN/DP (6ES7 317-2EK14-0AB0 V3.2) aus.
  - ⇒ Die CPU wird mit einer Profilschiene eingefügt.



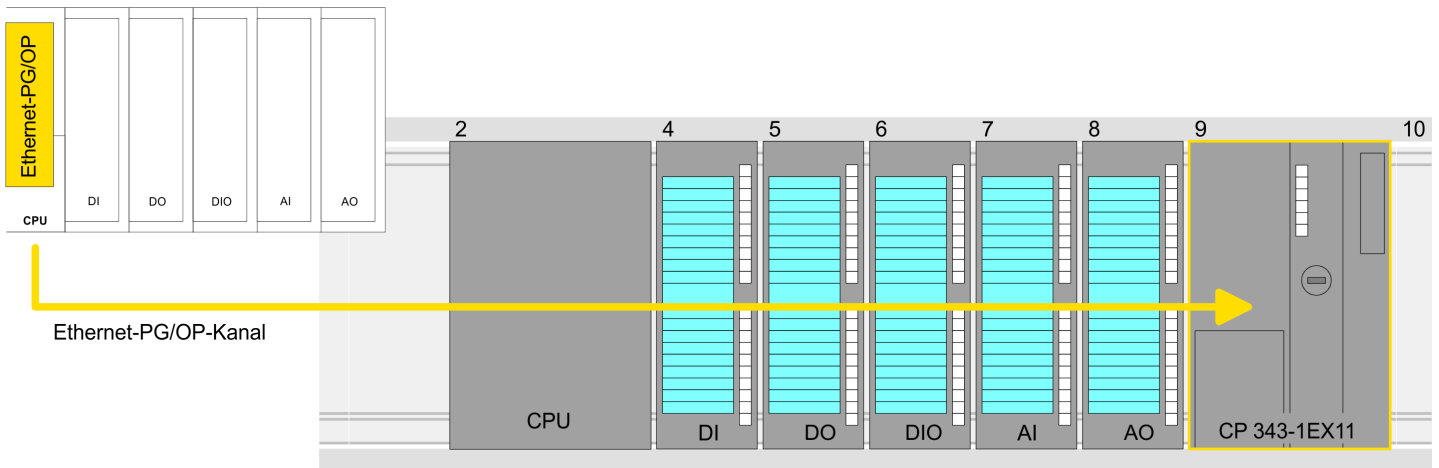
Geräteübersicht:

Baugruppe	...	Steckplatz	...	Typ	...
PLC ...		2		CPU 315-2PN/DP	
MPI/DP-Schnittstelle		2 X1		MPI/DP-Schnittstelle	
PROFINET-Schnittstelle		2 X2		PROFINET-Schnittstelle	
...		...		...	

**Ethernet-PG/OP-Kanal parametrieren**

Die CPU hat einen Ethernet-PG/OP-Kanal integriert. Über diesen Kanal können Sie Ihre CPU programmieren und fernwarten.

1. ➤ Projektieren Sie Ihre System 300 Module
2. ➤ Projektieren Sie für den Ethernet-PG/OP-Kanal immer als letztes Modul nach den reell gesteckten Modulen einen Siemens CP 343-1 (6GK7 343-1EX11 0XE0).
3. ➤ Öffnen Sie durch Doppelklick auf den CP 343-1EX11 den Eigenschaften-Dialog und geben Sie für den CP unter "Eigenschaften" die IP-Adress-Daten aus der Urtaufe an.
4. ➤ Übertragen Sie Ihr Projekt in Ihre CPU
  - ⇒ Die IP-Adress-Daten werden in Ihr Projekt übernommen.



**Geräteübersicht**

Baugruppe	...	Steckplatz	...	Typ	...
PLC...		2		CPU ...	
...		...		...	
		3			
DI...		4		DI...	
DO...		5		DO...	
DIO...		6		DIO...	
AI...		7		AI...	
AO...		8		AO...	
<span style="background-color: yellow;">■</span> CP 343-1		9		CP 343-1	



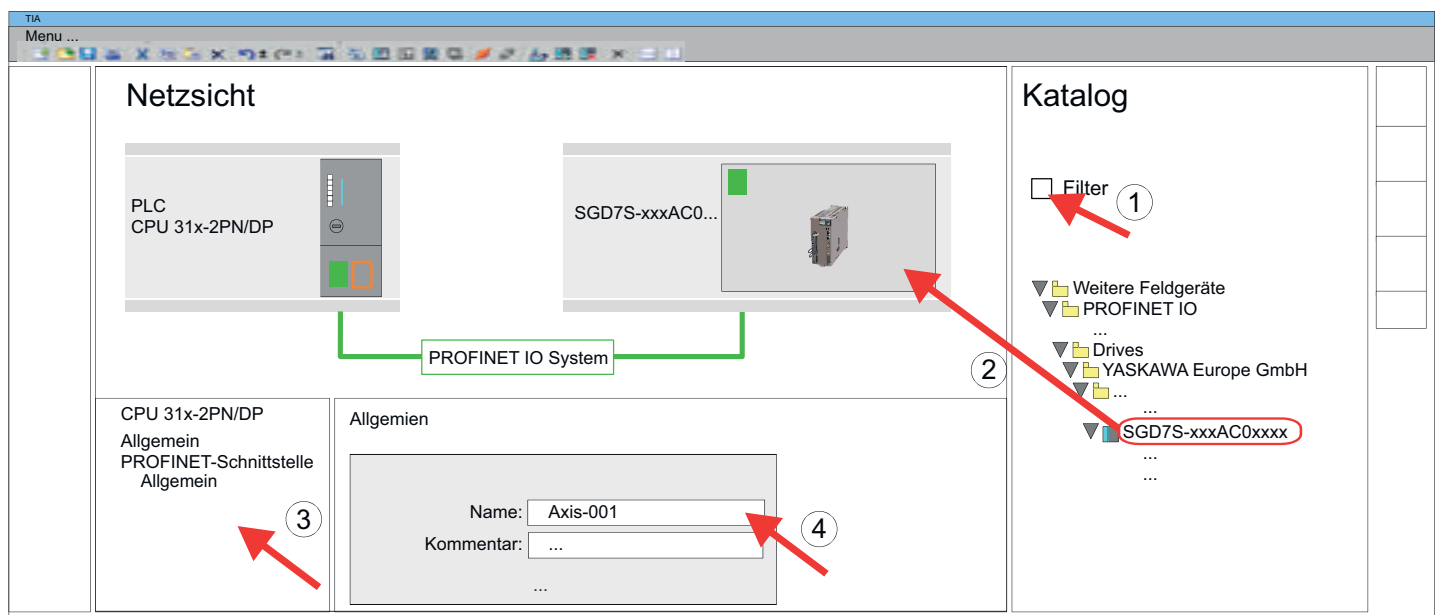
Näheres zur Urtaufe und zum Einsatz des Ethernet-PG/OP-Kanals finden Sie im Handbuch zu Ihrer CPU.

### 3.4.2.3 *Sigma-7* PROFINET Antrieb einfügen und konfigurieren

#### ***Sigma-7* PROFINET Antrieb einfügen und konfigurieren**

Die Konfiguration des Antriebs erfolgt im Siemens TIA Portal. Hierbei ist für jede Achse ein *Sigma-7* PROFINET-Device zu projektieren.

1. ➤ Wechseln Sie in die "Netzansicht".
2. ➤ Deaktivieren Sie "Filter" im Hardware-Katalog.
3. ➤ Navigieren Sie im Hardware-Katalog zu "Weitere Feldgeräte → PROFINET IO → Drives → YASKAWA Europe GmbH → YASKAWA Drives → Kopfmodul". Wählen Sie den *Sigma-7* PROFINET Antrieb "SGD7S-xxx..." an und ziehen Sie diesen auf das "PROFINET-IO-System".  
⇒ Der *Sigma-7* PROFINET Antrieb wird an den Master angebunden und kann nun konfiguriert werden.
4. ➤ Öffnen Sie die "Eigenschaften" des *Sigma-7* PROFINET Antriebs und vergeben Sie einen passenden "Gerätenamen" wie z.B. Axis-001.



5. Wechslen Sie in die "Gerätesicht" des Sigma-7 PROFINET Antriebs.

Wählen Sie aus dem Hardware-Katalog Ihren Sigma-7 PROFINET Antrieb "SGD7S-xxx..." an und ziehen Sie das Element "DO with free telegr.999..." auf den Steckplatz 1 der "Geräteübersicht" des Sigma-7 PROFINET Antriebs.

Modul	...	Steckplatz	E-Adr...	...	Typ	...
SGD7S-xxxAC0xxxx	...	0	...	...	SGD7S-xxxAC0xxxx	...
PN-IO	...	0 X1	...	...	SGD7S-xxxAC0xxxx	...
DO with free telegr.999...	...	1	...	...	DO with free telegr.999...	...
Parameter Access Point	...	1.1	2038*	...	Parameter Access Point	...
Free telegram, PZD...	...	1.2	...	...	Free telegram, PZD-16/16	...

6.



**Bitte folgendes beachten!**

- Die Verknüpfung zwischen den Achsen in der Hardware-Konfiguration und ihrem Anwenderprogramm erfolgt über die E-Adresse von "Parameter Access Point" des Submoduls "DO with free telegr.999..."
- Die Anfangsadressen von "E-Adresse" und "A-Adresse" des Submoduls "Free telegram PZD-16/16" müssen identisch sein.

7. Notieren Sie sich die E-Adresse von "Parameter Access Point" des Submoduls "DO with free telegr.999..." in der Geräteübersicht.

Modul	...	Steckplatz	E-Adr...	...	Typ	...
SGD7S-xxxAC0xxxx	...	0	...	...	SGD7S-xxxAC0xxxx	...
PN-IO	...	0 X1	...	...	SGD7S-xxxAC0xxxx	...
DO with free telegr.999...	...	1	...	...	DO with free telegr.999...	...
Parameter Access Point	...	1.1	2038*	...	Parameter Access Point	...
Free telegram, PZD...	...	1.2	...	...	Free telegram, PZD-16/16	...

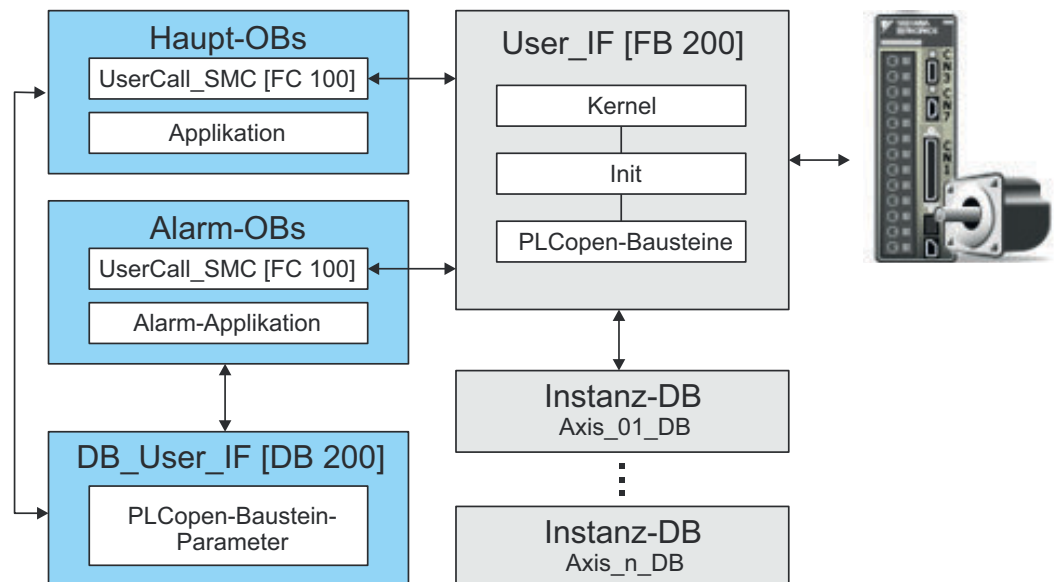
8. ➤ Speichern, übersetzen Sie Ihre Konfiguration und übertragen Sie diese in die CPU.
9. ➤ Für die Namenszuweisung wählen Sie "Online → Erreichbare Teilnehmer". Wählen Sie die PROFINET-Schnittstelle aus und klicken Sie auf "Suche starten".  
⇒ Die Ethernet-Teilnehmer werden aufgelistet.
10. ➤ Suchen Sie den Sigma7 PROFINET Antrieb "SGM7xxxACxx" und weisen Sie diesem den Namen aus der Hardware-Konfiguration wie z.B. Axis-001 zu.
11. ➤ Speichern, übersetzen Sie Ihre Konfiguration und übertragen Sie diese in die CPU.

### 3.4.3 Anwender-Programm

#### 3.4.3.1 Programmstruktur



Bitte beachten Sie, dass die Baustein-Nummern variieren können, da diese im Siemens TIA Portal automatisch generiert werden.



- **Init**
  - Die *Init*-Bausteine dienen zur Konfiguration der Achsen.
  - Die *Init*-Bausteine werden innerhalb des *User\_IF [FB 200]* zyklisch aufgerufen und bei Bedarf abgearbeitet.
  - Spezifischer Baustein für *Sigma-7* PROFINET.
  - Die Konfiguration für die Initialisierung erfolgt über den entsprechenden Instanz-DB.
- **Kernel**
  - Die *Kernel*-Bausteine werden innerhalb des *User\_IF [FB 200]* zyklisch aufgerufen.
  - Die *Kernel*-Bausteine kommunizieren mit dem Antrieb über PROFINET, verarbeiten die Benutzeraufträge und liefern Statusmeldungen zurück.
  - Der Austausch der Daten erfolgt mittels des entsprechenden Instanz-DB.



- **User\_IF [FB 200]**
  - Innerhalb des *User\_IF [FB 200]* werden alle Funktionsbausteine der Bibliothek zyklisch aufgerufen und bei Bedarf abgearbeitet.
  - Als Instanz-DB dient der beim Aufruf anzugebende DB der entsprechenden Achse.
  - Als Referenz zur Achse dient die Diagnoseadresse, welche in der Hardware-Konfiguration für die entsprechende Achse vorzugeben ist.
- **Instanz-DB - Achse-DB**
  - Für den *User\_IF [FB 200]* ist für jede Achse ein Instanz-DB anzulegen.
  - Im Instanz-DB handhabt der *User\_IF [FB 200]* Parameter-, Status- und Konfigurationsdaten für die entsprechende Achse.
  - Im Beispiel ist *Axis\_01\_DB* für Achse 1, *Axis02\_DB* für Achse 2 usw.
- **PLCopen-Bausteine**
  - Die PLCopen-Bausteine dienen zur Programmierung von Bewegungsabläufen und Statusabfragen.
  - Alle PLCopen-Bausteine werden innerhalb des *User\_IF [FB 200]* zyklisch aufgerufen und bei Bedarf abgearbeitet.
  - In der Applikation erfolgt der Einsatz dieser Bausteine mittels des *DB\_User\_IF [DB 200]*.
- **DB\_User\_IF [DB 200]**
  - Im *DB\_User\_IF [DB 200]* sind alle Parameter aller Bausteine verschaltet.
  - Durch Zugriff aus Ihrer Applikation auf den *DB\_User\_IF [DB 200]* können Sie Ihr Antriebssystem ansteuern.
- **UserCall\_SMC [FC 100]**
  - Im *UserCall\_SMC [FC 100]* sind die *User\_IF [FB 200]*-Aufrufe für jede Achse zu programmieren.
  - Für den Einsatz von Simple Motion Control ist der *UserCall\_SMC [FC 100]* im Anwenderprogramm zyklisch aufzurufen.
  - Damit die Variablen aus dem Simple Motion Control im entsprechenden Alarm-OB zur Verfügung stehen, müssen Sie bei jedem Alarm-OB einen *UserCall\_SMC [FC 100]*-Aufruf programmieren.



*Bitte beachten Sie, dass Sie nach maximal 2 unterschiedlichen Bewegungs-Aufgaben einen STOP/Halt-Befehl ausführen. Ansonsten werden nachfolgende Bewegungsaufgaben ignoriert!*

### 3.4.3.2 Programmierung

#### Bibliothek einbinden

1. ➔ Gehen Sie in den Service-Bereich der entsprechenden Webseite. Über folgende Webseiten haben Sie Zugriff auf die Bibliotheken:
  - <https://www.yaskawa.eu.com/en/service/drives-motion-software-download>
  - <http://www.vipa.com/de/service-support/downloads/yaskawa-lib>
2. ➔ Laden Sie aus dem Downloadbereich die Baustein Bibliothek *Sigma-7 - YASKAWA Motion Control PROFINET*.
3. ➔ Wechseln sie im Siemens TIA Portal in die *Projektansicht*.
4. ➔ Wählen Sie auf der rechten Seite die Task-Card "Bibliotheken".
5. ➔ Klicken Sie auf "Globale Bibliothek".
6. ➔ Klicken Sie innerhalb der "*Globalen Bibliothek*" auf die freie Fläche und wählen Sie "*Kontextmenü* ➔ *Bibliothek dearchivieren*".
7. ➔ Navigieren Sie zu ihrem Arbeitsverzeichnis und laden Sie die ZIP-Datei.
8. ➔ Wählen Sie die entsprechende ZIP-Datei an und klicken Sie auf [Öffnen].

#### Bausteine in Projekt kopieren

- ➔ Öffnen Sie die Bibliothek nach dem Entpackvorgang und ziehen Sie per Drag&Drop folgende Bausteine in "*Programmbausteine*" Ihres Projekts:
  - *Sigma-7* PROFINET:
    - User\_IF [FB 200]
    - DB\_User\_IF [DB 200]
    - User\_CallSMC [FC 100]
    - alle UDTs
    - Y\_Init [FB 849]
    - Y\_SIG7PN\_Kernel [FB 862]
    - Y\_SIG7PN\_DeviceDriver [FB 863]
    - Y\_SIG7PN\_Servolnit [FB 865]
    - Y\_SIG7PN\_ServoOrder [FB 866]
    - Y\_GCD [FC 1]
    - Y\_CheckREAL [FC 260]
  - Axis Control
    - PLCopen-Bausteine: Bausteine für die gewünschten Bewegungsabläufe

#### Instanz-DB für Achsen anlegen

- Für jede Achse ist ein Instanz-DB anzulegen.
1. ➔ Klicken Sie auf "*Projektnavigation* ➔ ...CPU... ➔ *Programmbausteine* ➔ *Neuen Baustein hinzufügen*".
    - ⇒ Das Dialogfenster "*Baustein hinzufügen*" öffnet sich.
  2. ➔ Wählen Sie den Bausteintyp "*DB Baustein*" und geben Sie folgende Parameter an:
    - Name
      - Geben Sie "Axis01\_DB" an.
    - Typ
      - Stellen Sie "*Baustein ... FB 200*" als "*Typ*" ein.
    - Nummer
      - Lassen Sie "*Automatisch*" aktiviert.
  3. ➔ Bestätigen Sie Ihre Eingaben mit [OK].
    - ⇒ Der Baustein wird angelegt.

**User\_CallSMC [FC 100]****Simple-Motion-Aufrufstruktur**

- ➔ Öffnen Sie den *User\_CallSMC [FC 100]* und programmieren Sie für jede Achse einen *User\_IF [FB 200]*-Aufruf nach folgender Struktur:

```
CALL User_IF, Axis n_DB
   DiagAddress :=DW#16# [Diagnoseadresse Achse n]
   Axis_IF     := "DB_USER_IF".Axis_IF[n]
```

**DiagAdd-ress** - Geben Sie hier die zuvor notierte E-Adresse von "Parameter Access Point" des Submoduls "DO with free telegr.999..." aus der Hardware-Konfiguration der Achse n an. ☞ Kap. 3.3.2.3 "Sigma-7 PROFINET Antrieb einfügen und konfigurieren" Seite 18

**Axis\_IF** - Achsdaten von Achse n innerhalb des *DB\_USER\_IF [DB 200]*

**Beispiel-Aufruf für 3 Achsen**

```
CALL User_IF, Axis01_DB //Instanz-DB Achse 1
   DiagAddress :=DW#16#7DC //aus HW-Konfig Achse 1
   Axis_IF     := "DB_USER_IF".Axis_IF[1] //Daten Achse 1 DB 200

CALL User_IF, Axis02_DB //Instanz-DB Achse 2
   DiagAddress :=DW#16#7DD //aus HW-Konfig Achse 2
   Axis_IF     := "DB_USER_IF".Axis_IF[2] //Daten Achse 2 DB 200

CALL User_IF, Axis03_DB //Instanz-DB Achse 3
   DiagAddress :=DW#16#7DE //aus HW-Konfig Achse 3
   Axis_IF     := "DB_USER_IF".Axis_IF[3] //Daten Achse 3 DB 200
```



Für die Bewegungsaufgaben müssen Sie die PLCopen-Bausteine in Ihr Projekt kopieren. Innerhalb des *User\_IF [FB 200]* werden diese zyklisch aufgerufen. Mittels des *DB\_USER\_IF [DB 200]* können Sie Bausteine mit Parameter versorgen und deren Aufruf aktivieren. Näheres hierzu finden Sie auch im Beispielprojekt.

**OB 1****Anwenderprogramm**

1. ➔ Öffnen Sie den OB 1 und programmieren Sie einen *User\_CallSMC [FC 100]*-Aufruf:  

```
CALL User_CallSMC
```
2. ➔ Programmieren Sie Ihre Anwenderapplikation.

**Alarm-OBs anlegen**

1. ➔ Klicken Sie auf "Projektnavigation → ...CPU... → Programmbausteine → Neuen Baustein hinzufügen".  
 ⇒ Das Dialogfenster "Neuen Baustein hinzufügen" öffnet sich.
2. ➔ Geben Sie OB 57 an und bestätigen Sie mit [OK].  
 ⇒ Der OB 57 wird angelegt.
3. ➔ Fügen Sie nacheinander OB 82 und OB 86 Ihrem Projekt hinzu.



Damit die Variablen aus dem Simple Motion Control im entsprechenden Alarm-OB zur Verfügung stehen, müssen Sie bei jedem Alarm-OB einen *User\_CallSMC [FC 100]*-Aufruf programmieren:

```
CALL User_CallSMC
```

### 3.4.3.3 Zeitlicher Ablauf

#### Zeitlicher Ablauf

1. ➤ Wählen Sie *"Bearbeiten → Übersetzen"* und übertragen Sie das Projekt in Ihre CPU. Näheres zur Übertragung Ihres Projekt finden Sie in der Onlinehilfe zum Siemens TIA Portal.  
⇒ Sie können jetzt Ihre Applikation in Betrieb nehmen.



#### VORSICHT!

Bitte beachten Sie immer die Sicherheitshinweise zu ihrem Antrieb, insbesondere bei der Inbetriebnahme!

2. ➤ Bevor eine Achse gesteuert werden kann, muss diese initialisiert werden. Setzen Sie hierzu im *DB\_User\_IF [DB 200]* den Wert *"Axis\_IF[x].Init.Execute"* auf TRUE.  
⇒
  - Der Ausgang *"Axis\_IF[x].Init.Busy"* meldet TRUE zurück.
  - Bei fehlerfreien Durchlauf meldet der Ausgang *"Axis\_IF[x].Init.Done"* TRUE zurück.
  - Bei einem Fehler meldet der Ausgang *"Axis\_IF[x].Init.Error"* TRUE zurück. Hierbei erhalten Sie den entsprechenden Fehlercode über den Ausgang *"Axis\_IF[x].Init.ErrorID"*.
  - Bei Änderung von Initialisierungsparametern im *DB\_User\_IF [DB 200]* ist der Wert *"Axis\_IF[x].Init.Execute"* erneut auf TRUE zu setzen.



*Fahren Sie erst fort, wenn der Init-Baustein keinen Fehler meldet!*

3. ➤ Programmieren Sie Ihre Applikation mit den entsprechenden *User\_IF [FB 200]* Aufruf-Abfolgen.

## 4 Bausteine zur Achskontrolle

### 4.1 Übersicht



*Bitte beachten Sie, dass die Baustein-Nummern variieren können, da diese im Siemens TIA Portal automatisch generiert werden.*

#### Antriebsspezifische Bausteine

Bausteine	Seite
UDT 862 - Y_SIG7PN_AXIS_CFG - Datenstruktur Achskonfiguration	↪ 38
UDT 850 ... UDT 859 - intern verwendete Datenstrukturen	↪ 38
FB 841 - Y_ServoFunction - Systemfunktionen	↪ 39
FB 847 - Y_ReadSafeState - Safety-Status lesen	↪ 40
FB 849 - Y_Init - Achskonfiguration	↪ 41
FB 862 - Y_SIG7PN_Kernel - Kernel	↪ 45
FB 863 - Y_SIG7PN_DeviceDriver - Diagnose intern	↪ 45
FB 865 - Y_SIG7PN_Servolnit - Initialisierung intern	↪ 45
FB 866 - Y_SIG7PN_ServoOrder - Auftragsinitialisierung intern	↪ 45

#### Komplexe Bewegungsaufgaben - PLCopen-Bausteine

Bausteine	Seite
UDT 860 - Y_SIG7PN_AXIS_REF - Datenstruktur Achsdaten	↪ 46
UDT 861 - MC_TRIGGER_REF - Datenstruktur Triggersignal	↪ 46
FB 800 - MC_Power - Achse freigeben bzw. sperren	↪ 46
FB 801 - MC_Home - Achse referenzieren	↪ 48
FB 802 - MC_Stop - Achse stoppen	↪ 50
FB 803 - MC_Halt - Achse anhalten	↪ 53
FB 804 - MC_MoveRelative - Achse relativ verfahren	↪ 55
FB 805 - MC_MoveVelocity - Achse verfahren mit konstanter Geschwindigkeit	↪ 57
FB 808 - MC_MoveAbsolute - Achse auf absolute Position verfahren	↪ 59
FB 811 - MC_Reset - Achse zurücksetzen	↪ 61
FB 812 - MC_ReadStatus - Status Achse lesen	↪ 63
FB 813 - MC_ReadAxisError - Fehler von Achse lesen	↪ 65
FB 816 - MC_ReadActualPosition - Aktuelle Position der Achse lesen	↪ 67
FB 817 - MC_ReadActualVelocity - Aktuelle Geschwindigkeit der Achse lesen	↪ 69
FB 818 - MC_ReadAxisInfo - Zusatzinformationen der Achse lesen	↪ 71
FB 819 - MC_ReadMotionState - Zustand Bewegungsauftrag lesen	↪ 73
FB 823 - MC_TouchProbe - Achsposition erfassen	↪ 75
FB 824 - MC_AbortTrigger - Achsposition erfassen abbrechen	↪ 77

Bausteine	Seite
FB 833 - Y_ReadParameter - Antriebsparameter lesen	↪ 78
FB 834 - Y_WriteParameter - Antriebsparameter schreiben	↪ 80
FB 835 - Y_HomeInit_LimitSwitch - Initialisierung Referenzfahrt auf Endschalter	↪ 82
FB 836 - Y_HomeInit_HomeSwitch - Initialisierung Referenzfahrt auf Referenzschalter	↪ 84
FB 837 - Y_HomeInit_ZeroPulse - Initialisierung Referenzfahrt auf Null Impuls	↪ 86
FB 838 - Y_HomeInit_SetPosition - Initialisierung Referenzfahrt auf aktuelle Position	↪ 88
FB 839 - MC_TorqueControl - Achse mit konstantem Drehmoment verfahren	↪ 89
FB 840 - MC_ReadActualTorque - aktuelles Drehmoment lesen	↪ 91

## 4.2 Antriebsspezifische Bausteine



Die PLCopen-Bausteine zur Achskontrolle finden Sie hier: ↪ Kap. 4 "Bausteine zur Achskontrolle" Seite 37

### 4.2.1 UDT 862 - Y\_SIG7PN\_AXIS\_CFG - Datenstruktur Achskonfiguration

Dies ist eine benutzerdefinierte Datenstruktur, die Konfigurationsdaten der Achse beinhaltet.

### 4.2.2 UDT 850 ... UDT 859 - intern verwendete Datenstrukturen

Dies sind benutzerdefinierte Datenstrukturen, welche intern in Datenstrukturen bzw. Bausteinen verwendet werden. Für den Einsatz der Bausteinbibliothek müssen Sie diese UDTs in Ihr Projekt kopieren.

### 4.2.3 FB 841 - Y\_ServoFunction - Systemfunktionen

**Beschreibung** Hier können Sie vorgeben, wie die Antriebsparameter gespeichert werden sollen.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ TRUE: Setzt alle internen achsbezogenen Fehler zurück.</li> </ul>
CmdType	INPUT	INT	<ul style="list-style-type: none"> <li>■ Temporäre Eingangsvariable zum Testen von Funktionen               <ul style="list-style-type: none"> <li>– 2: Standardparameter laden</li> <li>– 3: Speichern im EEPROM</li> <li>– 4: Software-Reset</li> <li>– 5: Reset Absolutwertgeber</li> </ul> </li> </ul>
Mode	INPUT	INT	<p>Mode</p> <ul style="list-style-type: none"> <li>■ CmdType: 2: Standardparameter laden               <ul style="list-style-type: none"> <li>– 1: Standard-SERVOPACK- und PROFINET-Parameter laden</li> <li>– 2: Standard-PROFINET-Parameter laden</li> <li>– 3: Standard-SERVOPACK-Parameter laden</li> <li>–</li> </ul> </li> <li>■ CmdType: 3: Speichern im EEPROM               <ul style="list-style-type: none"> <li>– 1: SERVOPACK-Parameter und PROFINET-Parameter im EEPROM speichern</li> <li>– 2: PROFINET-Parameter im EEPROM speichern</li> <li>– 3: SERVOPACK-Parameter im EEPROM speichern</li> </ul> </li> <li>■ CmdType: 4 und 5: 0 (fix)</li> </ul>
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag erfolgreich durchgeführt.</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Der FB ist nicht fertig und neue Ausgabewerte sind zu erwarten.</li> </ul> </li> </ul>
Active	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Baustein steuert die Achse.</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen finden Sie im Parameter <i>ErrorID</i>.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	<p>Zusätzliche Fehlerinformationen</p> <p>↳ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i></p>
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### 4.2.4 FB 847 - Y\_ReadSafeState - Safety-Status lesen

##### Beschreibung

Bei Einsatz einer Safety-Options-Karte von Yaskawa am Sigma-7-Antrieb können Sie mit diesem Baustein den Safety-Status abrufen.

##### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ TRUE: Im aktivierten Zustand wird der Parameterwert kontinuierlich geliefert.</li> </ul>
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ TRUE: Im FB sind gültige Ausgabe-Daten verfügbar.</li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ TRUE: Der FB ist nicht fertig und neue Ausgabewerte sind zu erwarten.</li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen finden Sie im Parameter <i>ErrorID</i>.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
SRI_A1_In	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ SRI-A1 Eingangssignal               <ul style="list-style-type: none"> <li>– 0: Safety Anforderung Eingangssignal ist AN.</li> </ul> </li> </ul>
SRI_A2_In	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ SRI-A2 Eingangssignal               <ul style="list-style-type: none"> <li>– 0: Safety Anforderung Eingangssignal ist AN.</li> </ul> </li> </ul>
SRI_B1_In	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ SRI-B1 Eingangssignal               <ul style="list-style-type: none"> <li>– 0: Safety Anforderung Eingangssignal ist AN.</li> </ul> </li> </ul>
SRI_B2_In	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ SRI-B2 Eingangssignal               <ul style="list-style-type: none"> <li>– 0: Safety Anforderung Eingangssignal ist AN.</li> </ul> </li> </ul>
EDM_A_Out	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ EDM-A Ausgangssignal               <ul style="list-style-type: none"> <li>– 0: Monitor für externe Geräte Ausgangssignal ist AUS.</li> </ul> </li> </ul>
EDM_B_Out	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ EDM-A Ausgangssignal               <ul style="list-style-type: none"> <li>– 0: Monitor für externe Geräte Ausgangssignal ist AUS.</li> </ul> </li> </ul>
Monitoring	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Safety-Funktion - Überwachung               <ul style="list-style-type: none"> <li>– 1 = Überwachung läuft</li> </ul> </li> </ul>
Safe	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Safety-Funktion - sicherer Zustand               <ul style="list-style-type: none"> <li>– 1 = sicherer Zustand</li> </ul> </li> </ul>
HWBB	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Safety-Funktion - HWBB               <ul style="list-style-type: none"> <li>– 1 = HWBB-Funktion läuft</li> </ul> </li> </ul>
ActiveMode	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Active Mode Status               <ul style="list-style-type: none"> <li>– 0 = Standby bzw. nicht ausgewählt</li> <li>– 1 = aktiv</li> </ul> </li> </ul>
Axis	IN_OUT	STRUCT	Referenz zur Achse



## 4.2.5 FB 849 - Y\_Init - Achskonfiguration

### Beschreibung

Dieser Baustein dient zur Konfiguration der Achse. Der Baustein ist speziell angepasst an die Verwendung eines Sigma-7-Antriebs, welcher über PROFINET angebunden ist.

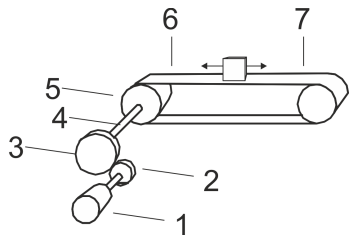
### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	Antrieb initialisieren
Resolution	INPUT	DINT	Inkrement in [Anwendereinheiten]
GearRatioNum	INPUT	DINT	Übersetzungsverhältnis <i>Gear Ratio</i> Zähler
GearRatioDenom	INPUT	DINT	Übersetzungsverhältnis <i>Gear Ratio</i> Nenner
FeedConstantNum	INPUT	DINT	Vorschub pro Motorumdrehung <i>Feed Constant</i> Zähler
FeedConstantDenom	INPUT	DINT	Vorschub pro Lastumdrehung <i>Feed Constant</i> Nenner
AxisType	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Konfiguration des Achstyps               <ul style="list-style-type: none"> <li>– FALSE: begrenzt (z.B. Förderband mit Begrenzung)</li> <li>– TRUE: endlos (z.B. Rotationstisch ohne Anschlag)</li> </ul> </li> </ul>
MinPosition	INPUT	REAL	SW-Schalter für min. Verfahrbereich <ul style="list-style-type: none"> <li>■ <i>Achstyp</i>: begrenzt               <ul style="list-style-type: none"> <li>– Minimale Position, welche angefahren werden kann.</li> </ul> </li> <li>■ <i>Achstyp</i>: endlos               <ul style="list-style-type: none"> <li>– Position, ab der auf den maximalen Positionswert gesprungen wird.</li> </ul> </li> </ul>
MaxPosition	INPUT	REAL	SW-Schalter für max. Verfahrbereich <ul style="list-style-type: none"> <li>■ <i>Achstyp</i>: begrenzt               <ul style="list-style-type: none"> <li>– Maximale Position, welche angefahren werden kann.</li> </ul> </li> <li>■ <i>Achstyp</i>: endlos               <ul style="list-style-type: none"> <li>– Position, ab der auf den minimalen Positionswert gesprungen wird.</li> </ul> </li> </ul>
MinUserPos	OUTPUT	REAL	Minimale Benutzerposition basierend auf den konfigurierten Anwendereinheiten in [Anwendereinheiten].
MaxUserPos	OUTPUT	REAL	Maximale Benutzerposition basierend auf den konfigurierten Anwendereinheiten in [Anwendereinheiten].
MaxUserVelocity	OUTPUT	REAL	Maximale Geschwindigkeit basierend auf den konfigurierten Anwendereinheiten [Anwendereinheiten/s].
MaxUserAcceleration	OUTPUT	REAL	Maximale Beschleunigung basierend auf den konfigurierten Anwendereinheiten [Anwendereinheiten/s <sup>2</sup> ].
MaxUserTorque	OUTPUT	REAL	Maximales Moment/Kraft [1,0% Rated Motor Torque] auf Motor-Seite, unabhängig von der konfigurierten Mechanik.
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Antrieb ist initialisiert</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Der FB ist nicht fertig und neue Ausgabewerte sind zu erwarten</li> </ul> </li> </ul>
Active	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Baustein steuert die Achse</li> </ul> </li> </ul>

Parameter	Deklaration	Datentyp	Beschreibung
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status <ul style="list-style-type: none"> <li>- TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen finden Sie im Parameter <i>ErrorID</i>.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen <a href="#">↗ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</a>
Axis	IN_OUT	STRUCT	Referenz zur Achse

### Bausteinbeschaltung - Förderband

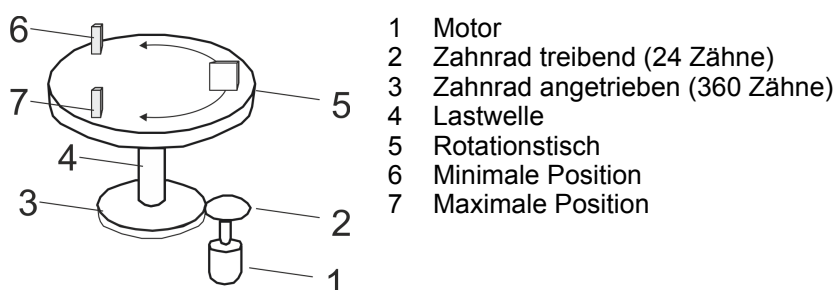
Parameter	Deklaration	Datentyp	Beispielwert	Pos.
Resolution	INPUT	DINT	10 [Inkrement pro Anwendereinheit]	-
GearRatioNum	INPUT	DINT	30 Umdrehungen (Motor)	1
GearRatioDenom	INPUT	DINT	12 Umdrehungen (Last)	1
FeedConstantNum	INPUT	DINT	314 [Anwendereinheit]	2
FeedConstantDenom	INPUT	DINT	10 Umdrehung (Last)	3
AxisType	INPUT	BOOL	FALSE (Strecke ist begrenzt)	-
MinPosition	INPUT	REAL	0,0 [Anwendereinheit]	6
MaxPosition	INPUT	REAL	1080,0 [Anwendereinheit]	7



- 1 Motor
- 2 Zahnrad treibend (12 Zähne)
- 3 Zahnrad angetrieben (30 Zähne)
- 4 Lastwelle
- 5 Förderband (Umfang 31,4 mm)
- 6 Minimale Position
- 7 Maximale Position

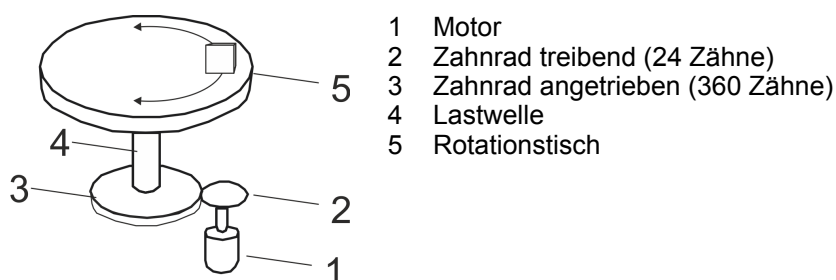
### Bausteinbeschaltung - Rotationstisch mit Anschlag

Parameter	Deklaration	Datentyp	Beispielwert	Pos.
Resolution	INPUT	DINT	10 [Inkmente pro Anwendereinheit]	-
GearRatioNum	INPUT	DINT	360 Umdrehungen (Motor)	1
GearRatioDenom	INPUT	DINT	24 Umdrehung (Last)	1
FeedConstantNum	INPUT	DINT	360 [Anwendereinheit]	2
FeedConstantDenom	INPUT	DINT	1 Umdrehung (Last)	3
AxisType	INPUT	BOOL	FALSE (Strecke ist begrenzt)	-
MinPosition	INPUT	REAL	0,0 [Anwendereinheit]	6
MaxPosition	INPUT	REAL	270,0 [Anwendereinheit]	7



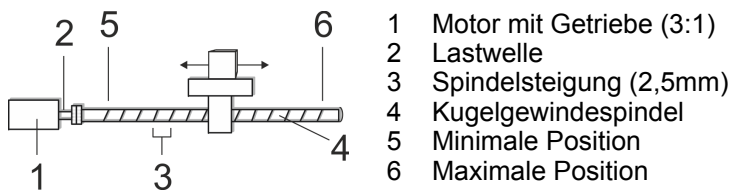
### Bausteinbeschaltung - Rotationstisch ohne Anschlag

Parameter	Deklaration	Datentyp	Beispielwert	Pos.
Resolution	INPUT	DINT	10 [Inkmente pro Anwendereinheit]	-
GearRatioNum	INPUT	DINT	360 Umdrehungen (Motor)	1
GearRatioDenom	INPUT	DINT	24 Umdrehung (Last)	1
FeedConstantNum	INPUT	DINT	360 [Anwendereinheit]	2
FeedConstantDenom	INPUT	DINT	1 Umdrehung (Last)	3
AxisType	INPUT	BOOL	TRUE (Strecke ist endlos)	-
MinPosition	INPUT	REAL	0,0 [Anwendereinheit]	-
MaxPosition	INPUT	REAL	270,0 [Anwendereinheit]	-



**Bausteinbeschaltung -  
Kugelgewindespindel**

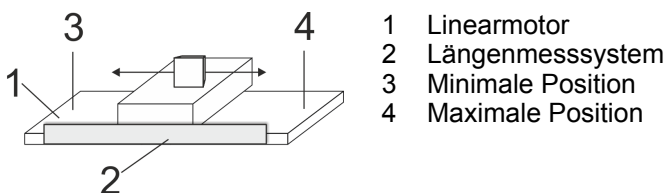
Parameter	Deklaration	Datentyp	Beispielwert	Pos.
Resolution	INPUT	DINT	100 [mm pro Anwendereinheit]	-
GearRatioNum	INPUT	DINT	3 Umdrehungen (Motor)	1
GearRatioDenom	INPUT	DINT	1 Umdrehung (Last)	1
FeedConstantNum	INPUT	DINT	25 [Anwendereinheit]	3
FeedConstantDenom	INPUT	DINT	10 Umdrehung (Last)	3
AxisType	INPUT	BOOL	FALSE (Strecke ist begrenzt)	-
MinPosition	INPUT	REAL	0 [Anwendereinheit]	5
MaxPosition	INPUT	REAL	1000 [Anwendereinheit]	6

**Bausteinbeschaltung -  
Linearachse**

Parameter	Deklaration	Datentyp	Beispielwert	Pos.
Resolution	INPUT	DINT	10 [Inkrement pro Anwendereinheit]	-
GearRatioNum	INPUT	DINT	1 Umdrehung (Motor)	siehe
GearRatioDenom	INPUT	DINT	1 Umdrehung (Last)	Hinweis
FeedConstantNum	INPUT	DINT	20 [Anwendereinheit]	siehe
FeedConstantDenom	INPUT	DINT	1000 Umdrehungen (Last)	Hinweis
AxisType	INPUT	BOOL	FALSE (Strecke ist begrenzt)	-
MinPosition	INPUT	REAL	0.0 [Anwendereinheit]	3
MaxPosition	INPUT	REAL	500.0 [Anwendereinheit]	4



- Beim Linearmotor ist eine Getriebeverhältnis von 1:1 erforderlich.
- In dieser Beispielapplikation müssen Sie mittels SigmaWin+ bzw. FB 834 - Y\_WriteParameter ↪ 80 den Parameter Pn20A "Number of External Encoder Pitches" auf 20 [Scale Pitch pro Umdrehung] einstellen.



#### 4.2.6 FB 862 - Y\_SIG7PN\_Kernel - Kernel

**Beschreibung** Dieser Baustein setzt die Antriebskommandos für eine Sigma-7 Achse über PROFINET um. Der Baustein wird zyklisch innerhalb des FB 200 aufgerufen.

##### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
DPV1_ID	INPUT	DWORD	DPV1 DIAG
DIAG_BUF	INPUT	BOOL	Diagnosepuffer <ul style="list-style-type: none"> <li>■ TRUE = Diagnosepuffer der CPU ist aktiv</li> <li>– relevant für den Service</li> </ul>
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### 4.2.7 FB 863 - Y\_SIG7PN\_DeviceDriver - Diagnose intern

**Beschreibung** Dieser Baustein wird intern verwendet.

#### 4.2.8 FB 865 - Y\_SIG7PN\_Servolnit - Initialisierung intern

**Beschreibung** Dieser Baustein wird intern verwendet.

#### 4.2.9 FB 866 - Y\_SIG7PN\_ServoOrder - Auftragsinitialisierung intern

**Beschreibung** Dieser Baustein wird intern verwendet.

## 4.3 Komplexe Bewegungsaufgaben - PLCopen-Bausteine

### 4.3.1 UDT 860 - Y\_SIG7PN\_AXIS\_REF - Datenstruktur Achsdaten

Dies ist eine benutzerdefinierte Datenstruktur, die Statusinformationen der Achse beinhaltet.

### 4.3.2 UDT 861 - MC\_TRIGGER\_REF - Datenstruktur Triggersignal

Diese ist eine benutzerdefinierte Datenstruktur, die Informationen zum Triggersignal beinhaltet.

### 4.3.3 FB 800 - MC\_Power - Achsenfreigabe

#### Beschreibung

Mit MC\_Power kann eine Achse freigegeben bzw. gesperrt werden.



*Bitte beachten Sie, dass solange der Antrieb noch nicht initialisiert ist, insbesondere bei der Erstinbetriebnahme, der FB 800 - MC\_Power die Fehlermeldung 0x8103 auslösen kann. Mit dem Aufruf des FB 849 - Y\_Init wird dieser Fehler automatisch zurückgesetzt. Das Zurücksetzen des Fehlers mit dem FB 811 - MC\_Reset ist nicht möglich.*

#### Parameter

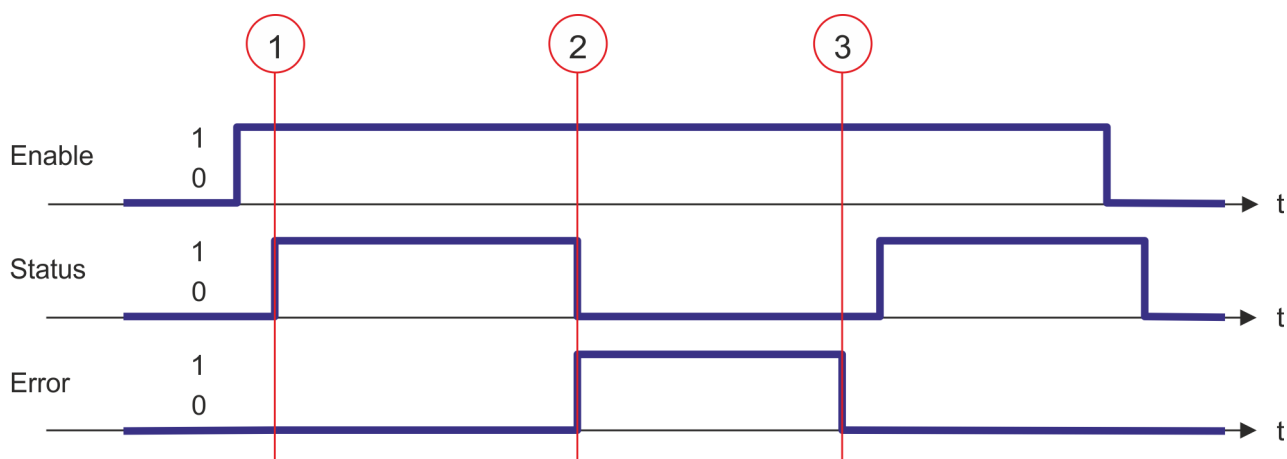
Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Achsenfreigabe               <ul style="list-style-type: none"> <li>– TRUE: Die Achse wird freigegeben</li> <li>– FALSE: Die Achse wird gesperrt</li> </ul> </li> </ul>
Status	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status Achse               <ul style="list-style-type: none"> <li>– TRUE: Achse nimmt Bewegungsaufträge an</li> <li>– FALSE: Achse nimmt keine Bewegungsaufträge an</li> </ul> </li> </ul>
Valid	OUTPUT	BOOL	Immer FALSE
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Fehler               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden. Die Achse wird gesperrt.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen <a href="#">↪ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</a>
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### Achse freigeben

Aufruf von MC\_Power mit *Enable* = TRUE. Sobald *Status* den Wert TRUE zeigt, ist die Achse freigegeben. In diesem Zustand können Bewegungsaufträge aktiviert werden.

#### Achse sperren

Aufruf von MC\_Power mit *Enable* = FALSE. Sobald *Status* den Wert FALSE zeigt, ist die Achse gesperrt. Bei Sperren der Achse wird ein ggf. aktiver Bewegungsauftrag abgebrochen und die Achse gestoppt.

**Zustandsdiagramm der Bausteinparameter**

- (1) Die Achse wird mit *Enable* = TRUE freigegeben. Zum Zeitpunkt (1) ist die Freigabe erfolgt. Anschließend können Bewegungsaufträge aktiviert werden.
- (2) Zum Zeitpunkt (2) tritt ein Fehler auf, der das Sperren der Achse zur Folge hat. Ein ggf. aktiver Bewegungsauftrag wird abgebrochen und die Achse gestoppt.
- (3) Der Fehler wird beseitigt und zum Zeitpunkt (3) quittiert. Da *Enable* weiterhin gesetzt ist, wird die Achse wieder freigegeben. Zuletzt wird die Achse mit *Enable* = FALSE gesperrt.

### 4.3.4 FB 801 - MC\_Home - Achse referenzieren

#### Beschreibung

Mit MC\_Home kann eine Achse referenziert werden. Dadurch kann ein Bezug zwischen der Position der Achse und der mechanischen Stellung hergestellt werden. Die Referenzfahrt-Methode und die zugehörigen Parameter müssen Sie direkt am Antrieb konfigurieren. Verwenden Sie hierzu einen der Y\_HomeInit\_... Bausteine. Sobald *Done* des Y\_HomeInit\_... Bausteins TRUE meldet, ist der FB 801 - MC\_Home aufzurufen.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Referenzfahrt</li> <li>– Flanke 0-1: Referenzfahrt wird gestartet</li> </ul>
Position	INPUT	REAL	<p>Bei erfolgreicher Referenzierung wird die Istposition der Achse einmalig gleich <i>Position</i> gesetzt.</p> <p><i>Position</i> ist in der verwendeten Anwendereinheit anzugeben.</p>
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag erfolgreich durchgeführt</li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Der Auftrag wurde während der Bearbeitung von einem anderen Auftrag abgebrochen.</li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul>
ErrorID	OUTPUT	WORD	<p>Zusätzliche Fehlerinformationen</p> <p>↳ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</p>
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

Start des Auftrags nur im PLCopen-State *Standstill* möglich. ↳ Kap. 5.1 "PLCopen-States" Seite 92

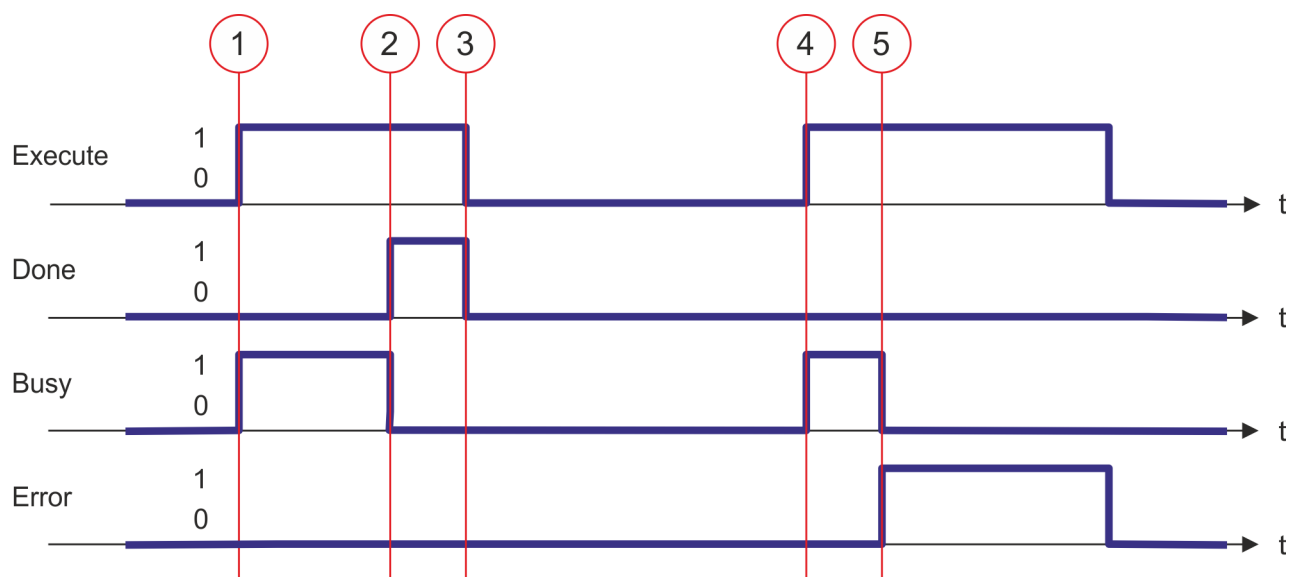
#### Achse referenzieren

Mit einer Flanke 0-1 an *Execute* wird die Referenzierung gestartet. Solange die Referenzierung läuft zeigt *Busy* den Wert TRUE. Sobald *Done* den Wert TRUE hat, ist die Referenzierung erfolgreich abgeschlossen. Die Istposition der Achse wurde auf den Wert von *Position* gesetzt.



- Ein laufender Auftrag wird auch beim Setzen von *Execute* gleich FALSE weiterhin ausgeführt.
- Ein laufender Auftrag kann durch einen Bewegungsauftrag (z.B. MC\_MoveRelative) nicht abgebrochen werden.



**Zustandsdiagramm der Bausteinparameter**

- (1) Mit Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird die Referenzierung gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) ist die Referenzierung abgeschlossen. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.
- (4) Zum Zeitpunkt (4) wird erneut die Referenzierung mit einer Flanke 0-1 an *Execute* gestartet und *Busy* liefert den Wert TRUE.
- (5) Zum Zeitpunkt (5) tritt ein Fehler bei der Referenzierung auf. *Busy* liefert den Wert FALSE und *Error* den Wert TRUE.

### 4.3.5 FB 802 - MC\_Stop - Achse stoppen

#### Beschreibung

Mit MC\_Stop wird die Achse gestoppt. Mit dem Parameter *Deceleration* kann das dynamische Verhalten beim Stoppvorgang bestimmt werden.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Achse stoppen</li> <li>– Flanke 0-1: Stoppen der Achse wird gestartet</li> </ul>
Deceleration	INPUT	REAL	Verzögerung beim Stoppen in [Anwendereinheiten/s <sup>2</sup> ]
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag erfolgreich durchgeführt</li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Der Auftrag wurde während der Bearbeitung von einem anderen Auftrag abgebrochen</li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

- Start des Auftrags in den PLCopen-States *Standstill*, *Homing*, *Discrete Motion* und *Continuous Motion* möglich.
- MC\_Stop führt die Achse in den PLCopen-State *Stopping* über. In *Stopping* können keine Bewegungsaufträge gestartet werden. Solange *Execute* gleich TRUE ist, bleibt die Achse im PLCopen-State *Stopping*. Wird *Execute* gleich FALSE gesetzt, geht die Achse in den PLCopen-State *Standstill* über. In *Standstill* können Bewegungsaufträge gestartet werden.
- ↪ *Kap. 5.1 "PLCopen-States" Seite 92*

#### Achse stoppen

Mit einer Flanke 0-1 an *Execute* wird das Stoppen der Achse gestartet. Solange das Stoppen der Achse läuft, zeigt *Busy* den Wert TRUE. Nachdem die Achse gestoppt wurde und somit die Geschwindigkeit 0 erreicht hat, wird *Busy* gleich FALSE und *Done* gleich TRUE geliefert.

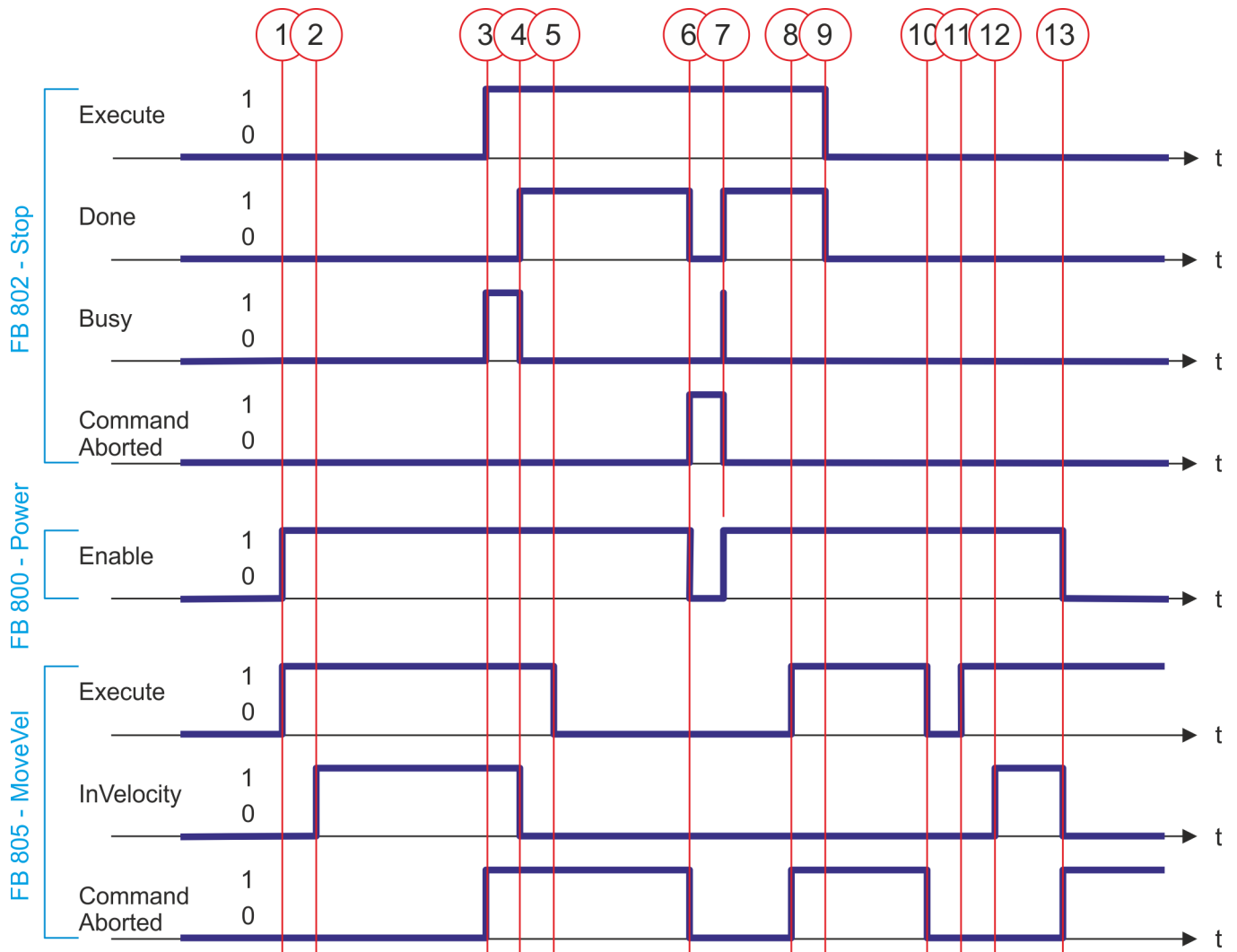


- Ein laufender Auftrag wird auch beim Setzen von *Execute* gleich FALSE bis zum Stopp der Achse ausgeführt.
- Ein laufender Auftrag kann durch einen Bewegungsauftrag (z.B. *MC\_MoveRelative*) nicht abgebrochen werden.

**Zustandsdiagramm der Bausteinparameter**

- (1) Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird das Stoppen der Achse gestartet und *Busy* liefert den Wert TRUE. Die Geschwindigkeit der Achse wird unter Berücksichtigung des Parameters *Deceleration* bis auf null verringert.
- (2) Zum Zeitpunkt (2) ist das Stoppen der Achse abgeschlossen, die Achse ist gestoppt. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.

### Verhalten bei Wiederkehr der Achsfreigabe



- (1) Mit Flanke 0-1 an *Enable* von FB 800 wird die Achse freigegeben. Mit Flanke 0-1 an *Execute* von FB 805 wird das Verfahren der Achse gestartet.
- (2) Die Achse erreicht die Geschwindigkeitsvorgabe und *InVelocity* liefert TRUE.
- (3) Mit Flanke 0-1 an *Execute* von FB 802 wird das Stoppen der Achse gestartet und *Busy* liefert TRUE. Die Achse wird bis zum Stillstand abgebremst. *CommandAborted* von FB 805 liefert TRUE.
- (4) Das Stoppen der Achse ist abgeschlossen, die Achse ist gestoppt. *Busy* liefert FALSE und *Done* TRUE. Da die Achse steht, liefert *InVelocity* des FB 805 FALSE.
- (5) Der FB 805 wird beendet.
- (6) *Enable* von FB 800 wird auf FALSE gesetzt und Achse gesperrt. Dies setzt in allen Bausteinen *CommandAborted* auf TRUE und *Done* von FB 802 auf FALSE.
- (7) Mit Flanke 0-1 an *Enable* von FB 800 wird die Achse wieder freigegeben. Hierdurch liefert *CommandAborted* von FB 802 FALSE und *Busy* kurzzeitig TRUE.
- (8) Mit Flanke 0-1 an *Execute* von FB 805 sollte das Verfahren der Achse gestartet werden. Der aktive FB 802 verhindert dies und *CommandAborted* liefert TRUE.
- (9) Der FB 802 wird beendet. Das Verfahren der Achse kann nur durch eine Flanke 0-1 von *Execute* des FB 805 gestartet werden.
- (10) Der FB 805 wird beendet und *CommandAborted* liefert FALSE.
- (11) Mit Flanke 0-1 an *Execute* von FB 805 wird das Verfahren der Achse gestartet.
- (12) Die Achse erreicht die Geschwindigkeitsvorgabe und *InVelocity* liefert TRUE.
- (13) *Enable* von FB 800 wird auf FALSE gesetzt und somit die Achse gesperrt. Dies setzt in allen Bausteinen *CommandAborted* auf TRUE.

### 4.3.6 FB 803 - MC\_Halt - Achse anhalten

**Beschreibung** Mit MC\_Halt wird die Achse bis zum Stillstand abgebremst. Mit dem Parameter *Deceleration* kann das dynamische Verhalten beim Bremsvorgang bestimmt werden.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Achse anhalten</li> <li>– Flanke 0-1: Anhalten der Achse wird gestartet</li> </ul>
Deceleration	INPUT	REAL	Verzögerung beim Bremsen in [Anwendereinheiten/s <sup>2</sup> ]
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag erfolgreich durchgeführt</li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul>
Active	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Baustein hat die Kontrolle über die Achse</li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Der Auftrag wurde während der Bearbeitung von einem anderen Auftrag abgebrochen</li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen <ul style="list-style-type: none"> <li>↳ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</li> </ul>
Axis	IN_OUT	STRUCT	Referenz zur Achse

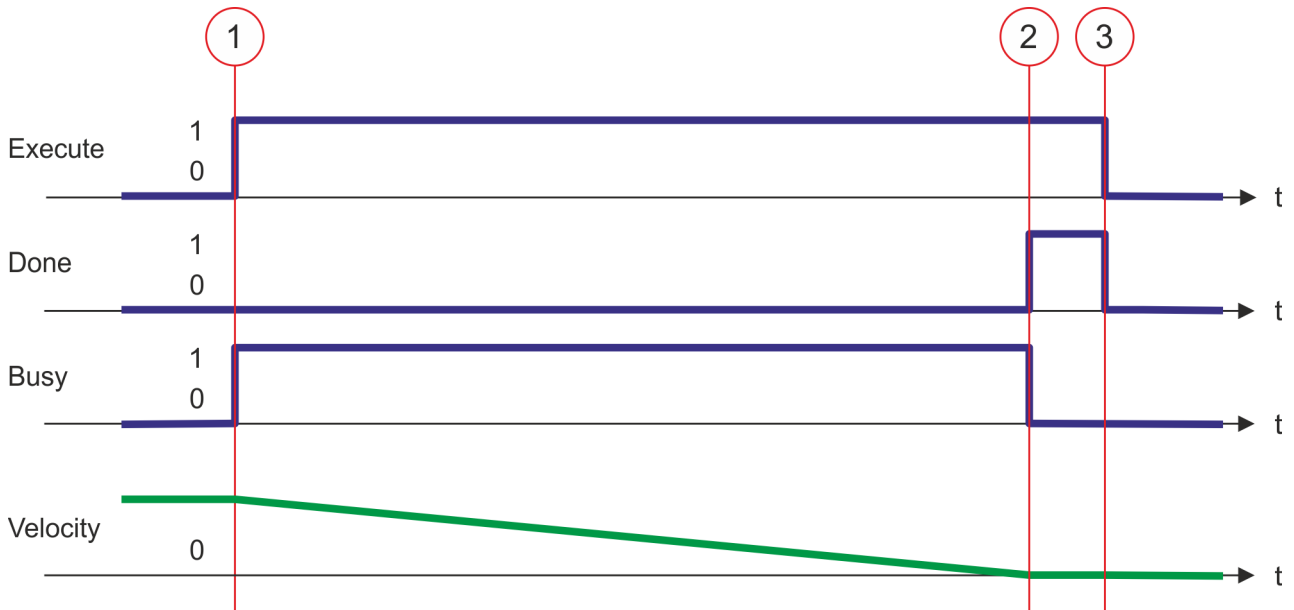
- PLCopen-State**
- Start des Auftrags in den PLCopen-States *Discrete Motion* und *Continuous Motion* möglich.
  - MC\_Halt führt die Achse in den PLCopen-State *Discrete Motion* über.
  - ↳ Kap. 5.1 "PLCopen-States" Seite 92

**Achse anhalten** Mit einer Flanke 0-1 an *Execute* wird das Anhalten der Achse gestartet. Solange das Anhalten der Achse läuft, zeigt *Busy* den Wert TRUE. Nachdem die Achse angehalten wurde und somit die Geschwindigkeit 0 erreicht hat, wird *Busy* gleich FALSE und *Done* gleich TRUE geliefert.



- Ein laufender Halt-Auftrag wird auch beim Setzen von *Execute* gleich FALSE bis zum Anhalten der Achse ausgeführt.
- Ein laufender Halt-Auftrag kann durch einen anderen Bewegungsauftrag (z.B. *MC\_MoveRelative*) abgebrochen werden.

### Zustandsdiagramm der Bausteinparameter



- (1) Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird das Anhalten der Achse gestartet und *Busy* liefert den Wert TRUE. Die Geschwindigkeit der Achse wird unter Berücksichtigung des Parameters *Deceleration* bis auf 0 verringert.
- (2) Zum Zeitpunkt (2) ist das Anhalten der Achse abgeschlossen, die Achse steht. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.

### 4.3.7 FB 804 - MC\_MoveRelative - Achse relativ verfahren

#### Beschreibung

Mit MC\_MoveRelative wird die Achse relativ zu der Position bei Auftragsstart um eine spezifizierte Distanz verfahren. Mit den Parametern *Velocity*, *Acceleration* und *Deceleration* wird das dynamische Verhalten beim Bewegungsvorgang bestimmt.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Achse relativ verfahren               <ul style="list-style-type: none"> <li>– Flanke 0-1: Das relative Verfahren der Achse wird gestartet</li> </ul> </li> </ul>
Distance	INPUT	REAL	Relative Wegstrecke in [Anwendereinheiten]
Velocity	INPUT	REAL	Max. Geschwindigkeit (muss nicht zwingend erreicht werden) in [Anwendereinheiten/s]
Acceleration	INPUT	REAL	Beschleunigung in [Anwendereinheiten/s <sup>2</sup> ]
Deceleration	INPUT	REAL	Verzögerung in [Anwendereinheiten/s <sup>2</sup> ]
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag erfolgreich durchgeführt; Zielposition erreicht</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul> </li> </ul>
Active	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Baustein hat die Kontrolle über die Achse</li> </ul> </li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Der Auftrag wurde während der Bearbeitung von einem anderen Auftrag abgebrochen</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

- Start des Auftrags in den PLCopen-States *Standstill*, *Discrete Motion* und *Continuous Motion* möglich.
- MC\_MoveRelative führt die Achse in den PLCopen-State *Discrete Motion* über.
- ↪ Kap. 5.1 "PLCopen-States" Seite 92

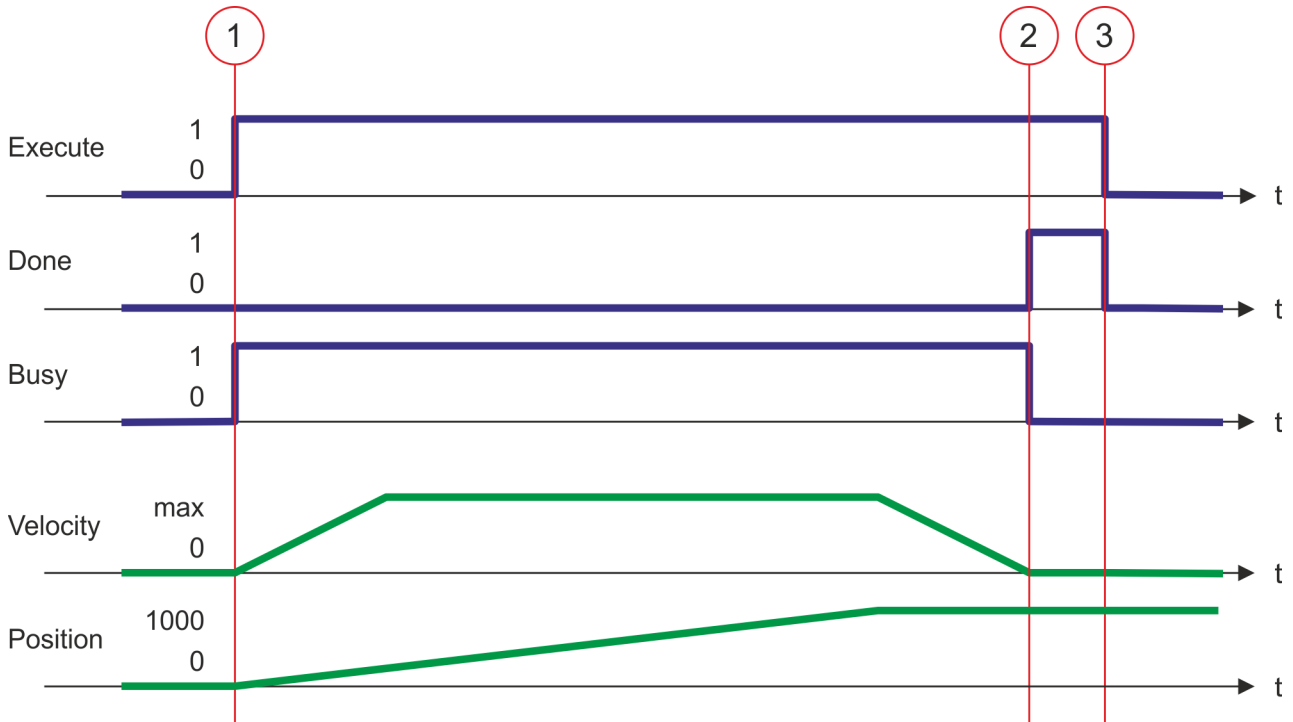
#### Achse relativ verfahren

Mit einer Flanke 0-1 an *Execute* wird das Verfahren der Achse gestartet. Solange das Verfahren der Achse läuft, zeigt *Busy* den Wert TRUE. Nachdem die Zielposition erreicht wurde, wird *Busy* gleich FALSE und *Done* gleich TRUE geliefert. Die Geschwindigkeit der Achse ist dann gleich 0.



- Ein laufender Auftrag wird auch beim Setzen von Execute gleich FALSE bis die Achse die Zielposition erreicht hat, ausgeführt.
- Ein laufender Auftrag kann durch einen anderen Bewegungsauftrag (z.B. MC\_MoveAbsolute) abgebrochen werden.

### Zustandsdiagramm der Bausteinparameter



- (1) Die Achse wird mit MC\_MoveRelative um eine  $Distance = 1000.0$  verfahren (Startposition bei Auftragsstart gleich  $0.0$ ). Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird das Verfahren der Achse gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) wurde die Achse um die  $Distance = 1000.0$  verfahren, d.h. die Zielposition wurde erreicht. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.



### 4.3.8 FB 805 - MC\_MoveVelocity - Achse verfahren mit konstanter Geschwindigkeit

#### Beschreibung

Mit MC\_MoveVelocity wird die Achse mit einer konstanten Geschwindigkeit verfahren. Mit den Parametern *Velocity*, *Acceleration* und *Deceleration* wird das dynamische Verhalten beim Bewegungsvorgang bestimmt.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Achse mit konstanter Geschwindigkeit verfahren               <ul style="list-style-type: none"> <li>– Flanke 0-1: Das Verfahren der Achse mit konstanter Geschwindigkeit wird gestartet</li> </ul> </li> </ul>
Velocity	INPUT	REAL	Geschwindigkeitsvorgabe in [Anwendereinheiten/s]
Acceleration	INPUT	REAL	Beschleunigung in [Anwendereinheiten/s <sup>2</sup> ]
Deceleration	INPUT	REAL	Verzögerung in [Anwendereinheiten/s <sup>2</sup> ]
InVelocity	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Geschwindigkeitsvorgabe               <ul style="list-style-type: none"> <li>– TRUE: Geschwindigkeitsvorgabe erreicht</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul> </li> </ul>
Active	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Baustein hat die Kontrolle über die Achse</li> </ul> </li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Der Auftrag wurde während der Bearbeitung von einem anderen Auftrag abgebrochen</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

- Start des Auftrags in den PLCopen-States *Standstill*, *Discrete Motion* und *Continuous Motion* möglich.
- MC\_MoveVelocity führt die Achse in den PLCopen-State *Continuous Motion* über.
- ↪ Kap. 5.1 "PLCopen-States" Seite 92

Komplexe Bewegungsaufgaben - PLCopen-Bausteine > FB 805 - MC\_MoveVelocity - Achse verfahren mit konstanter Geschwindigkeit

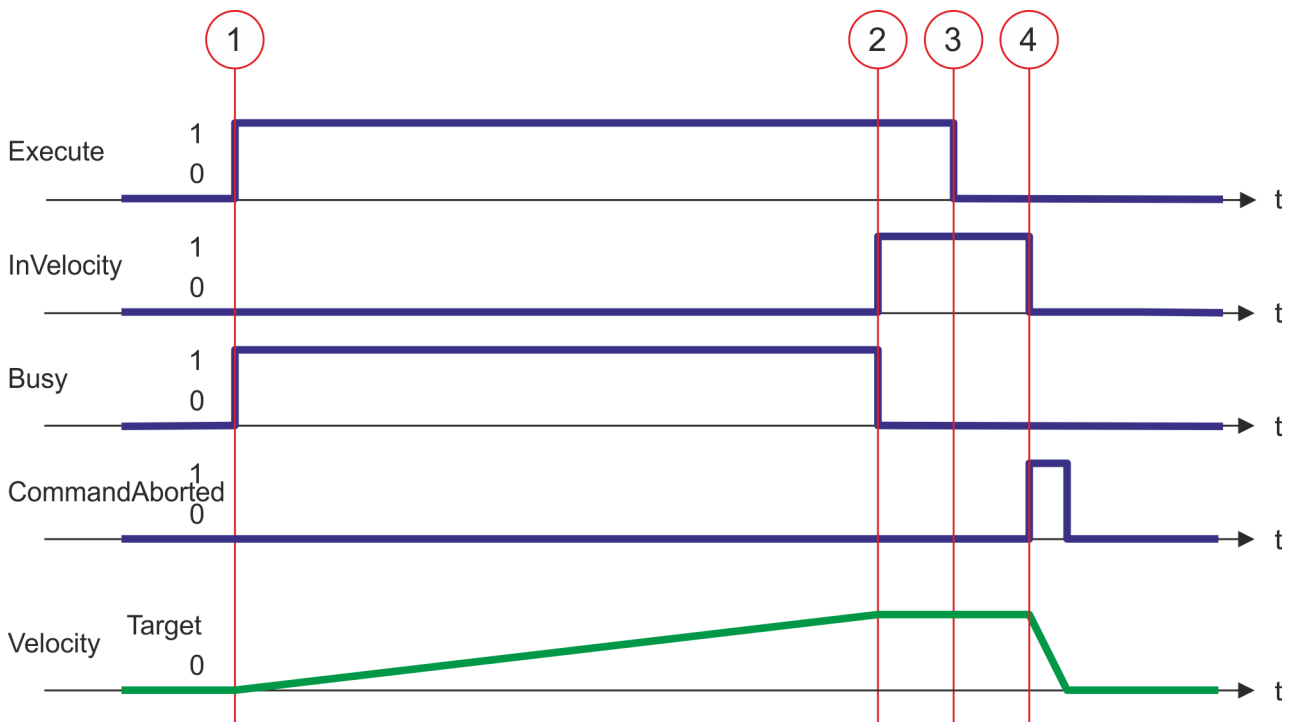
### Achse mit Geschwindigkeitsvorgabe verfahren

Mit einer Flanke 0-1 an *Execute* wird das Verfahren der Achse mit Geschwindigkeitsvorgabe gestartet. Solange die Geschwindigkeitsvorgabe nicht erreicht ist, zeigt *Busy* den Wert TRUE und *InVelocity* den Wert FALSE. Ist die Geschwindigkeitsvorgabe erreicht, wird *Busy* gleich FALSE und *InVelocity* gleich TRUE. Die Achse wird mit dieser Geschwindigkeit konstant weiter verfahren.



- Ein laufender Auftrag wird auch beim Setzen von *Execute* gleich FALSE weiterhin ausgeführt, auch wenn die Geschwindigkeitsvorgabe erreicht wurde.
- Ein laufender Auftrag kann durch einen anderen Bewegungsauftrag (z.B. *MC\_MoveAbsolute*) abgebrochen werden.

### Zustandsdiagramm der Bausteinparameter



- (1) Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird das Verfahren der Achse mit Geschwindigkeitsvorgabe gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) erreicht die Achse die Geschwindigkeitsvorgabe und *Busy* liefert den Wert FALSE und *InVelocity* den Wert TRUE.
- (3) Das Rücksetzen von *Execute* auf FALSE zum Zeitpunkt (3) hat keine Auswirkung auf die Achse. Die Achse wird weiterhin konstant mit der Geschwindigkeitsvorgabe verfahren und *InVelocity* liefert weiterhin den Wert TRUE.
- (4) Zum Zeitpunkt (4) wird der *MC\_Velocity*-Auftrag durch einen *MC\_Stop*-Auftrag abgebrochen. Die Achse wird bis zum Stillstand abgebremst.

### 4.3.9 FB 808 - MC\_MoveAbsolute - Achse auf absolute Position verfahren

**Beschreibung** Mit MC\_MoveAbsolute wird die Achse auf eine absolute Position verfahren. Mit den Parametern *Velocity*, *Acceleration* und *Deceleration* wird das dynamische Verhalten beim Bewegungsvorgang bestimmt.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Verfahren der Achse starten               <ul style="list-style-type: none"> <li>– Flanke 0-1: Das Verfahren der Achse wird gestartet</li> </ul> </li> </ul>
Position	INPUT	REAL	Absolute Position in [Anwendereinheiten]
Velocity	INPUT	REAL	Maximale Geschwindigkeit (muss nicht zwingend erreicht werden) in [Anwendereinheiten/s]
Acceleration	INPUT	REAL	Beschleunigung in [Anwendereinheiten/s <sup>2</sup> ]
Deceleration	INPUT	REAL	Verzögerung in [Anwendereinheiten/s <sup>2</sup> ]
Direction	INPUT	Byte	<ul style="list-style-type: none"> <li>■ Richtung               <ul style="list-style-type: none"> <li>– 0: Kürzeste Entfernung</li> <li>– 1: Positive Richtung</li> <li>– 2: Negative Richtung</li> </ul> </li> </ul>
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag erfolgreich durchgeführt. Die Zielposition wurde erreicht.</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul> </li> </ul>
Active	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Baustein hat die Kontrolle über die Achse</li> </ul> </li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Der Auftrag wurde während der Bearbeitung von einem anderen Auftrag abgebrochen</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

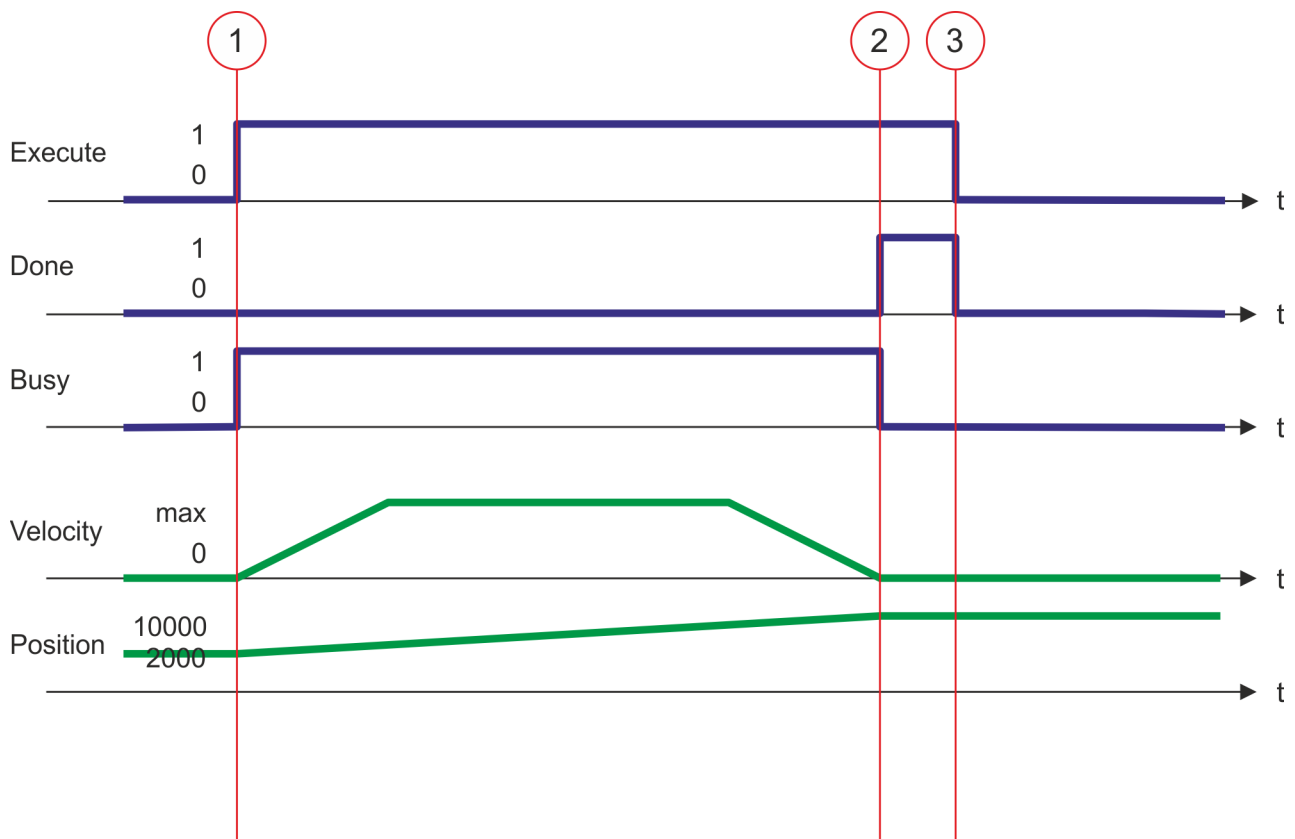
- Start des Auftrags in den PLCopen-States *Standstill*, *Discrete Motion* und *Continuous Motion* möglich.
- MC\_MoveAbsolute führt die Achse in den PLCopen-State *Discrete Motion* über.
- ↪ Kap. 5.1 "PLCopen-States" Seite 92

**Achse absolut verfahren**

Mit einer Flanke 0-1 an *Execute* wird das Verfahren der Achse gestartet. Solange das Verfahren der Achse läuft, zeigt *Busy* den Wert TRUE. Nachdem die Zielposition erreicht wurde, wird *Busy* = FALSE und *Done* = TRUE geliefert. Die Geschwindigkeit der Achse ist dann gleich 0.



- Ein laufender Auftrag wird auch beim Setzen von *Execute* gleich FALSE, bis die Achse die Zielposition erreicht hat, ausgeführt.
- Ein laufender Auftrag kann durch einen anderen Bewegungsauftrag (z.B. MC\_MoveVelocity) abgebrochen werden.

**Zustandsdiagramm der Bausteinparameter**

- (1) Die Achse wird mit MC\_MoveAbsolute auf die absolute Position = 10000.0 verfahren (Startposition bei Auftragsstart gleich 2000.0). Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird das Verfahren der Achse gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) hat die Achse die Zielposition erreicht. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.

### 4.3.10 FB 811 - MC\_Reset - Reset Achse

#### Beschreibung

Mit MC\_Reset können Sie einen Reset (Neuinitialisieren) der Achse durchführen. Dabei werden alle internen Fehler der Achse zurückgesetzt.



*Bitte beachten Sie, dass solange der Antrieb noch nicht initialisiert ist, insbesondere bei der Erstinbetriebnahme, der FB 800 - MC\_Power die Fehlermeldung 0x8103 auslösen kann. Mit dem Aufruf des FB 849 - Y\_Init wird dieser Fehler automatisch zurückgesetzt. Das Zurücksetzen des Fehlers mit dem FB 811 - MC\_Reset ist nicht möglich.*

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Reset Achse               <ul style="list-style-type: none"> <li>– Flanke 0-1: Reset der Achse wird durchgeführt</li> </ul> </li> </ul>
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag erfolgreich durchgeführt. Reset wurde durchgeführt</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↗ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

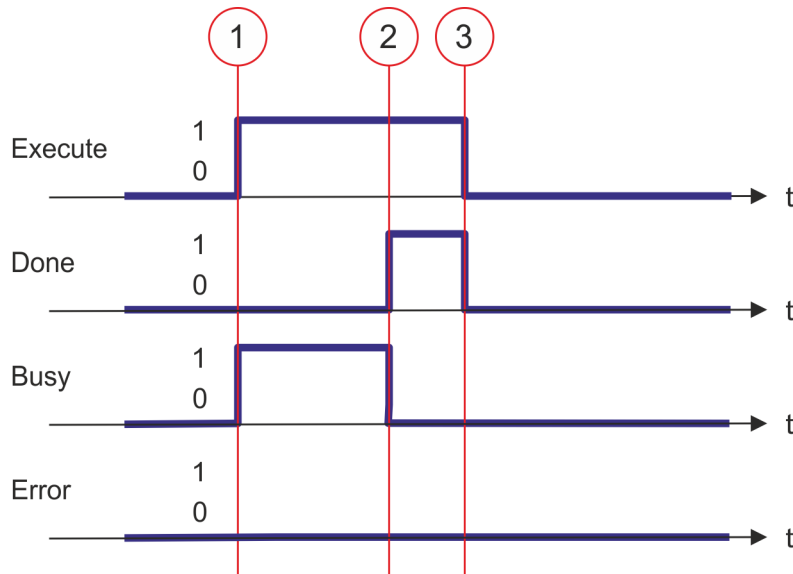
- Start des Auftrags im PLCopen-State *ErrorStop*, *Disabled* und *Standstill* möglich. Folgende Meldungen können zurückgesetzt werden:
  - Warnungen
  - Fehler
- MC\_Reset führt die Achse in Abhängigkeit von MC\_Power entweder in den PLCopen-State *Standstill* (Aufruf von MC\_Power mit *Enable* = TRUE) oder *Disabled* (Aufruf von MC\_Power mit *Enable* = FALSE) über.
- ↗ Kap. 5.1 "PLCopen-States" Seite 92

#### Reset an Achse durchführen

Mit einer Flanke 0-1 an *Execute* wird der Reset der Achse gestartet. Solange der Reset der Achse läuft, zeigt *Busy* den Wert TRUE. Nachdem die Achse neu initialisiert wurde, wird *Busy* gleich FALSE und *Done* gleich TRUE geliefert.



*Ein laufender Auftrag wird auch beim Setzen von Execute gleich FALSE ausgeführt, bis der Auftrag abgeschlossen ist.*

**Zustandsdiagramm der Bausteinparameter**

- (1) Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird der Reset der Achse gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) ist der Reset erfolgreich abgeschlossen. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.

### 4.3.11 FB 812 - MC\_ReadStatus - Status Achse lesen

**Beschreibung** Mit MC\_ReadStatus kann der PLCopen-State der Achse ermittelt werden.

#### Parameter

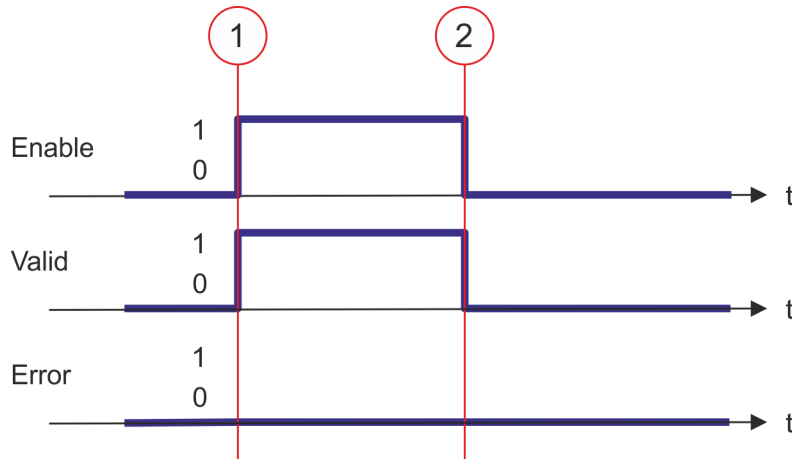
Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Statusanzeige               <ul style="list-style-type: none"> <li>– TRUE: Der Status wird an den Ausgängen permanent angezeigt</li> <li>– FALSE: Alle Ausgänge werden gleich FALSE bzw. 0 geliefert</li> </ul> </li> </ul>
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status gültig               <ul style="list-style-type: none"> <li>– TRUE: Der angezeigte Status ist gültig</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Disabled	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status Achse: Sperrung               <ul style="list-style-type: none"> <li>– TRUE: Achse ist gesperrt; ein Bewegungsauftrag kann nicht aktiviert werden</li> </ul> </li> </ul>
Standstill	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status Bewegungsauftrag               <ul style="list-style-type: none"> <li>– TRUE: Kein Bewegungsauftrag aktiv; Bewegungsauftrag kann aktiviert werden</li> </ul> </li> </ul>
Homing	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status Achse: Referenzierung               <ul style="list-style-type: none"> <li>– TRUE: Achse wird referenziert (MC_Homing ist aktiv)</li> </ul> </li> </ul>
DiscreteMotion	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status Achsbewegung: Diskret               <ul style="list-style-type: none"> <li>– TRUE: Achse wird durch eine diskrete Bewegung verfahren (MC_MoveRelative, MC_MoveAbsolute oder MC_Halt ist aktiv)</li> </ul> </li> </ul>
ContinuousMotion	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status Achsbewegung: Kontinuierlich               <ul style="list-style-type: none"> <li>– TRUE: Achse wird durch eine kontinuierliche Bewegung verfahren (MC_MoveVelocity ist aktiv)</li> </ul> </li> </ul>
SynchronizedMotion	OUTPUT	BOOL	Dieser Parameter wird aktuell nicht verwendet.
Stopping	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status Achse: Stop               <ul style="list-style-type: none"> <li>– TRUE: Achse wird gestoppt (MC_Stop ist aktiv)</li> </ul> </li> </ul>
ErrorStop	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Achsfehler               <ul style="list-style-type: none"> <li>– TRUE: Achsfehler aufgetreten; ein Bewegungsauftrag kann nicht aktiviert werden</li> </ul> </li> </ul>
Axis	IN_OUT	STRUCT	Referenz zur Slave-Achse

#### PLCopen-State

- Start des Auftrags in jedem PLCopen-State möglich.
- ↪ *Kap. 5.1 "PLCopen-States" Seite 92*

**Status der Achse ermitteln** Mit *Enable* = TRUE wird an den Ausgängen der Zustand der Achse entsprechend dem Zustandsdiagramm nach PLCopen geliefert.

### Zustandsdiagramm der Bausteinparameter



- (1) Zum Zeitpunkt (1) wird *Enable* = TRUE gesetzt. Damit liefert *Valid* den Wert TRUE und an den Ausgängen wird der Zustand entsprechend des PLCopen-Zustandsdiagramms angezeigt.
- (2) Zum Zeitpunkt (2) wird *Enable* = FALSE gesetzt. Damit werden sämtliche Ausgänge gleich FALSE bzw. 0 gesetzt.



### 4.3.12 FB 813 - MC\_ReadAxisError - Fehler von Achse lesen

**Beschreibung** Mit MC\_ReadAxisError wird der aktuell anstehende Fehler direkt vom Antrieb gelesen.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Der FB liefert kontinuierlich Ausgabe-Daten zurück.</li> </ul>
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Im FB sind gültige Ausgabe-Daten verfügbar.</li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag ist in Bearbeitung.</li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen <ul style="list-style-type: none"> <li>↳ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</li> </ul>
AxisErrorID	OUTPUT	WORD	Achsfehler-ID; der gelieferte Wert ist Hersteller-spezifisch kodiert.
AxisWarningID	OUTPUT	WORD	Achswarnung-ID; der gelieferte Wert ist Hersteller-spezifisch kodiert.
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

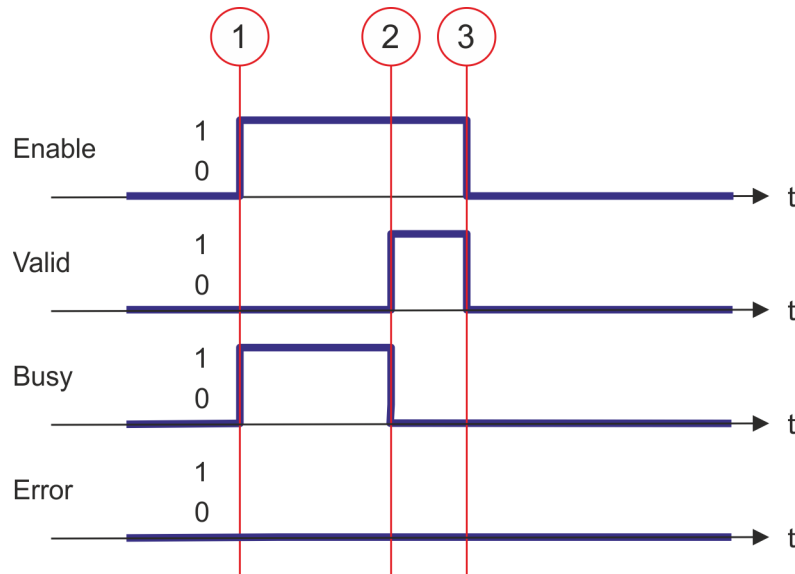
- Start des Auftrags in jedem PLCopen-State möglich.
- ↳ Kap. 5.1 "PLCopen-States" Seite 92

#### Fehler der Achse lesen

Mit einer Flanke 0-1 an *Enable* wird das Lesen des Achsfehlers gestartet. Solange das Lesen des Achsfehlers läuft, zeigt *Busy* den Wert TRUE. Nachdem der Achsfehler gelesen wurde, wird *Busy* gleich FALSE und *Valid* gleich TRUE geliefert. Der Ausgang *AxisErrorID* zeigt den aktuell anstehenden Achsfehler an.



Ein laufender Auftrag wird auch beim Setzen von *Enable* gleich FALSE weiterhin ausgeführt.

**Zustandsdiagramm der Bausteinparameter**

- (1) Mit der Flanke 0-1 an *Enable* zum Zeitpunkt (1) wird das Lesen des Achsfehlers gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) ist das Lesen des Achsfehlers erfolgreich abgeschlossen. *Busy* liefert den Wert FALSE und *Valid* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Enable* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.

### 4.3.13 FB 816 - MC\_ReadActualPosition - Aktuelle Position der Achse lesen

**Beschreibung** Mit MC\_ReadActualPosition wird die aktuelle Position der Achse gelesen.

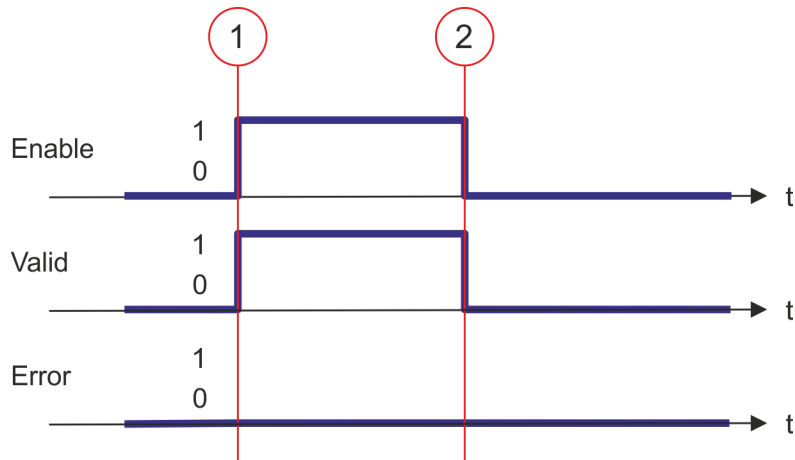
#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Position Achse lesen               <ul style="list-style-type: none"> <li>– TRUE: Die Position der Achse wird kontinuierlich gelesen</li> <li>– FALSE: Alle Ausgänge werden gleich FALSE bzw. 0 geliefert</li> </ul> </li> </ul>
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Position gültig               <ul style="list-style-type: none"> <li>– TRUE: Die gelesene Position ist gültig</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung.</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Position	OUTPUT	REAL	Absolute Position der Achse in [Anwendereinheiten]
Axis	IN_OUT	STRUCT	Referenz zur Achse

**PLCopen-State**

- Start des Auftrags in jedem PLCopen-State möglich.
- ↪ *Kap. 5.1 "PLCopen-States" Seite 92*

**Position der Achse lesen** Mit *Enable* gleich TRUE wird die aktuelle Position der Achse ermittelt und unter *Position* abgelegt.

**Zustandsdiagramm der Bausteinparameter**

- (1) Zum Zeitpunkt (1) wird *Enable* = TRUE gesetzt. Damit liefert *Valid* den Wert TRUE und am Ausgang *Position* wird die aktuelle Position der Achse angezeigt.
- (2) Zum Zeitpunkt (2) wird *Enable* = FALSE gesetzt. Damit werden sämtliche Ausgänge gleich FALSE bzw. 0 gesetzt.

### 4.3.14 FB 817 - MC\_ReadActualVelocity - Aktuelle Geschwindigkeit der Achse lesen

**Beschreibung** Mit MC\_ReadActualVelocity wird die aktuelle Geschwindigkeit der Achse gelesen.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Geschwindigkeit Achse lesen               <ul style="list-style-type: none"> <li>– TRUE: Die Geschwindigkeit der Achse wird kontinuierlich gelesen</li> <li>– FALSE: Alle Ausgänge werden gleich FALSE bzw. 0 geliefert</li> </ul> </li> </ul>
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Geschwindigkeit gültig               <ul style="list-style-type: none"> <li>– TRUE: Die gelesene Geschwindigkeit ist gültig</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung.</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Velocity	OUTPUT	REAL	Aktuelle Geschwindigkeit der Achse in [Anwendereinheiten/s]
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

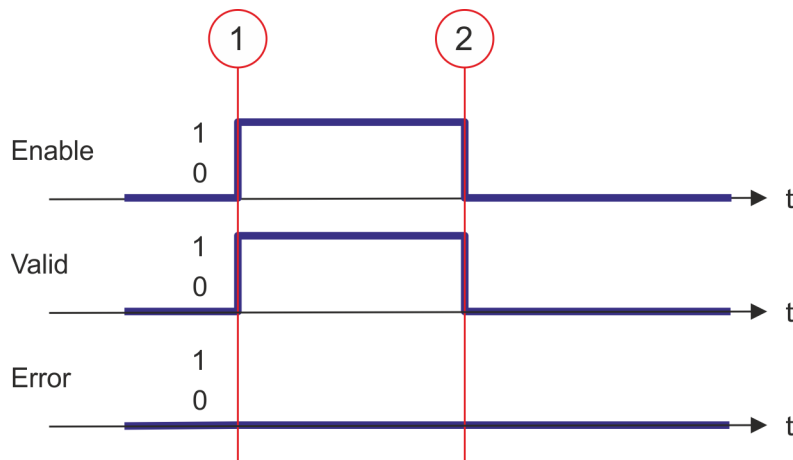
- Start des Auftrags in jedem PLCopen-State möglich.
- ↪ *Kap. 5.1 "PLCopen-States" Seite 92*

#### Geschwindigkeit der Achse lesen

Mit *Enable* gleich TRUE wird die aktuelle Geschwindigkeit der Achse ermittelt und unter *Velocity* abgelegt.

Komplexe Bewegungsaufgaben - PLCopen-Bausteine > FB 817 - MC\_ReadActualVelocity - Aktuelle Geschwindigkeit der Achse lesen

### Zustandsdiagramm der Bausteinparameter



- (1) Zum Zeitpunkt (1) wird *Enable* = TRUE gesetzt. Damit liefert *Valid* den Wert TRUE und am Ausgang *Velocity* wird die aktuelle Geschwindigkeit der Achse angezeigt.
- (2) Zum Zeitpunkt (2) wird *Enable* = FALSE gesetzt. Damit werden sämtliche Ausgänge gleich FALSE bzw. 0 gesetzt.

### 4.3.15 FB 818 - MC\_ReadAxisInfo - Zusatzinformationen der Achse lesen

**Beschreibung** Mit MC\_ReadAxisInfo werden einige Zusatzinformationen der Achse angezeigt.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Zusatzinformationen Achse lesen               <ul style="list-style-type: none"> <li>– TRUE: Die Zusatzinformationen der Achse werden kontinuierlich gelesen</li> <li>– FALSE: Alle Ausgänge werden gleich FALSE bzw. 0 geliefert</li> </ul> </li> </ul>
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Zusatzinformationen gültig               <ul style="list-style-type: none"> <li>– TRUE: Die gelesene Zusatzinformationen sind gültig</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↗ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96
HomeAbsSwitch	OUTPUT	BOOL	Referenzschalter <ul style="list-style-type: none"> <li>■ TRUE: Der Referenzschalter ist aktiviert</li> </ul>
LimitSwitchPos	OUTPUT	BOOL	Endschalter positive Richtung <ul style="list-style-type: none"> <li>■ TRUE: Endschalter positive Richtung ist aktiviert</li> </ul>
LimitSwitchNeg	OUTPUT	BOOL	Endschalter negative Richtung (NOT-Bit am Antrieb) <ul style="list-style-type: none"> <li>■ TRUE: Endschalter negative Richtung ist aktiviert</li> </ul>
Simulation	OUTPUT	BOOL	Parameter aktuell nicht unterstützt; immer FALSE
CommunicationReady	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Information Achse: Datenaustausch               <ul style="list-style-type: none"> <li>– TRUE: Datenaustausch mit der Achse initialisiert; Achse ist kommunikationsbereit</li> </ul> </li> </ul>
ReadyForPowerOn	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Information Achse: Freigabe möglich               <ul style="list-style-type: none"> <li>– TRUE: Die Freigabe der Achse ist möglich</li> </ul> </li> </ul>
PowerOn	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Information Achse: freigegeben               <ul style="list-style-type: none"> <li>– TRUE: Die Freigabe der Achse ist erfolgt</li> </ul> </li> </ul>
IsHomed	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Information Achse: referenziert               <ul style="list-style-type: none"> <li>– TRUE: Die Achse ist referenziert</li> </ul> </li> </ul>
AxisWarning	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Information Achse: Fehler               <ul style="list-style-type: none"> <li>– TRUE: Mindestens 1 Fehler wird von der Achse gemeldet</li> </ul> </li> </ul>
SwLimitSwitchPos	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Softwareendschalter positive Richtung               <ul style="list-style-type: none"> <li>– TRUE: Softwareendschalter in positive Richtung ist aktiviert.</li> </ul> </li> </ul>
SwLimitSwitchNeg	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Softwareendschalter negative Richtung               <ul style="list-style-type: none"> <li>– TRUE: Softwareendschalter in negative Richtung ist aktiviert.</li> </ul> </li> </ul>

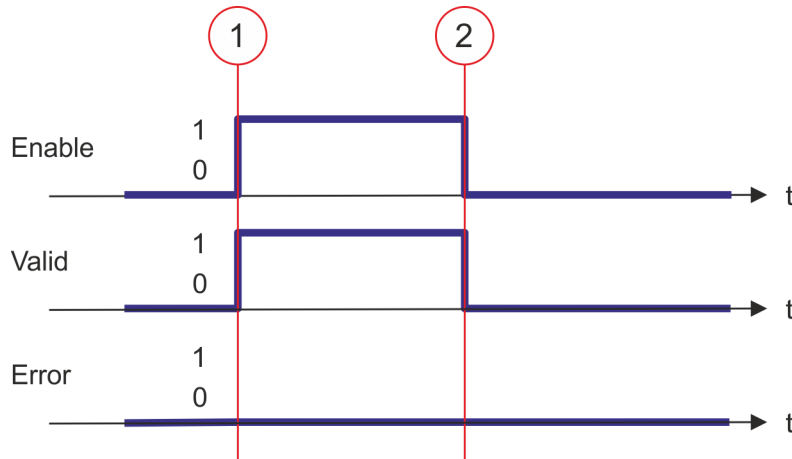
Parameter	Deklaration	Datentyp	Beschreibung
TorqueLimit	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>Information Achse: Maximales Drehmoment</li> <li>– TRUE: Maximales Drehmoment ist erreicht.</li> </ul>
SafetyActive	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>Information Achse: Safety-Status</li> <li>– TRUE: HWBB bzw. Safety-Funktion ist aktiviert.</li> </ul>
Axis	IN_OUT	STRUCT	Referenz zur Achse

**PLCopen-State**

- Start des Auftrags in jedem PLCopen-State möglich.
- ↪ Kap. 5.1 "PLCopen-States" Seite 92

**Status der Achse ermitteln**

Mit *Enable* gleich TRUE werden an den Ausgängen die Zusatzinformationen zur Achse geliefert.

**Zustandsdiagramm der Bausteinparameter**

- (1) Zum Zeitpunkt (1) wird *Enable* = TRUE gesetzt. Damit liefert *Valid* den Wert TRUE und an den Ausgängen werden die Zusatzinformationen zur Achse angezeigt.
- (2) Zum Zeitpunkt (2) wird *Enable* = FALSE gesetzt. Damit werden sämtliche Ausgänge gleich FALSE bzw. 0 gesetzt.



### 4.3.16 FB 819 - MC\_ReadMotionState - Zustand Bewegungsauftrag lesen

**Beschreibung** Mit MC\_ReadMotionState wird der aktuelle Zustand des Bewegungsauftrags angezeigt.

#### Parameter

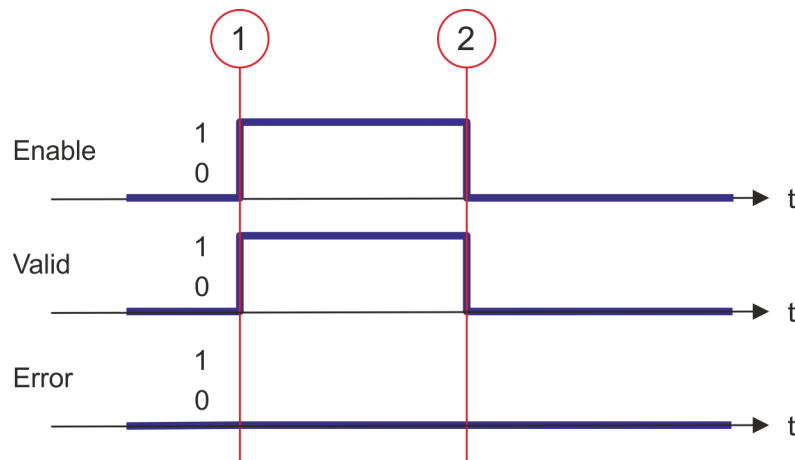
Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Zustand Bewegungsauftrag lesen               <ul style="list-style-type: none"> <li>– TRUE: Zustand Bewegungsauftrag wird kontinuierlich gelesen</li> <li>– FALSE: Alle Ausgänge werden gleich FALSE bzw. 0 geliefert</li> </ul> </li> </ul>
Source	INPUT	Byte	Nur Source = 0 wird unterstützt; an den Ausgängen werden die Istzustände des Bewegungsauftrags angezeigt.
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Zustand gültig               <ul style="list-style-type: none"> <li>– TRUE: Der gelesene Zustand des Bewegungsauftrags ist gültig</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
ConstantVelocity	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Zustand Bewegungsauftrag: Geschwindigkeit               <ul style="list-style-type: none"> <li>– TRUE: Geschwindigkeit ist konstant</li> </ul> </li> </ul>
Accelerating	OUTPUT	BOOL	Bitte beachten Sie, dass dieser Parameter bei Einsatz von Frequenzumrichter über EtherCAT nicht unterstützt wird! <ul style="list-style-type: none"> <li>■ Zustand Bewegungsauftrag: Beschleunigung               <ul style="list-style-type: none"> <li>– TRUE: Achse wird beschleunigt; die Geschwindigkeit der Achse erhöht sich.</li> </ul> </li> </ul>
Decelerating	OUTPUT	BOOL	Bitte beachten Sie, dass dieser Parameter bei Einsatz von Frequenzumrichter über EtherCAT nicht unterstützt wird! <ul style="list-style-type: none"> <li>■ Zustand Bewegungsauftrag: Bremsvorgang               <ul style="list-style-type: none"> <li>– TRUE: Achse wird gebremst; die Geschwindigkeit der Achse wird geringer.</li> </ul> </li> </ul>
DirectionPositive	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Zustand Bewegungsauftrag: Position zunehmend               <ul style="list-style-type: none"> <li>– TRUE: Die Position der Achse nimmt zu</li> </ul> </li> </ul>
DirectionNegative	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Zustand Bewegungsauftrag: Position abnehmend               <ul style="list-style-type: none"> <li>– TRUE: Die Position der Achse nimmt ab</li> </ul> </li> </ul>
Axis	IN_OUT	STRUCT	Referenz zur Achse

#### PLCopen-State

- Start des Auftrags in jedem PLCopen-State möglich.
- ↪ *Kap. 5.1 "PLCopen-States" Seite 92*

#### Zustand des Bewegungsauftrags lesen

Mit *Enable* gleich TRUE wird an den Ausgängen der Zustand des Bewegungsauftrags der Achse geliefert.

**Zustandsdiagramm der Bausteinparameter**

- (1) Zum Zeitpunkt (1) wird *Enable* = TRUE gesetzt. Damit liefert *Valid* den Wert TRUE und an den Ausgängen wird der Zustand des Bewegungsauftrags angezeigt.
- (2) Zum Zeitpunkt (2) wird *Enable* = FALSE gesetzt. Damit werden sämtliche Ausgänge gleich FALSE bzw. 0 gesetzt.

### 4.3.17 FB 823 - MC\_TouchProbe - Achsposition erfassen

#### Beschreibung

Dieser Baustein erfasst einmalig die Achsposition in Abhängigkeit eines Trigger-Signals. Das Trigger-Signal kann über die am Eingang *TriggerInput* angegebene Variable konfiguriert werden. Als Trigger-Signal kann z.B. ein Digitaleingang oder die Gebernulldspuren dienen.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	Mit einer Flanke 0-1 an <i>Execute</i> wird die Erfassung der Achsposition aktiviert.
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag erfolgreich durchgeführt. Die Achsposition wurde erfasst.</li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag ist in Bearbeitung.</li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Der Auftrag wurde während der Bearbeitung von einem anderen Auftrag abgebrochen.</li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen <a href="#">↪ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</a>
RecordedPosition	OUTPUT	REAL	Erfasste Achsposition zum Zeitpunkt des Trigger-Signals in [Anwendereinheiten].
Axis	IN_OUT	STRUCT	Referenz zur Achse.
TriggerInput	IN_OUT	STRUCT	Referenz zum Trigger-Eingang. Struktur <ul style="list-style-type: none"> <li>■ .Probe <ul style="list-style-type: none"> <li>– 01: TouchProbe-Register 1</li> <li>– 02: TouchProbe-Register 2</li> </ul> </li> <li>■ .TriggerSource <ul style="list-style-type: none"> <li>– 00: Eingang</li> <li>– 01: Encoder Nullimpuls</li> </ul> </li> <li>■ .TriggerMode <ul style="list-style-type: none"> <li>– 00: SingleTrigger</li> </ul> </li> <li>■ .Reserviert</li> </ul>

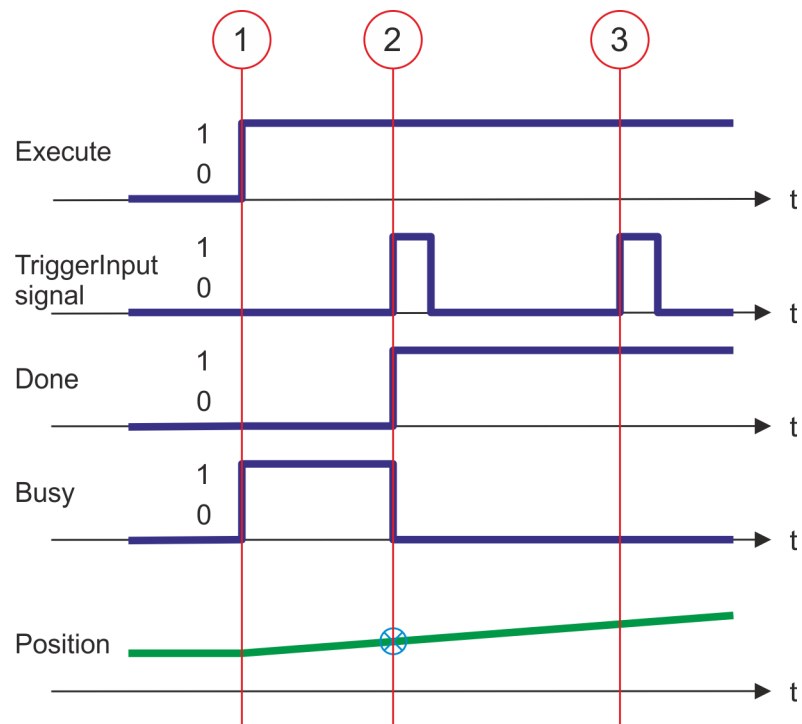


- Ein laufender Auftrag wird auch beim Setzen von *Execute* gleich *FALSE*, solange ausgeführt, bis dieser abgearbeitet ist. Die erfasste Achsposition wird dann für einen Zyklus am Ausgang *RecordedPosition* ausgegeben. ↪ Kap. 5.2 "Verhalten der Ein- und Ausgänge" Seite 94
- Damit der Befehl ausgeführt werden kann, muss die Kommunikation mit der Achse OK und der PLCopen-State ungleich *Homing* sein.
- Ein laufender Auftrag kann durch einen neuen *MC\_TouchProbe* auf der gleichen Achse abgebrochen werden.
- Ein laufender Auftrag kann durch den Befehl *MC\_AbortTrigger* abgebrochen werden.
- Ein laufender Auftrag kann durch den Befehl *MC\_Home* abgebrochen werden.

### Achsposition erfassen

Mit einer Flanke 0-1 an *Execute* wird die Erfassung der Achsposition aktiviert. Solange der Befehl abgearbeitet wird, zeigt *Busy* den Wert *TRUE*. Nach Abarbeitung des Befehls, wird *Busy* gleich *FALSE* und *Done* gleich *TRUE* geliefert. Der erfasste Wert wird in *RecordedPosition* ausgegeben.

### Zustandsdiagramm der Bausteinparameter



- (1) Mit einer Flanke 0-1 an *Execute* wird die Erfassung der Achsposition aktiviert und *Busy* liefert den Wert *TRUE*.
- (2) Mit einer Flanke 0-1 an *TriggerInput* wird die Achsposition erfasst und in *RecordedPosition* abgelegt. Die Erfassung ist beendet. *Done* liefert *TRUE* und *Busy* *FALSE*.
- (3) Da die einmalige Erfassung beendet ist, werden weitere Triggersignale ignoriert.

### 4.3.18 FB 824 - MC\_AbortTrigger - Achsposition erfassen abbrechen

**Beschreibung** Dieser Baustein bricht die durch MC\_TouchProbe gestartete Erfassung der Achsposition ab.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	IN	BOOL	Mit einer Flanke 0-1 an <i>Execute</i> wird die Erfassung der Achsposition abgebrochen.
Done	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag erfolgreich durchgeführt. Die Erfassung der Achsposition wurde abgebrochen.</li> </ul>
Busy	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Auftrag ist in Bearbeitung.</li> </ul>
Error	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status</li> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul>
ErrorID	OUT	WORD	Zusätzliche Fehlerinformationen ↪ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96
Axis	IN_OUT	STRUCT	Referenz zur Achse.
TriggerInput	IN_OUT	STRUCT	Referenz zum Trigger-Eingang. Struktur <ul style="list-style-type: none"> <li>■ .Probe               <ul style="list-style-type: none"> <li>– 01: TouchProbe-Register 1</li> <li>– 02: TouchProbe-Register 2</li> </ul> </li> <li>■ .TriggerSource               <ul style="list-style-type: none"> <li>– 00: Eingang</li> <li>– 01: Encoder Nullimpuls</li> </ul> </li> <li>■ .TriggerMode               <ul style="list-style-type: none"> <li>– 00: SingleTrigger</li> </ul> </li> <li>■ .Reserviert</li> </ul>



Damit der Befehl ausgeführt werden kann, muss die Kommunikation mit der Achse OK sein.

#### Erfassung der Achsposition abbrechen

Mit einer Flanke 0-1 an *Execute* wird die Erfassung der Achsposition abgebrochen. Solange der Befehl abgearbeitet wird, zeigt *Busy* den Wert TRUE. Nach Abarbeitung des Befehls, wird *Busy* gleich FALSE und *Done* gleich TRUE geliefert.

### 4.3.19 FB 833 - Y\_ReadParameter - Antriebsparameter lesen

**Beschreibung** Mit dem Y\_ReadParameter wird ein Parameterwert aus dem angeordneten Antrieb gelesen.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Antriebsparameter lesen               <ul style="list-style-type: none"> <li>– Flanke 0-1: Das Lesen des Antriebsparameters wird durchgeführt.</li> </ul> </li> </ul>
Index	INPUT	WORD	Index des Antriebsparameters
Subindex	INPUT	BYTE	Subindex des Antriebsparameters
DIAG_BUF	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Diagnosepuffer               <ul style="list-style-type: none"> <li>– TRUE: CPU Eintrag Diagnosepuffer (relevant für Servicepersonal)</li> </ul> </li> </ul>
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag erfolgreich durchgeführt. Parameter wurde ausgelesen</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
ReadValue	OUTPUT	DWORD	Wert des gelesenen Parameters
Axis	IN_OUT	STRUCT	Referenz zur Achse

**PLCopen-State**

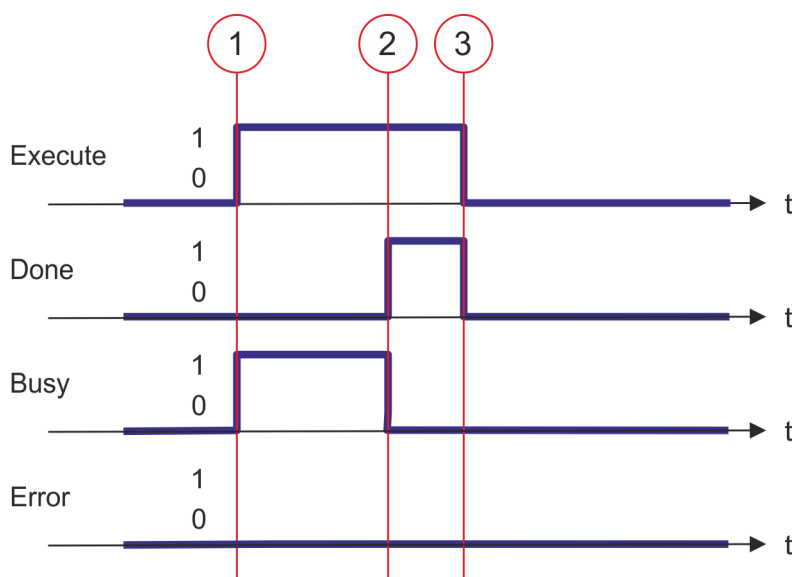
- Start des Auftrags in jedem PLCopen-State möglich.
- ↪ *Kap. 5.1 "PLCopen-States" Seite 92*

**Antriebsparameter lesen**

Mit einer Flanke 0-1 an *Execute* wird das Lesen des Antriebsparameters gestartet. Solange das Lesen des Parameters läuft, zeigt *Busy* den Wert TRUE. Nachdem der Parameter gelesen wurde, wird *Busy* gleich FALSE und *Done* gleich TRUE geliefert. Der Ausgang *Value* zeigt den Wert des Parameters an.



- Für den Zugriff auf die Antriebs-Parameter Pn000...Pn6FF müssen Sie einen Offset von 0x9000 zum Index addieren.  
Beispiel: Parameter Pn50A → Index 0x950A
- Parameter mit folgenden Einheiten werden konvertiert:
  - Einheit "Pos.unit" → user unit
  - Einheit "Vel.unit" → user unit/s
  - Einheit "Acc.unit" → user unit/s<sup>2</sup>
  - Einheit 0,1% Rated Motor Torque → 1,0% Rated Motor Torque
  - Einheit (0,1% Rated Motor Torque)/s → (1,0% Rated Motor Torque)/s
- Ein laufender Auftrag wird auch beim Setzen von *Execute* gleich FALSE weiterhin ausgeführt.

**Zustandsdiagramm der Bausteinparameter**

- (1) Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird das Lesen des Parameters gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) ist das Lesen des Parameters erfolgreich abgeschlossen. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.

### 4.3.20 FB 834 - Y\_WriteParameter - Antriebsparameter schreiben

#### Beschreibung

Mit dem Y\_WriteParameter wird ein Parameterwert in den angebundnen Antrieb geschrieben.



#### **Bitte folgende Parameter nicht ändern**

Beim Aufruf des Init-Bausteines Y\_SIG7PN\_Servolnit werden folgende Parameter angepasst. Diese sollten Sie nicht ändern:

- PnC00 ... PnC0F - Setpoint telegram: PZD 1 ... 16
- PnC10 ... PnC1F - Actual value telegram: PZD 1 ... 16
- PnC20 - Telegram selection
- PnB02 - Position user unit: Numerator
- PnB04 - Position user unit: Denominator
- PnB06 - Velocity user unit: Numerator
- PnB08 - Velocity user unit: Denominator
- PnB0A - Acceleration user unit: Numerator
- PnB42 - Position range limit (min.)
- PnB44 - Position range limit (max.)
- PnB48 - Software position limit (min.)
- PnB4A - Software position limit (max.)
- PnB0C - Acceleration user unit: Denominator
- PnB4C (607Fh): Max. profile velocity (Default: Bezug auf PnBF0 (2312h): Max. motor speed)
- PnB7C (60C5h): Max. acceleration (Default: Bezug auf PnBF2 (2313h): Max. motor acceleration)
- PnB7E (60C6h): Max. deceleration (Default: Bezug auf PnBF2 (2313h): Max. motor acceleration)
- Pn205 - Multiturn limit

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Antriebsparameter schreiben               <ul style="list-style-type: none"> <li>– Flanke 0-1: Das Schreiben des Antriebsparameters wird durchgeführt.</li> </ul> </li> </ul>
Index	INPUT	WORD	Index des Antriebsparameters.
Subindex	INPUT	BYTE	Subindex des Antriebsparameters.
SetValue	INPUT	DINT	Rücksetzen der Warteschlange und Initialisierung des Puffers mit <i>SetValue</i> .
DIAG_BUF	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Diagnosepuffer               <ul style="list-style-type: none"> <li>– TRUE: CPU-Diagnosepuffer aktiv (relevant für den Service).</li> </ul> </li> </ul>
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag erfolgreich durchgeführt. Parameter wurde ausgelesen.</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Auftrag ist in Bearbeitung.</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>



Parameter	Deklaration	Datentyp	Beschreibung
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↳ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96
Axis	IN_OUT	STRUCT	Referenz zur Achse.

**PLCopen-State**

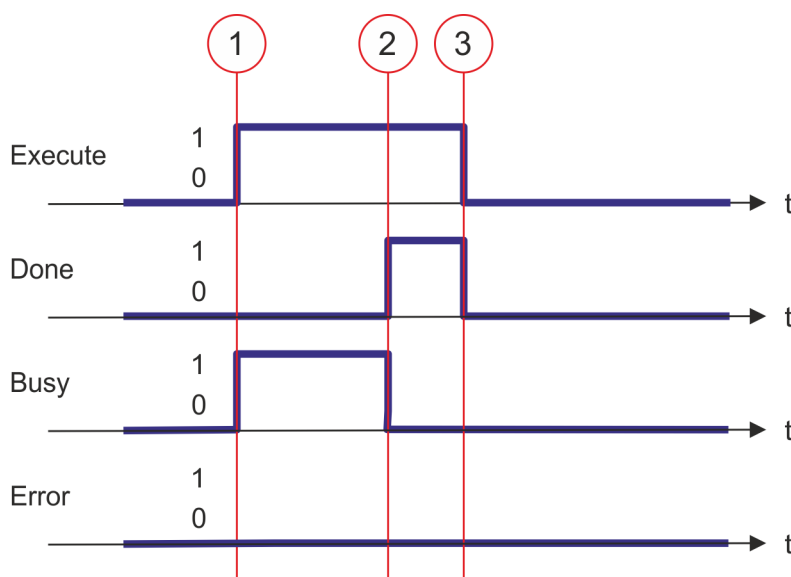
- Start des Auftrags in jedem PLCopen-State möglich.
- ↳ Kap. 5.1 "PLCopen-States" Seite 92

**Antriebsparameter schreiben**

Mit einer Flanke 0-1 an *Execute* wird das Schreiben des Parameters gestartet. Solange das Schreiben des Parameters läuft, zeigt *Busy* den Wert TRUE. Nachdem der Parameter geschrieben wurde, wird *Busy* gleich FALSE und *Done* gleich TRUE geliefert.



- Für den Zugriff auf die Antriebs-Parameter Pn000...Pn6FF müssen Sie einen Offset von 0x9000 zum Index addieren.  
Beispiel: Parameter Pn50A → Index 0x950A
- Ein laufender Auftrag wird auch beim Setzen von *Execute* gleich FALSE weiterhin ausgeführt.

**Zustandsdiagramm der Bausteinparameter**

- (1) Mit der Flanke 0-1 an *Execute* zum Zeitpunkt (1) wird das Schreiben des Parameters gestartet und *Busy* liefert den Wert TRUE.
- (2) Zum Zeitpunkt (2) ist das Schreiben des Parameters erfolgreich abgeschlossen. *Busy* liefert den Wert FALSE und *Done* den Wert TRUE.
- (3) Zum Zeitpunkt (3) ist der Auftrag abgeschlossen und *Execute* wird gleich FALSE gesetzt und dadurch sämtliche Ausgangsparameter auf FALSE bzw. 0 gesetzt.

Komplexe Bewegungsaufgaben - PLCopen-Bausteine > FB 835 - Y\_Homelnit\_LimitSwitch - Initialisierung Referenzfahrt auf Endschalter

### 4.3.21 FB 835 - Y\_Homelnit\_LimitSwitch - Initialisierung Referenzfahrt auf Endschalter

#### Beschreibung

Dieser Baustein initialisiert die Referenzfahrt (Homing) auf den Endschalter.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Initialisierung der Referenzfahrt Methode               <ul style="list-style-type: none"> <li>– Flanke 0-1: Werte der Eingangsparameter werden übernommen und die Initialisierung der Referenzfahrt Methode gestartet.</li> </ul> </li> </ul>
Direction	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Richtung der Referenzfahrt               <ul style="list-style-type: none"> <li>– TRUE: auf positiven Endschalter</li> <li>– FALSE: auf negativen Endschalter</li> </ul> </li> </ul>
WithIndexPulse	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Referenzfahrt               <ul style="list-style-type: none"> <li>– TRUE: mit Nullimpuls</li> <li>– FALSE: ohne Nullimpuls</li> </ul> </li> </ul>
VelocitySearchSwitch	INPUT	REAL	Geschwindigkeit für die Suche nach dem Schalter in [Anwendereinheiten/s]
VelocitySearchZero	INPUT	REAL	Geschwindigkeit für die Suche nach dem Index in [Anwendereinheiten/s]
Acceleration	INPUT	REAL	Beschleunigung in [Anwendereinheiten/s <sup>2</sup> ]
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung wurde ohne Fehler beendet.</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung ist aktiv</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen <i>☞ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Axis	IN_OUT	STRUCT	Referenz zur Achse

**Initialisierung Referenzfahrt auf Endschalter**

Mit einer Flanke 0-1 an *Execute* werden die Werte der Eingangsparameter übernommen und die Initialisierung der Referenzfahrt Methode gestartet. So lange die Initialisierung aktiv ist, wird der Ausgang *Busy* auf TRUE gesetzt. Ist die Initialisierung ohne Fehler beendet worden, wird der Ausgang *Done* auf TRUE gesetzt. Tritt bei der Initialisierung ein Fehler auf, wird der Ausgang *Error* auf TRUE gesetzt und am Ausgang *ErrorID* eine Fehlernummer ausgegeben.

**Initialisierung der Referenzfahrt Methode**

1. ➤ Überprüfen der Kommunikation zur Achse.
2. ➤ Prüfen auf erlaubte PLCopen Zustände.
3. ➤ Prüfung der Eingangswerte:
  - Eingang *VelocitySearchSwitch* [Anwendereinheiten]
    - $0.0 < VelocitySearchSwitch \leq \text{max. Geschwindigkeit}$
    - Die max. Geschwindigkeit können Sie mit FB 833 - Y\_ReadParameter (*Index = 2312h*) ermitteln.
    - ↗ Kap. 4.3.19 "FB 833 - Y\_ReadParameter - Antriebsparameter lesen" Seite 78
  - Eingang *VelocitySearchZero* [Anwendereinheiten]
    - Nullimpuls wird nicht verwendet: 0.0
    - Nullimpuls wird verwendet:  $0.0 < VelocitySearchZero \leq \text{max. Geschwindigkeit}$
  - Eingang *Acceleration* [Anwendereinheiten]
    - $0 < Acceleration \leq \text{max. Beschleunigung}$
    - Die max. Beschleunigung können Sie mit FB 833 - Y\_ReadParameter (*Index = 2313*) ermitteln.
    - ↗ Kap. 4.3.19 "FB 833 - Y\_ReadParameter - Antriebsparameter lesen" Seite 78
4. ➤ Übertragung der Antriebsparameter:
  - "Homing Method" Referenzfahrtmethode in Abhängigkeit vom Eingang *Direction*  
Siehe Tabelle unten!
  - "Homing Speed during search for switch" [Inc/s]  
Geschwindigkeit für die Schaltersuche
  - "Homing Speed during search for zero" [Inc/s]  
Geschwindigkeit für die Indexsuche
  - "Homing Acceleration" [Inc/s<sup>2</sup>]  
Anfahr- und Bremsbeschleunigung für die Referenzfahrt

Homing Method	Direction	WithIndexPulse
1	FALSE	TRUE
2	TRUE	TRUE
17	FALSE	FALSE
18	TRUE	FALSE

Komplexe Bewegungsaufgaben - PLCopen-Bausteine > FB 836 - Y\_Homelnit\_HomeSwitch - Initialisierung Referenzfahrt auf Referenzschalter

### 4.3.22 FB 836 - Y\_Homelnit\_HomeSwitch - Initialisierung Referenzfahrt auf Referenzschalter

#### Beschreibung

Dieser Baustein initialisiert die Referenzfahrt (Homing) auf den Referenzschalter.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Initialisierung der Referenzfahrt Methode               <ul style="list-style-type: none"> <li>– Flanke 0-1: Werte der Eingangsparameter werden übernommen und die Initialisierung der Referenzfahrt Methode gestartet.</li> </ul> </li> </ul>
WithIndexPulse	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Referenzfahrt               <ul style="list-style-type: none"> <li>– TRUE: mit Nullimpuls</li> <li>– FALSE: ohne Nullimpuls</li> </ul> </li> </ul>
SameDirIndexPulse	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Richtung Nullimpulssuche               <ul style="list-style-type: none"> <li>– TRUE: Nach dem Erkennen des Referenzschalters ohne Richtungswechsel den Nullimpuls suchen.</li> <li>– FALSE: Nach dem Erkennen des Referenzschalters einen Richtungswechsel zur Nullimpulssuche durchführen.</li> </ul> </li> </ul>
OnRisingEdge	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Flanke Referenzschalter               <ul style="list-style-type: none"> <li>– TRUE: Flanke 0-1</li> <li>– FALSE: Flanke 1-0</li> </ul> </li> </ul>
VelocitySearchSwitch	INPUT	REAL	Geschwindigkeit für die Schaltersuche in [Anwendereinheiten/s]
VelocitySearchZero	INPUT	REAL	Geschwindigkeit für die Indexsuche in [Anwendereinheiten/s]
Acceleration	INPUT	REAL	Beschleunigung in [Anwendereinheiten/s <sup>2</sup> ]
Done	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung wurde ohne Fehler beendet.</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung ist aktiv</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Axis	IN_OUT	STRUCT	Referenz zur Achse.

**Initialisierung Referenzfahrt auf Referenzschalter**

Mit einer Flanke 0-1 an *Execute* werden die Werte der Eingangsparameter übernommen und die Initialisierung der Referenzfahrt Methode gestartet. So lange die Initialisierung aktiv ist, wird der Ausgang *Busy* auf TRUE gesetzt. Ist die Initialisierung ohne Fehler beendet worden, wird der Ausgang *Done* auf TRUE gesetzt. Tritt bei der Initialisierung ein Fehler auf, wird der Ausgang *Error* auf TRUE gesetzt und am Ausgang *ErrorID* eine Fehlernummer ausgegeben.

**Initialisierung der Referenzfahrt Methode**

1. ➤ Überprüfen der Kommunikation zur Achse.
2. ➤ Prüfen auf erlaubte PLCopen Zustände.
3. ➤ Prüfung der Eingangswerte:
  - Eingang *VelocitySearchSwitch* [Anwendereinheiten]
    - $0.0 < VelocitySearchSwitch \leq \text{max. Geschwindigkeit}$
    - Die max. Geschwindigkeit können Sie mit FB 833 - Y\_ReadParameter (*Index = 2312h*) ermitteln.
    - ↪ Kap. 4.3.19 "FB 833 - Y\_ReadParameter - Antriebsparameter lesen" Seite 78
  - Eingang *VelocitySearchZero* [Anwendereinheiten]
    - Nullimpuls wird nicht verwendet: 0.0
    - Nullimpuls wird verwendet:  $0.0 < VelocitySearchZero \leq \text{max. Geschwindigkeit}$
  - Eingang *Acceleration* [Anwendereinheiten]
    - $0 < Acceleration \leq \text{max. Beschleunigung}$
    - Die max. Beschleunigung können Sie mit FB 833 - Y\_ReadParameter (*Index = 2313*) ermitteln.
    - ↪ Kap. 4.3.19 "FB 833 - Y\_ReadParameter - Antriebsparameter lesen" Seite 78
4. ➤ Übertragung der Antriebsparameter:
  - "Homing Method" Referenzfahrtmethode in Abhängigkeit vom Eingang "Direction"  
Siehe Tabelle unten!
  - "Homing Speed during search for switch" [Inc/s]  
Geschwindigkeit für die Schaltersuche
  - "Homing Speed during search for zero" [Inc/s]  
Geschwindigkeit für die Indexsuche
  - "Homing Acceleration" [Inc/s<sup>2</sup>]  
Anfahr- und Bremsbeschleunigung für die Referenzfahrt

**Parameter**

Homing Method	WithIndexPulse	OnRisingEdge	SameDirIndexPulse
3	TRUE	TRUE	FALSE
4	TRUE	TRUE	TRUE
5	TRUE	FALSE	TRUE
6	TRUE	FALSE	FALSE
19	FALSE	TRUE	FALSE
20	FALSE	TRUE	TRUE
21	FALSE	FALSE	TRUE
22	FALSE	FALSE	FALSE

Komplexe Bewegungsaufgaben - PLCopen-Bausteine > FB 837 - Y\_Homelnit\_ZeroPulse - Initialisierung Referenzfahrt auf Null Impuls

### 4.3.23 FB 837 - Y\_Homelnit\_ZeroPulse - Initialisierung Referenzfahrt auf Null Impuls

**Beschreibung** Dieser Baustein initialisiert die Referenzfahrt (Homing) auf Null Impuls.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	IN	BOOL	<ul style="list-style-type: none"> <li>■ Initialisierung der Referenzfahrt Methode               <ul style="list-style-type: none"> <li>– Flanke 0-1: Werte der Eingangsparameter werden übernommen und die Initialisierung der Referenzfahrt Methode gestartet.</li> </ul> </li> </ul>
Direction	IN	BOOL	<ul style="list-style-type: none"> <li>■ Richtung der Referenzfahrt               <ul style="list-style-type: none"> <li>– TRUE: Positive Richtung</li> <li>– FALSE: Negative Richtung</li> </ul> </li> </ul>
VelocitySearchZero	IN	REAL	Geschwindigkeit für die Indexsuche in [Anwendereinheiten/s]
Acceleration	IN	REAL	Beschleunigung in [Anwendereinheiten/s <sup>2</sup> ]
Done	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung wurde ohne Fehler beendet.</li> </ul> </li> </ul>
Busy	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung ist aktiv</li> </ul> </li> </ul>
Error	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUT	WORD	Zusätzliche Fehlerinformationen ↗ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Axis	IN_OUT	STRUCT	Referenz zur Achse.

#### Initialisierung Referenzfahrt auf Nullimpuls

Mit einer Flanke 0-1 an *Execute* werden die Werte der Eingangsparameter übernommen und die Initialisierung der Referenzfahrt Methode gestartet. So lange die Initialisierung aktiv ist, wird der Ausgang *Busy* auf TRUE gesetzt. Ist die Initialisierung ohne Fehler beendet worden, wird der Ausgang *Done* auf TRUE gesetzt. Tritt bei der Initialisierung ein Fehler auf, wird der Ausgang *Error* auf TRUE gesetzt und am Ausgang *ErrorID* eine Fehlernummer ausgegeben.

#### Initialisierung der Referenzfahrt Methode

1. ➤ Überprüfen der Kommunikation zur Achse.
2. ➤ Prüfen auf erlaubte PLCopen Zustände.

**3.** Prüfung der Eingangswerte:

- Eingang *VelocitySearchSwitch* [Anwendereinheiten]
  - $0.0 < VelocitySearchSwitch \leq \text{max. Geschwindigkeit}$
  - Die max. Geschwindigkeit können Sie mit FB 833 - Y\_ReadParameter (*Index = 2312h*) ermitteln.
  - ↪ *Kap. 4.3.19 "FB 833 - Y\_ReadParameter - Antriebsparameter lesen" Seite 78*
- Eingang *VelocitySearchZero* [Anwendereinheiten]
  - Nullimpuls wird nicht verwendet: 0.0
  - Nullimpuls wird verwendet:  $0.0 < VelocitySearchZero \leq \text{max. Geschwindigkeit}$
- Eingang *Acceleration* [Anwendereinheiten]
  - $0 < Acceleration \leq \text{max. Beschleunigung}$
  - Die max. Beschleunigung können Sie mit FB 833 - Y\_ReadParameter (*Index = 2313*) ermitteln.
  - ↪ *Kap. 4.3.19 "FB 833 - Y\_ReadParameter - Antriebsparameter lesen" Seite 78*

**4.** Übertragung der Antriebsparameter:

- "Homing Method" Referenzfahrtmethode in Abhängigkeit vom Eingang "Direction". Siehe Tabelle unten!
- "Homing Speed during search for switch" [Inc/s]  
Geschwindigkeit für die Schaltersuche
- "Homing Speed during search for zero" [Inc/s]  
Geschwindigkeit für die Indexsuche
- "Homing Acceleration" [Inc/s<sup>2</sup>]  
Anfahr- und Bremsbeschleunigung für die Referenzfahrt

Homing Method	Direction
33	FALSE
34	TRUE

Komplexe Bewegungsaufgaben - PLCopen-Bausteine > FB 838 - Y\_Homelnit\_SetPosition - Initialisierung Referenzfahrt auf aktuelle Position

### 4.3.24 FB 838 - Y\_Homelnit\_SetPosition - Initialisierung Referenzfahrt auf aktuelle Position

**Beschreibung** Dieser Baustein initialisiert die Referenzfahrt (Homing) auf die aktuelle Position.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	IN	BOOL	<ul style="list-style-type: none"> <li>■ Initialisierung der Referenzfahrt Methode               <ul style="list-style-type: none"> <li>– Flanke 0-1: Werte der Eingangsparameter werden übernommen und die Initialisierung der Referenzfahrt Methode gestartet.</li> </ul> </li> </ul>
Done	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung wurde ohne Fehler beendet.</li> </ul> </li> </ul>
Busy	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Initialisierung ist aktiv</li> </ul> </li> </ul>
Error	OUT	BOOL	<ul style="list-style-type: none"> <li>■ Status               <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen können dem Parameter <i>ErrorID</i> entnommen werden.</li> </ul> </li> </ul>
ErrorID	OUT	WORD	Zusätzliche Fehlerinformationen ↗ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Axis	IN_OUT	STRUCT	Referenz zur Achse.

#### Initialisierung Referenzfahrt auf Endschalter

Mit einer Flanke 0-1 an *Execute* werden die Werte der Eingangsparameter übernommen und die Initialisierung der Referenzfahrt Methode gestartet. So lange die Initialisierung aktiv ist, wird der Ausgang *Busy* auf TRUE gesetzt. Ist die Initialisierung ohne Fehler beendet worden, wird der Ausgang *Done* auf TRUE gesetzt. Tritt bei der Initialisierung ein Fehler auf, wird der Ausgang *Error* auf TRUE gesetzt und am Ausgang *ErrorID* eine Fehlernummer ausgegeben.

#### Initialisierung der Referenzfahrt Methode

1. ➤ Überprüfen der Kommunikation zur Achse.
2. ➤ Prüfen auf erlaubte PLCopen Zustände.
3. ➤ Übertragung der Antriebsparameter:
  - "Homing Method" = 35



### 4.3.25 FB 839 - MC\_TorqueControl - Achse mit konstantem Drehmoment verfahren

**Beschreibung** Mit MC\_TorqueControl können Sie ein maximales Drehmoment vorgeben.

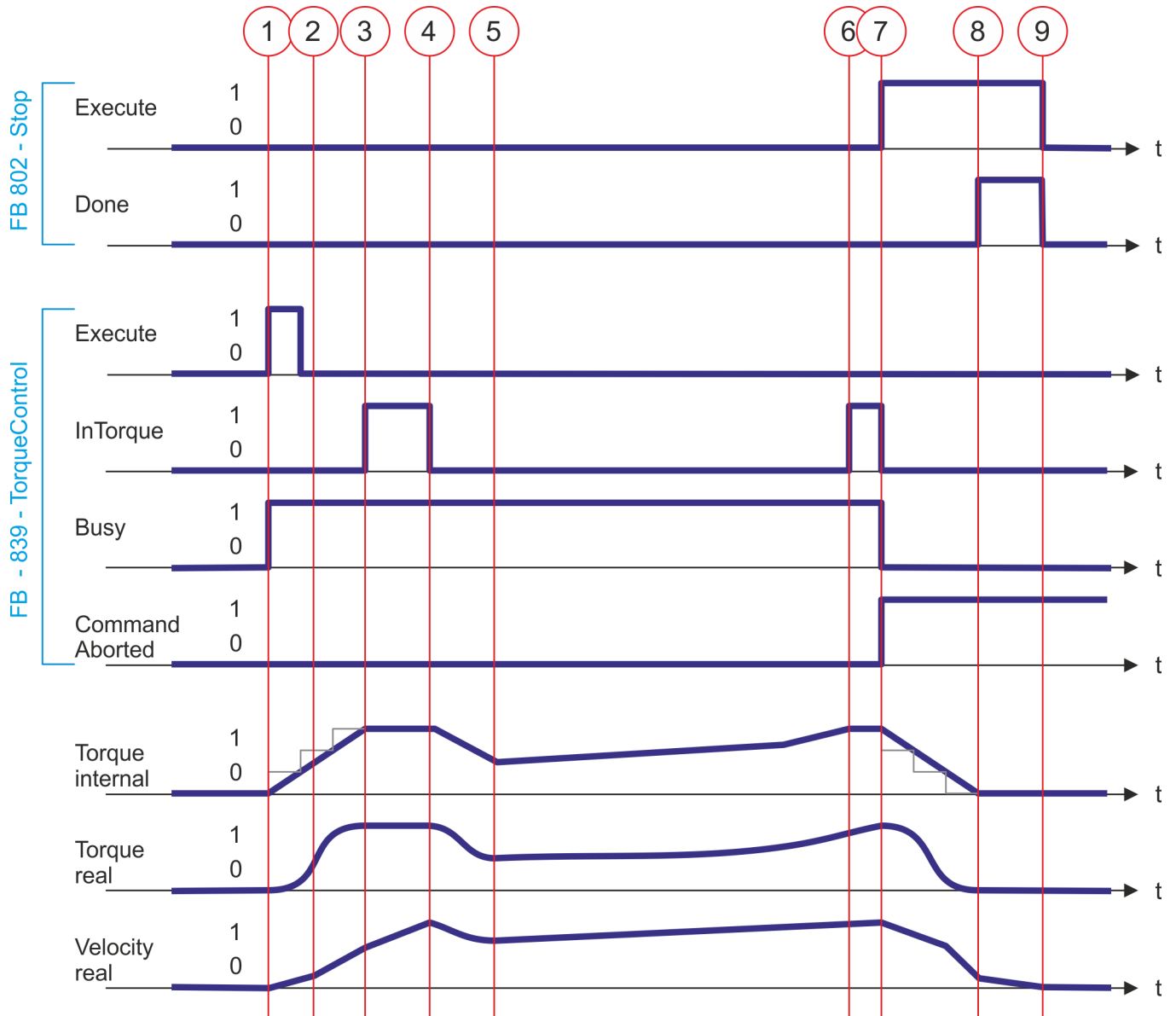
#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Execute	INPUT	BOOL	<ul style="list-style-type: none"> <li>■ Mit Flanke 0-1 wird der Bewegungsauftrag ausgeführt.</li> </ul>
Torque	INPUT	REAL	Wert von Drehmoment/Kraft in [1.0% Rated Motor Torque] auf Motorseite, unabhängig von der konfigurierten Mechanik.
TorqueRamp	INPUT	REAL	Drehmomentrampe pro Sekunde[(1.0% Rated Motor Torque)/s].
Velocity	INPUT	REAL	Wert der maximalen Geschwindigkeit auf Motorseite, unabhängig von der konfigurierten Mechanik. Die Vorgabe hat mit folgenden Einheiten zu erfolgen: <ul style="list-style-type: none"> <li>■ Rotatorischer Motor: [rpm]</li> <li>■ Linearer Motor: [mm/s]</li> </ul>
InTorque	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Torque <ul style="list-style-type: none"> <li>– TRUE: Vorgegebenes Drehmoment erreicht.</li> </ul> </li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status <ul style="list-style-type: none"> <li>– TRUE: Auftrag läuft - Drehmoment wurde noch nicht erreicht.</li> </ul> </li> </ul>
Active	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status <ul style="list-style-type: none"> <li>– TRUE: Baustein steuert die Achse.</li> </ul> </li> </ul>
CommandAborted	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status <ul style="list-style-type: none"> <li>– TRUE: Der Auftrag wurde während der Verarbeitung durch einen anderen Auftrag abgebrochen.</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>■ Status <ul style="list-style-type: none"> <li>– TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen finden Sie im Parameter <i>ErrorID</i>.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	Zusätzliche Fehlerinformationen ↪ <i>Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</i>
Axis	IN_OUT	STRUCT	Referenz zur Achse.



*Bitte beachten Sie, dass dieser Baustein bei der Prüfung und Übertragung der Parameter azyklische Ressourcen der CPU verwendet. Aufgrund dessen startet die Bewegung erst, wenn die Parameter übertragen sind.*

### Status-Diagramm und Stopp-Verhalten



- (1) Mit Flanke 0-1 an *Enable* von FB 839 wird das Verfahren der Achse gestartet und *Busy* liefert TRUE.
- (2) Mit schrittweise zunehmendem Drehmoment erhöht sich die Geschwindigkeit bis einer der Maximalwerte *Torque* bzw. *Velocity* erreicht wird.
- (3) Die Achse erreicht das vorgegebene Drehmoment und *InTorque* liefert TRUE. Die Achse wird weiter beschleunigt bis *Velocity* erreicht wird.
- (4) Die Achse erreicht die vorgegebene Geschwindigkeit *Velocity* und das Drehmoment wird verringert. Hierbei liefert *InTorque* FALSE.
- (5) Aufgrund des Motorwiderstands verringert sich die Geschwindigkeit. Durch schrittweise Erhöhung des Drehmoments wird diesem entgegengewirkt.
- (6) Die Achse erreicht das vorgegebene Drehmoment und *InTorque* liefert TRUE.
- (7) Mit Flanke 0-1 an *Execute* von FB 802 wird das Stoppen der Achse gestartet. Die Achse wird bis zum Stillstand abgebremst. *CommandAborted* von FB 839 liefert TRUE, *InTorque* und *Busy* FALSE.
- (8) Das Stoppen der Achse ist abgeschlossen, die Achse ist gestoppt und *Done* des FB 802 liefert TRUE.
- (9) Mit Flanke 1-0 an *Execute* von FB 802 wird der Stopp-Auftrag beendet und *Done* des FB 802 liefert FALSE.

### 4.3.26 FB 840 - MC\_ReadActualTorque - Aktuelles Drehmoment lesen

#### Beschreibung

Solange *Enable* TRUE ist, gibt dieser Baustein den Wert des Drehmoments zurück. Sobald Ausgabedaten vorhanden sind, wird *Valid* TRUE. Wird *Enable* zurückgesetzt, wird auch *Valid* zurückgesetzt, auch wenn neue Ausgabe-Daten zur Verfügung stehen.

#### Parameter

Parameter	Deklaration	Datentyp	Beschreibung
Enable	INPUT	BOOL	<ul style="list-style-type: none"> <li>Im aktivierten Zustand wird der Parameterwert kontinuierlich geliefert.</li> </ul>
Valid	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>Ein gültiger Satz von Ausgabewerten ist im FB verfügbar.</li> </ul>
Busy	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>Status <ul style="list-style-type: none"> <li>TRUE: Auftrag läuft.</li> </ul> </li> </ul>
Error	OUTPUT	BOOL	<ul style="list-style-type: none"> <li>Status <ul style="list-style-type: none"> <li>TRUE: Ein Fehler ist aufgetreten. Zusätzliche Fehlerinformationen finden Sie im Parameter <i>ErrorID</i>.</li> </ul> </li> </ul>
ErrorID	OUTPUT	WORD	<p>Zusätzliche Fehlerinformationen.</p> <p>↳ Kap. 6 "ErrorID - Zusätzliche Fehlerinformationen" Seite 96</p>
Torque	OUTPUT	REAL	Drehmoment/Kraft der Achse in [1,0% Rated Motor Torque] auf der Motorseite, unabhängig von der konfigurierten Mechanik.
Axis	IN_OUT	STRUCT	Referenz zur Achse.

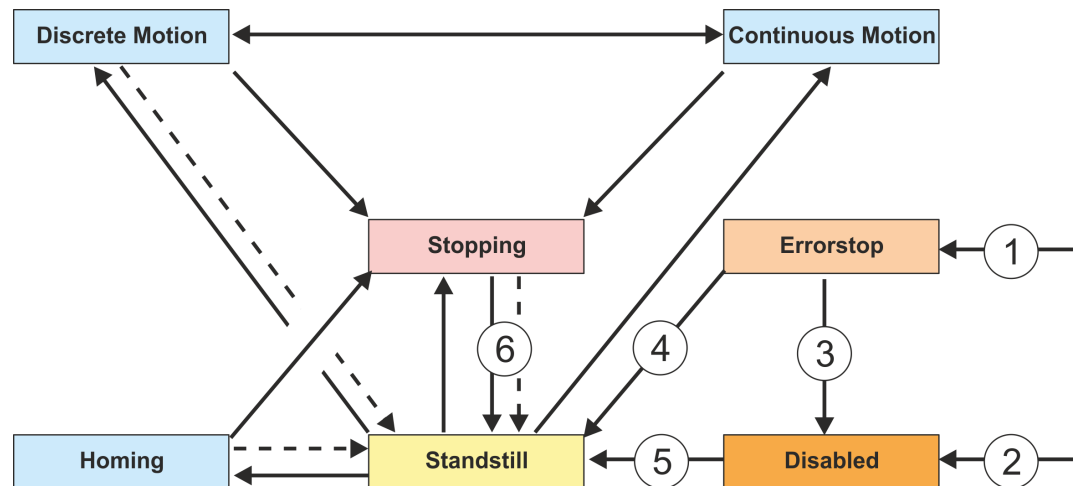
## 5 Zustände und Verhalten der Ausgänge

### 5.1 PLCopen-States

#### Zustandsdiagramm

Im *Zustandsdiagramm* sind alle Zustände aufgeführt, die eine Achse annehmen kann. Eine Achse befindet sich immer in einem dieser Zustände. Je nach Ausgangszustand kann ein Zustandswechsel automatisch oder über die Bausteine der Achskontrolle erfolgen. Grundsätzlich werden Bewegungsaufgaben sequenziell abgearbeitet. Mit folgendem Funktionsbaustein können Sie den Status abfragen:

- ↪ Kap. 4.3.11 "FB 812 - MC\_ReadStatus - Status Achse lesen" Seite 63



-- ➔ Rücksprung wenn fertig

- (1) Aus jedem Status: An der Achse ist ein Fehler aufgetreten
- (2) Aus jedem Status: MC\_Power.Enable = FALSE und es gibt keinen Fehler an der Achse
- (3) MC\_Reset und MC\_Power.Status = FALSE
- (4) MC\_Reset und MC\_Power.Status = TRUE und MC\_Power.Enable = TRUE
- (5) MC\_Power.Enable = TRUE und MC\_Power.Status = TRUE
- (6) MC\_Stop.Done = TRUE und MC\_Stop.Execute = FALSE

Es gibt folgende Zustände

- Disabled
  - Grundzustand einer Achse.
  - Achse kann durch keinen Funktionsbaustein bewegt werden.
- Errorstop
  - Es ist ein Fehler an der Achse aufgetreten.
  - Achse wird gestoppt und ist für weitere Bewegungsaufgaben gesperrt.
  - Achse bleibt in diesem Zustand bis der Fehler behoben ist und ein RESET ausgelöst wird.
  - Fehler an einer Achse werden auch über den entsprechenden Funktionsbaustein zurück gemeldet.
  - Fehler an einem Funktionsbaustein führen nicht in diesen Zustand
- Standstill
  - Bereit für Bewegungsaufgaben
  - Es liegt kein Fehler an der Achse vor
  - Es sind keine Bewegungsaufgaben an der Achse aktiv
  - Achse wird mit Spannung versorgt
- Stopping
  - Achse wird aktuell gestoppt:
    - ↪ Kap. 4.3.5 "FB 802 - MC\_Stop - Achse stoppen" Seite 50
  - Der Zustand *Stopping* ist aktiv solange ein Stop Kommando aktiv ist (*Execute* = 1). Auch wenn die Achse schon gestoppt ist. Danach wechselt der Zustand automatisch nach *Standstill*.

- Homing
  - Die Achse führt aktuell eine Referenzfahrt durch:
    - ↳ Kap. 4.3.4 "FB 801 - MC\_Home - Achse referenzieren" Seite 48
  - Sobald die Achse referenziert ist, wechselt der Zustand automatisch nach *Standstill*.
- Discrete Motion
  - Die Achse führt aktuell eine Bewegungsaufgabe durch:
    - ↳ Kap. 4.3.9 "FB 808 - MC\_MoveAbsolute - Achse auf absolute Position verfahren" Seite 59
    - ↳ Kap. 4.3.7 "FB 804 - MC\_MoveRelative - Achse relativ verfahren" Seite 55
    - ↳ Kap. 4.3.6 "FB 803 - MC\_Halt - Achse anhalten" Seite 53
  - Sobald das Ziel der Bewegungsaufgabe erreicht ist, wechselt der Zustand automatisch nach *Standstill*.
- Continuous Motion
  - Die Achse führt eine dauerhafte Bewegungsaufgabe durch:
    - ↳ Kap. 4.3.8 "FB 805 - MC\_MoveVelocity - Achse verfahren mit konstanter Geschwindigkeit" Seite 57

## 5.2 Verhalten der Ein- und Ausgänge

### Ausschließlichkeit der Ausgänge

- Die Ausgänge *Busy*, *Done*, *Error* und *CommandAborted* schließen sich gegenseitig aus, es kann also an einem Funktionsbaustein nur einer dieser Ausgänge zu einer Zeit TRUE sein.
- Sobald der Eingang *Execute* TRUE wird, muss einer der Ausgänge TRUE werden. Ebenfalls kann nur einer der Ausgänge *Active*, *Error*, *Done* und *CommandAborted* zu einer Zeit TRUE sein.

### Ausgangs-Zustand

- Die Ausgänge *Done*, *InVelocity*, *Error*, *ErrorID* und *CommandAborted* werden mit einer Flanke 1-0 am Eingang *Execute* zurückgesetzt, wenn der Funktionsbaustein nicht aktiv ist (*Busy* = FALSE).
- Die Kommandoausführung wird durch eine Flanke 1-0 an *Execute* nicht beeinflusst.
- Falls *Execute* bereits während der Kommandoausführung zurückgesetzt wird, so ist sichergestellt, dass einer der Ausgänge am Ende des Kommandos für einen SPS-Zyklus gesetzt wird. Erst danach werden die Ausgänge zurückgesetzt.

### Eingangs-Parameter

- Die Eingangs-Parameter werden mit Flanke 0-1 an *Execute* übernommen.
- Zur Änderung der Parameter muss das Kommando neu getriggert werden.
- Falls ein Eingangs-Parameter nicht an den Funktionsbaustein übergeben wird, so bleibt der zuletzt an diesen Baustein übergebene Wert gültig.
- Beim ersten Aufruf muss ein sinnvoller Default-Wert übergeben werden.

### Position und Distanz

- Der Eingang *Position* bezeichnet einen absoluten Positionswert.
- *Distance* bezeichnet ein relatives Maß als Abstand zweier Positionen.
- Sowohl *Position*, als auch *Distance* werden in technischen Einheiten, z.B. [mm] oder [°], entsprechend der Skalierung der Achse angegeben.

### Parameter für das dynamische Verhalten

- Die Dynamikparameter für *Move*-Funktionen werden in technischen Einheiten mit der Zeitbasis Sekunde angegeben.  
Ist eine Achse beispielsweise in Millimetern skaliert, so sind die Einheiten für *Velocity* [mm/s], *Acceleration* [mm/s<sup>2</sup>], und *Deceleration* [mm/s<sup>2</sup>].

### Fehlerbehandlung

- Alle Funktionsbausteine haben zwei Fehlerausgänge um Fehler während der Kommandoausführung anzuzeigen.
- *Error* zeigt den Fehler an und *ErrorID* gibt eine ergänzende Fehlernummer aus.
- Die Ausgänge *Done* und *InVelocity*, bezeichnen eine erfolgreiche Kommandoausführung und werden nicht gesetzt, wenn *Error* TRUE wird.

**Fehlertypen**

- Funktionsbausteinfehler
  - Funktionsbausteinfehler sind Fehler, die ausschließlich den Funktionsbaustein und nicht die Achse betreffen wie z.B. fehlerhafte Parametrierung.
  - Funktionsbausteinfehler müssen nicht explizit zurückgesetzt werden, sondern werden selbständig zurückgesetzt, wenn der Eingang *Execute* zurückgesetzt wird.
- Kommunikationsfehler
  - Kommunikationsfehler wie z.B. der Funktionsbaustein kann die Achse nicht adressieren.
  - Kommunikationsfehler deuten oft auf eine fehlerhafte Konfiguration oder Parametrierung hin.
  - Ein Reset ist nicht möglich, sondern der Funktionsbaustein kann neu getriggert werden, nachdem die Konfiguration korrigiert wurde.
- Achsfehler
  - Achsfehler treten üblicherweise während der Fahrt auf wie z.B. Schleppabstandsfehler.
  - Ein Achsfehler muss durch *MC\_Reset* zurückgesetzt werden.

**Verhalten des *Done*-Ausgangs**

- Der *Done*-Ausgang wird gesetzt, wenn ein Kommando erfolgreich ausgeführt wurde.
- Wenn mit mehreren Funktionsbausteinen an einer Achse gearbeitet wird und das laufende Kommando durch einen weiteren Baustein unterbrochen wird, so wird der *Done*-Ausgang des ersten Bausteins nicht gesetzt.

**Verhalten des *CommandAborted*-Ausgangs**

- *CommandAborted* wird gesetzt, wenn ein Kommando durch einen anderen Baustein unterbrochen wird.

**Verhalten des *Busy*-Ausgangs**

- Der *Busy*-Ausgang zeigt an, dass der Funktionsbaustein aktiv ist.
- *Busy* wird sofort mit Flanke 0-1 an *Execute* gesetzt und wird erst zurückgesetzt, wenn das Kommando erfolgreich oder auch nicht erfolgreich beendet wurde.
- Solange *Busy* TRUE ist, muss der Funktionsbaustein zyklisch aufgerufen werden, um das Kommando ausführen zu können.

**Verhalten des *Active*-Ausgangs**

- Wenn die Bewegung einer Achse durch mehrere Funktionsbausteine gesteuert wird, so zeigt der *Active*-Ausgang jedes Bausteins an, dass das Kommando durch die Achse ausgeführt wird.

***Enable*-Eingang und *Valid*-Ausgang**

- Im Gegensatz zu *Execute* führt der *Enable*-Eingang dazu, dass eine Aktion permanent und wiederholt ausgeführt wird, solange *Enable* TRUE ist. *MC\_ReadStatus* aktualisiert beispielsweise zyklisch den Zustand einer Achse solange *Enable* TRUE ist.
- Ein Funktionsbaustein mit einem *Enable*-Eingang zeigt durch den *Valid*-Ausgang an, dass die an den Ausgängen angezeigten Daten gültig sind. Die Daten können jedoch ständig aktualisiert werden während *Valid* TRUE ist.

## 6 ErrorID - Zusätzliche Fehlerinformationen

ErrorID	Beschreibung	Bemerkung
0x0000	Kein Fehler	
0x8001	Unzulässiger Wert beim Parameter <i>Position</i> (außerhalb des Bereichs).	
0x8002	Unzulässiger Wert beim Parameter <i>Distance</i> (außerhalb des Bereichs).	
0x8003	Unzulässiger Wert beim Parameter <i>Velocity</i> .	
0x8004	Unzulässiger Wert beim Parameter <i>Acceleration</i> .	
0x8005	Unzulässiger Wert beim Parameter <i>Deceleration</i> .	
0x8011	Unzulässiger Wert beim Parameter <i>Source</i> .	
0x8012	Unzulässiger Wert beim Parameter <i>Direction</i> .	
0x801D	Parameterkommunikation mit allgemeinem Fehler. Die Fehlerursache ist nicht näher beschrieben.	
0x802F	Keine Systemressourcen verfügbar.	
0x8050	Rack: Falsche Geräte ID / Hersteller ID.	MC_Power, Y_Init Y_ServoFunction
0x8051	Nicht unterstützte Firmwareversion.	Y_Init
0x8052	Der konfigurierte Steckplatz ist kein PN Device.	MC_Power
0x8053	Konfiguriertes Rack nicht auf dem Steckplatz.	MC_Power
0x8054	Geräteadresse (Diagnoseadresse) ist "0".	MC_Power
0x8057	Falscher Gerätezustand / Modul nicht betriebsbereit. Rack nicht verfügbar und / oder Modul nicht verfügbar.	MC_Power Y_Init Y_ServoFunction
0x8058	Geräteadresse des Moduls ist nicht erkennbar.	
0x8059	Geräteverbindung verloren.	Y_Init Y_ServoFunction
0x805A	Gerät nicht betriebsbereit, Gerät nicht verfügbar. Der erwartete Typ ist nicht gleich dem tatsächlichen Typ.	MC_Power
0x805B	Modul: Falsche Geräte ID / Hersteller ID.	MC_Power
0x805C	Falsche Telegrammkonfiguration im Antrieb. Die erwartete Telegrammnummer "999" ist nicht vorhanden.	MC_Power Y_Init
0x805D	Falsche Telegrammkonfiguration in der SPS. Die konfigurierte Telegrammnummer ist nicht "999".	MC_Power Y_Init
0x805E	Modul ist nicht betriebsbereit.	MC_Power
0x805F	Treiber ist nicht bereit.	Y_Init, Y_ServoFunction Y_ReadParameter, Y_WriteParameter
0x8060	Lesen der SERVOPACK Informationen fehlgeschlagen.	Y_Init



ErrorID	Beschreibung	Bemerkung
0x8061	Ungültige Konfiguration: Kombination der Eingänge <i>Resolution</i> , <i>GearboxFactor</i> und <i>Traversing</i> wird nicht unterstützt. Interner <i>pos.unit</i> Faktor liegt außerhalb des Bereichs.	Y_Init
0x8062	Ungültige Konfiguration: Interner Wert für <i>Resolution</i> , <i>Gearbox-Factor</i> und <i>Traversing</i> ist "0".	Y_Init
0x8063	Ungültige Konfiguration: Motorauflösung zu hoch.	Y_Init
0x8064	Unbekannter / nicht unterstützter Encoder-/Motortyp.	Y_Init
0x8065	Einstellung der Anwendereinheit fehlgeschlagen.	Y_Init
0x8066	Ungültige Positionsgrenzen konfiguriert.	Y_Init
0x8067	Schreiben der Positionsgrenzen fehlgeschlagen.	Y_Init
0x8068	Lesen der Motorinformationen fehlgeschlagen.	Y_Init
0x8069	Einstellung der Kommunikationsparameter fehlgeschlagen.	Y_Init
0x806A	Einstellung der Achse fehlgeschlagen: Geben Sie Informationen für eine Endlos-Achse an.	Y_Init
0x806B	Sichern der PROFINET Parameter fehlgeschlagen.	Y_Init
0x806C	Software-Reset zur Aktivierung der Kommunikation und Anwendungseinstellungen fehlgeschlagen.	Y_Init
0x806D	Achseinstellung fehlgeschlagen: Alarm Multiturn-Limit ist nicht verfügbar.	Y_Init
0x806E	Achseinstellung fehlgeschlagen: Multiturn-Limit Einstellungen.	Y_Init
0x806F	Achseinstellung fehlgeschlagen: Basisblock Status ohne Alarm erwartet.	Y_Init
0x8070	Software-Reset zur Aktivierung der Multiturn-Limit Einstellungen ist fehlgeschlagen.	Y_Init
0x8090	Angegebene logische Adresse ungültig.	MC_Power
0x8094	Es wurde kein Subnetz mit der angegebenen SUBNET-ID konfiguriert.	MC_Power
0x8095	Unzulässiger Wert für den Parameter STATION der Hardware-Konfiguration.	MC_Power
0x8096	Unzulässiger Wert für den Parameter SLOT der Hardware-Konfiguration.	MC_Power
0x8097	Unzulässiger Wert für den Parameter SUBSLOT der Hardware-Konfiguration.	MC_Power
0x8099	Der SLOT ist nicht konfiguriert.	MC_Power
0x809A	Die Adresse des Interface-Moduls ist nicht für den ausgewählten SLOT konfiguriert.	MC_Power
0x8101	Es ist keine Kommunikation mit der Achse möglich.	
0x8102	Befehl ist im aktuellen PLCopen-State nicht zulässig.	
0x8103	Befehl wird von der Achse nicht unterstützt.	

ErrorID	Beschreibung	Bemerkung
0x8104	Achse ist nicht einschaltbereit, mögliche Gründe: <ul style="list-style-type: none"> <li>■ Kommunikation zur Achse nicht bereit.</li> <li>■ Antrieb ist nicht im Zustand "S2: Ready For Switching On" → Antriebsfehler evtl. mit MC_Reset zurücksetzen.</li> <li>■ Kommunikation wurde unterbrochen, z.B. durch Aus- / Einschalten der CPU. Fehler mit MC_Reset zurücksetzen.</li> </ul>	
0x8201	Wegen Mangels an internen Ressourcen kann der Befehl aktuell nicht ausgeführt werden (kein freier Slot im Befehlspeicher / Parameterpuffer).	
0x8203	Die Referenzfahrt hat keine gültigen Daten, da die Referenzfahrt nicht konfiguriert wurde.	MC_Home
0x8204	Die Konfiguration der Referenzfahrt ist fehlgeschlagen (Home Method, Home Offset, Home Speed Switch, Home Speed Zero, Home Acceleration).	MC_Home
0x820A	Schreiben <i>TorqueRamp</i> fehlgeschlagen.	MC_TorqueControl
0x820B	Schreiben <i>Velocity</i> fehlgeschlagen.	MC_TorqueControl
0x820C	Parameter Puffer: Interner Fehler	
0x820D	Parameter Puffer: Zeitüberschreitung	
0x820E	Interner Fehler: Programmierfehler	
0x820F	Timeout: <i>DeviceCom</i> Timer für Leseanforderung / Schreibanforderung ist abgelaufen (900ms).	
0x8211	Timeout: <i>DeviceCom</i> Timer, der die max. Bearbeitungszeit begrenzt, ist abgelaufen (25s).	
0x8212	Timeout: <i>DeviceDriver</i> Timer für azyklische Anforderung abgelaufen (5s).	
0x8306	Kommunikationsfehler an der Master-Achse. Slave-Achse wird mit Schnellhalt gestoppt.	
0x8316	FB, der in mehr als einem zyklischen OB aufgerufen wird, ist nicht zulässig.	
0x8317	Aufgerufener FB wird vom OB nicht unterstützt.	
0x8340	Ungültiger Wert in <i>TriggerInput.Probe</i> .	MC_TouchProbe, MC_AbortTrigger
0x8341	Ungültiger Wert in <i>TriggerInput.Source</i> .	MC_TouchProbe, MC_AbortTrigger
0x8342	Ungültiger Wert in <i>TriggerInput.TriggerMode</i> .	MC_TouchProbe, MC_AbortTrigger
0x8350	Ungültiger Wert in <i>VelocitySearchSwitch</i> .	Y_HomeInit...
0x8351	Ungültiger Wert in <i>VelocitySearchZero</i> .	Y_HomeInit...
0x8352	Ungültige Kombination von Eingängen.	Y_HomeInit...
0x8363	Stoppbefehl ist aktiv. Während ein Stoppbefehl aktiv ist, ist kein neuer Befehl zulässig ( <i>Execute</i> = TRUE).	
0x8400	Unerwarteter Drive-State, Drive-State $\neq$ <i>Operation enabled</i>	MC_Power
0x8401	Unerwarteter Drive-State, Drive-State = <i>Quick stop active</i>	MC_Power

ErrorID	Beschreibung	Bemerkung
0x8402	Unerwarteter Drive-State, Drive-State = <i>Fault reaction active</i>	MC_Power
0x8403	Unerwarteter Drive-State, Drive-State = <i>Fault</i>	MC_Power
0x8410	Zeitüberschreitung beim Versuch den Antrieb zurückzusetzen.	MC_Reset
0x851B	Ungültiger Wert am Parameter <i>Torque</i> .	MC_TorqueControl
0x851C	Ungültiger Wert am Parameter <i>TorqueRamp</i> .	MC_TorqueControl
0x851D	Ungültiger Wert am Parameter <i>CmdType</i> bzw. <i>Mode</i> .	Y_ServoFunction
0x8A00	Unzulässige Parameternummer: Zugriff auf nicht verfügbaren Parameter.	Y_ReadParameter, Y_WriteParameter
0x8A01	Parameterwert kann nicht geändert werden: Zugriff auf einen Parameterwert, der nicht geändert werden kann.	Y_WriteParameter
0x8A02	Unterer oder oberer Grenzwert überschritten: Ändern Sie den Zugriff mit einem Wert außerhalb der Grenzwerte.	Y_WriteParameter
0x8A03	Ungültiger Subindex: Zugriff auf nicht verfügbaren Subindex.	Y_ReadParameter, Y_WriteParameter
0x8A04	Kein Array: Zugriff mit Subindex auf nicht indizierten Parameter.	Y_ReadParameter, Y_WriteParameter
0x8A05	Falscher Datentyp: Zugriff mit einem Wert, welcher nicht mit dem Datentyp des Parameters übereinstimmt.	Y_ReadParameter, Y_WriteParameter
0x8A06	Einstellung nicht erlaubt (darf nur zurückgesetzt werden): Zugriff ändern mit Wert $\neq$ "0", wenn dies nicht zulässig ist.	Y_WriteParameter
0x8A0B	Keine Bedienpriorität: Zugriff ändern ohne Rechte zum Ändern von Parametern.	Y_ReadParameter, Y_WriteParameter
0x8A11	Anforderung kann wegen des Betriebsmodus nicht ausgeführt werden: Informationen finden Sie im Benutzerhandbuch Sigma-7 SERVOPACK.	Y_ReadParameter, Y_WriteParameter
0x8A14	Wert unzulässig: Informationen finden Sie im Benutzerhandbuch Sigma-7 SERVOPACK.	Y_WriteParameter
0x8A17	Unzulässiges Format: Schreibauftrag: Unzulässiges Format oder Format der Parameterdaten, das nicht unterstützt wird.	Y_ReadParameter, Y_WriteParameter

## Anhang

## Inhalt

<b>A</b>	<b>Änderungshistorie.....</b>	<b>102</b>
----------	-------------------------------	------------

## A Änderungshistorie

Rev.	Änderungen
19-16	Das Handbuch wurde neu erstellt.
19-40	Bibliothek einbinden <ul style="list-style-type: none"><li>■ Kapitel "Einbinden in Siemens TIA Portal" wurde neu hinzugefügt.</li></ul> Einsatz Sigma-7 PROFINET <ul style="list-style-type: none"><li>■ Kapitel "Einsatz im Siemens TIA Portal" wurde neu hinzugefügt.</li></ul>