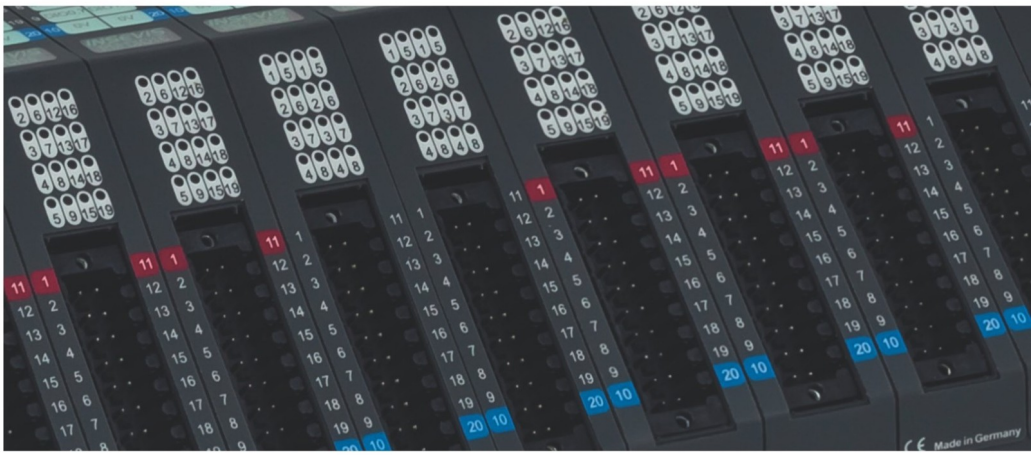
INSEVIS



Beispieldokumentation
Sample documentation

Überschrift / Thema deutsch

Überschrift / Thema englisch

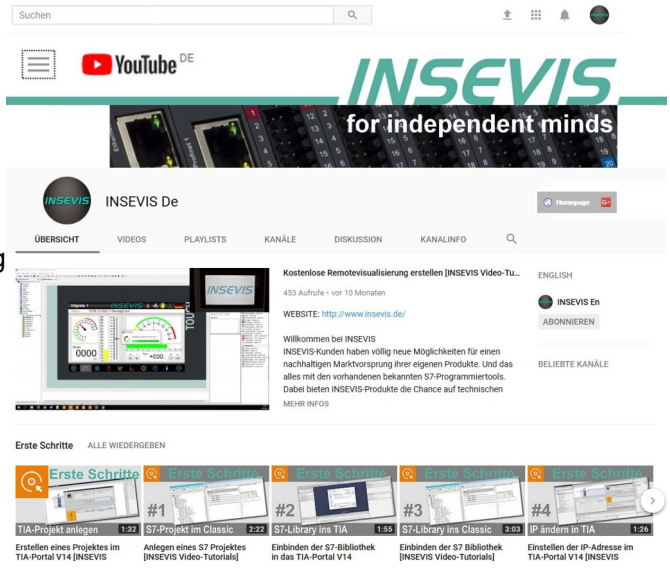
Hinweis zum besseren Verständnis durch Zusatzinformationen

Im deutschen INSEVIS-YouTube-Kanal INSEVIS DE stehen mehrere Playlists mit **Hantierungsvideos** für einzelne Details zur Verfügung.

Ebenfalls stehen **Handbücher** für die einzelnen Produktgruppen im Downloadbereich der Webseite insevis.de zur Verfügung

Bitte nutzen Sie diese Informationsquellen in Ergänzung zur vorliegenden Dokumentation. So können Sie sich noch leichter mit den INSEVIS-Funktionen vertraut machen.

Möchten Sie Erweiterungswünsche oder Fehler zu diesen Beispielen melden oder wollen Sie anderen eigene Bespielprogramme kostenlos zur Verfügung stellen? Gern werden Ihre Programme -auf Wunsch mit Benennung des Autors- allen INSEVIS- Kunden zur Verfügung gestellt.



Hinweis zu den verschiedenen Versionen der Beispielprogramme

Im Lieferumfang der Beispielprogramme können sich auch ältere Ausgabestände bzw. Versionen befinden. Diese wurden nicht aktualisiert und auf die neueste Siemens-Programmiersoftware angepasst, um einen Zugriff mit älteren Programmier-systemen weiterhin zu ermöglichen. Generell werden INSEVIS-Beispielprogramme immer mit dem aktuell neuesten Siemens-Programmiertools erstellt.

BEISPIELBESCHREIBUNG

Inhaltsverzeichnis

Motivation

Seit Jahren werden von Firmen der Antriebstechnikbranche herstellerspezifische S7-Bausteine zur leichteren Einbindung ihrer Antriebstechnik in die Steuerungswelt der Simatic- und kompatiblen SPSen angeboten. Dies geschieht oft mit einem effektiven, den Möglichkeiten des Antriebs angepasstem, monolithischem Funktionsbaustein, der eine optimierte, jedoch willkürliche Schnittstelle aufweist und zudem meist auf ein Bussystem zugeschnitten ist (in der Regel Profibus-DP, aber auch Interbus-S und CANopen mittels Feldbus-Master-Baugruppen anderer Hersteller).

Die PLCopen (http://www.plcopen.org) als internationale Organisation hat sich unter anderem zum Ziel gesetzt, Engineering-Aufwand durch einheitliche Software-Schnittstellen zu reduzieren. Im Antriebsbereich wurden darum Standards mit Einzelfunktionen für Antriebe definiert, eine Zertifizierung von Antrieben und implementierten Schnittstellen ist möglich. Bei Verwendung von Bussystemen wie CANopen mit Antriebsschnittstellen (DS402 Antriebsprofil) ist zudem der Aufwand zur Anpassung an eine konkretes Busprotokoll gering.

Im folgenden wird der Betrieb an einem Servodrive Maxon EPOS2 24/5 (http://www.maxonmotor.de) beschrieben. Die erstellte S7-Software wurde für INSEVIS-SPS'n erstellt und ist an den PLCopen-Standard angelehnt.

An folgenden Geräten erfolgte der Test der Software:

EPOS2

Testgerät	:	EPOS2 24/5
Software-Version	:	0x2122
Hardware-Version	:	0x6220
EPOS-Studio	:	1.44 Revision 1

INSEVIS

Testgerät	:	CC300V
Betriebssystem	:	2.0.23
S7-Bibliothek	:	Insevis_S7-library_from_2_0_22

Es werden aktuell nicht alle verfügbaren Modi unterstützt, wie z.B. der „Master Encoder Mode“, da die HW-Plattformen zu diversitär sind, um mit dem gleichen Encoder-Typ zu arbeiten. Bei Bedarf könnte ein MC_Gear_E2 grundsätzlich nach-implementiert werden.

Die Firma inmotec Automation GmbH (support@inmotec.de) erstellt und erweitert antriebsnahe Software für INSEVIS-Steuerungen.

Grundprinzipien des Software-Entwurfs

1. Alle Antriebsfunktionen (sogenannte Motion-Control-Bausteine MC_) werden als einzelne

Funktionsbausteine implementiert, z.B. ist der Funktionsbaustein „MC_Power_E2“ ein S7-FB, der zum Bestromen des Servomotors dient. Da der Servomotor nicht nur bestromt werden muss, sondern auch Bewegungsfunktionen ausführen soll, sind weitere Funktionsbausteine erforderlich, auch werden selbstverständlich mehrere Achsen unterstützt. Um die Vielzahl von Instanzen von Funktionsbausteinen mit einem separaten Instanzdatenbaustein zu vermeiden, empfiehlt sich die Instanziierung von Funktionsbausteinen im STAT-Bereich der Variablendefinition des „Container“-Funktionsbausteins.

2. Die MC-Bausteine verwenden keine globalen Ressourcen wie M-Merker, T-Zeiten oder Z-Zähler, sondern deren instanziierbaren IEC-Varianten.
3. Alle Antriebsfunktionen auf der INSEVIS-SPS kommunizieren über asynchrone CANopen-PDO's nach DS301, so dass der Kommunikationsaufwand (Busauslastung) reduziert ist. Beim Antriebsprofil DS402 werden ausschließlich Betriebsarten verwendet, die keine äquidistante Übertragung von Sollwerten erfordern. Der sogenannte „Interpolated mode“ wird nicht verwendet.
4. Die Funktionsbausteine werden im Original mit SCL (Structured Control Language), einer Engineering-Option zu Step7 der Firma Siemens erstellt, die Verwendung der Funktionsbausteine erfordert jedoch kein installiertes SCL-Paket auf dem Entwicklungsrechner des Anwenders.
5. Um Diversitäten bei Antrieben abzufangen und Namenskonflikte mit bereits vorhandenen Bausteinen aus Bibliotheken zu vermeiden (z.B. bei Technologie-SPSen der Firma Siemens), erhalten die MC-Bausteine einen Postfix wie „_E2“ in Abhängigkeit vom jeweiligen Antrieb. Es bleibt zu bemerken, dass der Instanz-Name (im Beispiel „Axis00“ bei Tausch von Antrieben unberührt bleibt).
6. Da sich Bausteine nicht gegenseitig referenzieren, können Baustein-Adressen (absolute Nummern) dem Bedarf des Anwenderprogramms angepasst werden.

Antriebsfunktionen (MC-Bausteine und -Typen)

Grundsätzlich wird der Betrieb mit Encoder empfohlen, wenn Positionieraufgaben zu erfüllen sind. Erfolgt die Drehzahlerfassung über Hallsensoren, können keine Positionieraufgaben realisiert werden. Lediglich der Drehzahlbetrieb ist sinnvoll. Es wird empfohlen, dann den Motor mit mindestens 1000 U/min zu betreiben.

MC-Baustein/Symbol	Adresse	Funktionalität
MC_ReadStatus_E2	FB40	Visualisierung der Antriebszustände (entstromt, stoppend, profilbasierte Bewegungsfunktionen aktiv, Endlosdrehen aktiv, Referenzfahrt aktiv)
MC_ReadAxisError_E2	FB41	Visualisierung des Fehlercodes des Antriebs
MC_ReadActualPosition_E2	FB42	Visualisierung der aktuellen Position des Motors
MC_ReadActualVelocity_E2	FB43	Visualisierung der aktuellen Geschwindigkeit des Motors
MC_Reset_E2	FB44	Fehler im Antrieb rücksetzen
MC_Power_E2	FB45	Motor bestromen/entstromen mit Schnellstopprampe
MC_Stop_E2	FB46	Bestromten Motor stoppen mit Schnellstopprampe
MC_MoveAbsolute_E2	FB47	Absolute Position anfahren
MC_MoveRelative_E2	FB48	Relative Distanz abfahren
MC_MoveVelocity_E2	FB50	Endlosbewegung (Drehzahlvorgabe)
MC_Home_E2	FB52	Referenzfahrt ausführen
MC_Jog_E2	FB53	Hand+/- fahren, stoppt an den Software-Endgrenzen
E2_Input	FB54	Eingänge auslesen
InDataE2Type	UDT100	Datentyp für Eingangsdaten CANopen, pro Achse einmal instantiieren
OutDataE2Type	UDT101	Datentyp für Ausgangsdaten CANopen, pro Achse einmal instantiieren
SWPosE2Type	UDT102	Datentyp Statuswort CANopen, NUR IINTERNE

MC-Baustein/Symbol	Adresse	Funktionalität
		VERWENDUNG
CWPosE2Type	UDT103	Datentyp Steuerwort CANopen, NUR IINTERNE VERWENDUNG
AxisRefE2Type	UDT104	Datentyp Achsreferenz, pro Achse einmal instantiieren

Einstellungen für EPOS2

Da der EPOS2 feste Formate für Positionen (Post-Quad-Inkremente), Geschwindigkeiten (U/min) und Rampen (U/min/s) verwendet, ist eine Umsetzung, die die Vorgabe eher nachvollziehbaren Ingenieurwerten ermöglicht, sinnvoll.

Der Anwender hat daher folgende Parameter in die Achsreferenz zu schreiben, damit direkt in Nutzereinheiten, Nutzereinheiten/s und Nutzereinheiten/s^2 programmiert werden kann:

```
      L     1.000000e+001                    // z.B. 10 mm
Vorschub/Umdrehung
      T     "AxisRef".Data.Axis00.fMechanicPitch
       L     4.000000e+003                        // Inkremente Enc. nach
                                                 // Vervierfachung, z.B. 4000
bei
                                                 // einem 1000-Impuls-Encoder
      T     "AxisRef".Data.Axis00.fEncPQPerMotRev
```

Zusätzlich ist die CANopen-Knotennummer für z.B. azyklische Abfragen (Fehlernummer) vorzugeben:

```
          L     4
      T     "AxisRef".Data.Axis00.iNode
```

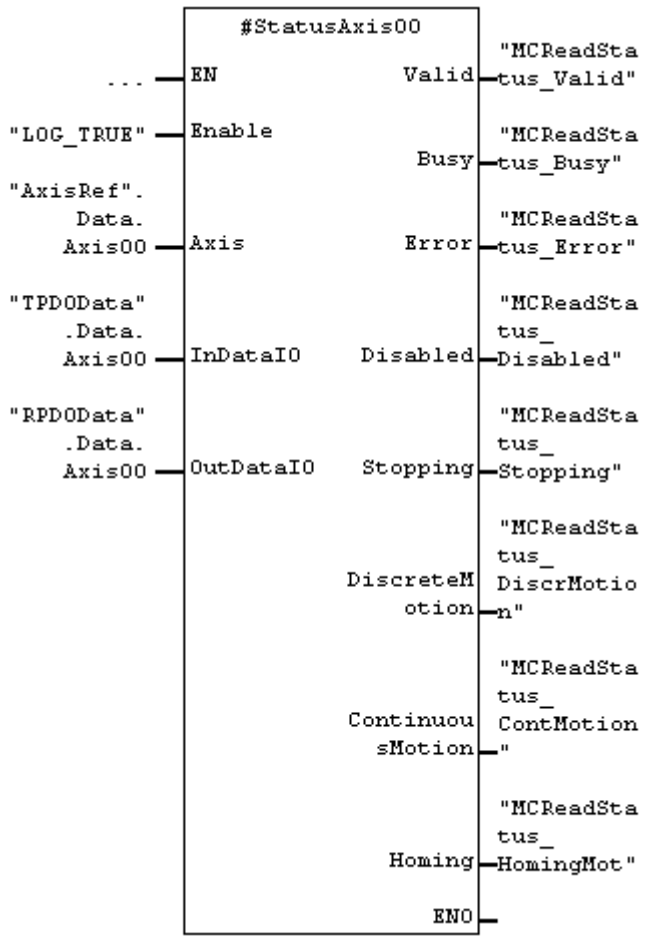
MC_ReadStatus_E2 (FB)

Der MC_ReadStatus_E2 wird zur Visualisierung (Statusbildung) verschiedener Antriebszustände verwendet. Anhand dieser Informationen kann das SPS-Programm Aktivitäten des Antriebs verfolgen.

ⓘ	Dieser Baustein ist zwingend in das SPS-Programm einzubinden, da neben der Statusbildung auch die komplette Achsreferenz bearbeitet wird. Daher ist die Bausteingröße auch deutlich größer als bei den andern MC-Bausteinen. Erläuterungen findet man im Abschnitt zur Achsreferenz. Es ist nur eine Instanz dieses FB's sinnvoll und erlaubt.

Name	Variablenbereich	Typ	Funktion
Enable	IN	Bool	Statusbildung aktivieren Für die Abarbeitung der Achsreferenz ist der Enable-Eingang unbedeutend, der FB muss einmal pro Zyklus aufgerufen werden.
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)

Name	Variablenbereich	Typ	Funktion
InDataIO	IN_OUT	InDataE2Type (UDT)	Referenz auf IO-Daten (Eingangsdaten CANopen)
OutDataIO	IN_OUT	OutDataE2Type (UDT)	Referenz auf IO-Daten (Ausgangsdaten CANopen)
Valid	OUT	Bool	FB-Daten sind gültig (sobald Enable = True)
Busy	OUT	Bool	FB-Funktion läuft (immer False, da aus PDO-Daten erzeugt)
Error	OUT	Bool	Achse mit Fehler
Disabled	OUT	Bool	Achse stromlos
Stopping	OUT	Bool	Achse stoppt (gerade)
DiscreteMotion	OUT	Bool	Achse positioniert
ContinuousMotion	OUT	Bool	Achse positioniert endlos
Homing	OUT	Bool	Achse führt Homing-Fahrt aus



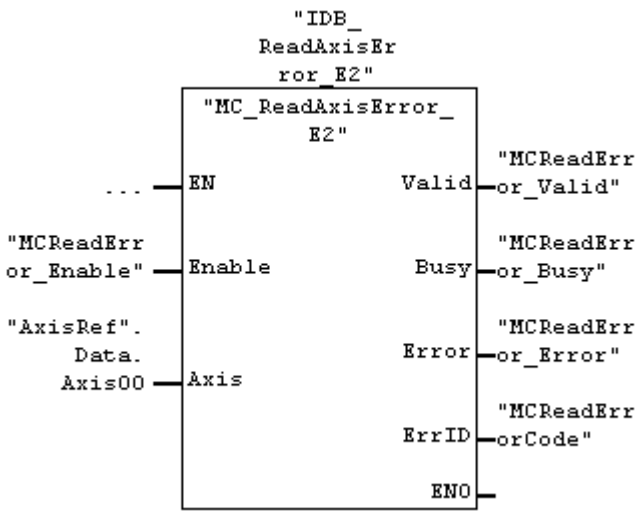
Im Stat-Bereich eines Container-FB's instanziierter MC_ReadStatus_E2 mit dem Instanznamen StatusAxis00.

MC_ReadAxisError_E2 (FB)

Der MC_ReadAxisError_E2 wird zur Visualisierung des Fehlercodes des Achse verwendet. **Die Bedeutung des Fehlercodes ist der Hilfeanleitung des EPOS2 zu entnehmen**.

Name	Variablenbereich	Typ	Funktion
Enable	IN	Bool	Fehlercode lesen
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Valid	OUT	Bool	FB-Daten sind gültig (sobald Enable = True)
Busy	OUT	Bool	FB-Funktion (SDO-Transfer) läuft (gerade)
Error	OUT	Bool	Achse mit Fehler
ErrorID	OUT	DWORD	Fehlercode Achse (hier 32-Bit)

ⓘ	Gelesen wird das EPOS2-CANopen-Objekt 0x1003 auf Subindex 1, welches den aktuellen Fehlercode des Gerätes enthält. Das Objekt wird mittels CANopen-SDO gelesen, sobald das Fehlerbit im Status-Wort auf TRUE gesetzt wird (Flanke) oder der Enable-Eingang auf TRUE (Flanke) gesetzt wird. Erfolgt nach einer TIME-OUT-Zeit von 150ms keine Antwort vom Gerät oder wird beim SDO-Transfer ein Fehler gemeldet, wird ein Fehlercode DW#16#FFFFFFFF (SDO-Transfer-Fehler) angezeigt. **Der FB kann in der aktuellen Version nicht in dem STAT-Bereich eines Instanz-DB's eines übergeordneten FB's instanziiert werden! Es muss eine separate Instanz für den MC_ReadAxisError_E2 angelegt werden.** Es ist nur eine Instanz dieses FB's sinnvoll und erlaubt.



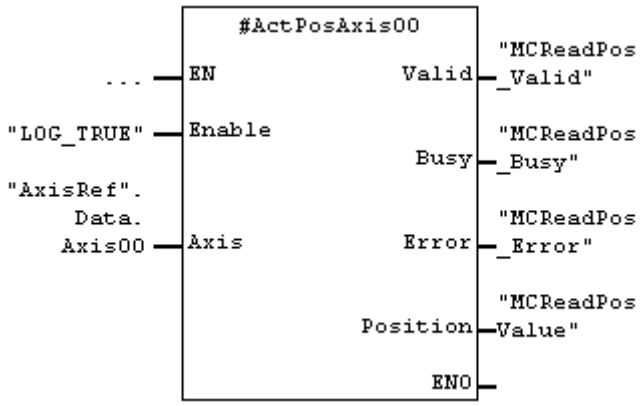
Separat instanziierter MC_ReadAxisError_E2 mit dem Instanznamen ErrorAxis00.

MC_ReadActualPosition_E2 (FB)

Der MC_ReadActualPosition_E2 stellt die absolute Istposition der Achse bereit.

Name	Variablenbereich	Typ	Funktion
Enable	IN	Bool	Fehlercode lesen

Name	Variablenbereich	Typ	Funktion
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Valid	OUT	Bool	FB-Daten sind gültig (sobald Enable = True)
Busy	OUT	Bool	FB-Funktion läuft (immer False, da PDO-Datum)
Error	OUT	Bool	FB-Fehler (immer False, da PDO-Datum)
Position	OUT	REAL	Achsposition in Nutzereinheiten, z.B. „mm“

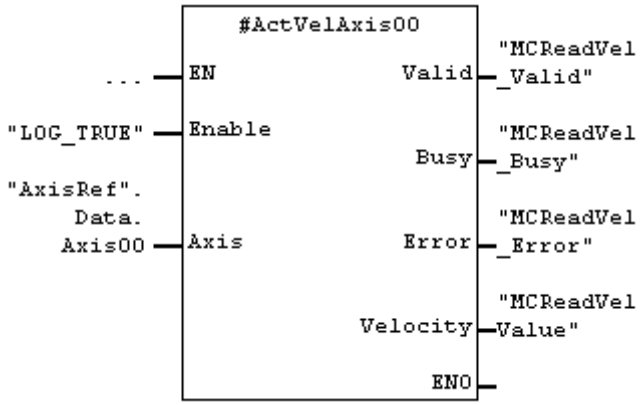


Im Stat-Bereich eines Container-FB's instanziierter MC_ReadActualPosition_E2 mit dem Instanznamen ActPosAxis00.

MC_ReadActualVelocity_E2 (FB)

Der MC_ReadActualVelocity_E2 stellt die Istgeschwindigkeit der Achse bereit.

Name	Variablenbereich	Typ	Funktion
Enable	IN	Bool	Fehlercode lesen
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Valid	OUT	Bool	FB-Daten sind gültig (sobald Enable = True)
Busy	OUT	Bool	FB-Funktion läuft (immer False, da PDO-Datum)
Error	OUT	Bool	FB-Fehler (immer False, da PDO-Datum)
Velocity	OUT	REAL	Achsgeschwindigkeit in Nutzereinheiten, z.B. „mm/s“



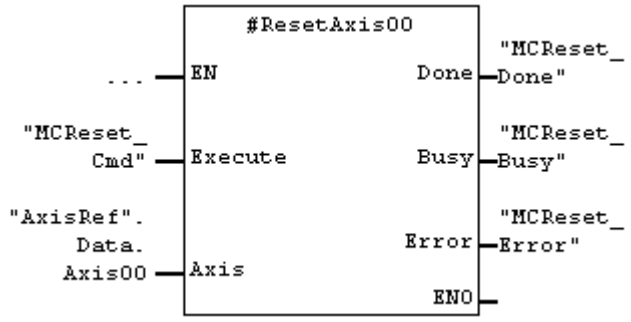
Im Stat-Bereich eines Container-FB's instanziierter MC_ReadActualVelocity_E2 mit dem Instanznamen ActVelAxis00.

MC_Reset_E2 (FB)

Mit demBaustein MC_Reset_E2 wird die Servoachse bei Fehler quittiert.

ⓘ	Es ist nur eine Instanz dieses FB's sinnvoll und erlaubt.

Name	Variablenbereich	Typ	Funktion
Execute	IN	Bool	0-1-Flanke quittiert Achse
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Done	OUT	Bool	FB-Funktion fertig und Achse ohne Fehler
Busy	OUT	Bool	FB-Funktion läuft
Error	OUT	Bool	Achse mit Fehler



Im Stat-Bereich instanziierter MC_Reset_E2 mit dem Instanznamen ResetAxis00.

MC_Power_E2 (FB)

Mit dem Baustein MC_Power_E2 wird die Achse bestromt (Motor hat Drehmoment bzw. Kraft) oder entstromt (stromlos).

ⓘ	Es ist nur eine Instanz dieses FB's sinnvoll und erlaubt.

Name	Variablenbereich	Typ	Funktion
Enable	IN	Bool	0-1-Flanke bestromt Achse,

Name	Variablenbereich	Typ	Funktion
			1-0-Flanke führt Schnellstopp mit anschließendem Stromlosschalten aus
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Status	OUT	Bool	1 bestromt 0 stromlos
Busy	OUT	Bool	FB-Funktion läuftt (Bestromen gerade aktiv)
Error	OUT	Bool	Achse mit Fehler

Funktionsbausteine im Profile Position Mode

Die Bausteine MC_Stop_E2, MC_MoveAbsolute_E2, MC_MoveRelative_E2 werden im Profile Position Mode ausgeführt. Neben den an den FB-Parametern verfügbaren Vorgaben sind folgende CANopen-Paramter über CANopen-SDO oder einfacher über das EPOS-Studio zu beschreiben.

EPOS-Parameter	Objekt-Index bzw. Subindex	Einstellung EPOS-Studio bzw. SDO-Transfer erforderlich	MC_Stop	MC_MoveAbsolute MC_MoveRelative
Position Window (Positionierfenster für DONE-Meldung)	0x6067 / 0x00 [Inkremente]	Ja		
Position Window Time (Zeit im Positionierfenster für DONE-Meldung)	0x6067 / 0x00 [ms]	Ja		
Software Position Limit (Software-Endgrenzen)	0x607D / 0x01 Minimal 0x607D / 0x02 Maximal [Inkremente] Die Software-Positionsüberwachung kann mit -2147483648 bzw. +2147483647 deaktiviert werden.	Ja		
Maximal Profile Velocity (Maximale Geschwindigkeit)	0x607F / 0x00 [U/min]	Ja		
QuickStop Deceleration	0x6085 / 0x00 [U/min/s]		Ja (Decel) [Units/s^2]	

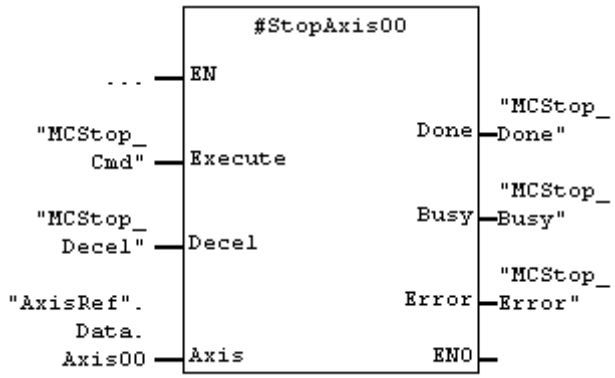
EPOS-Parameter	Objekt-Index bzw. Subindex	Einstellung EPOS-Studio bzw. SDO-Transfer erforderlich	MC_Stop	MC_MoveAbsolute MC_MoveRelative
(Schnellstopprampe)				
Max Acceleration (Maximale Beschleunigung/Verzögerung)	0x60C5 / 0x00 [U/min/s]	Ja		
Target Position (Zielposition oder Distanz von der aktuellen Sollposition)	0x607A / 0x00 [Inkremente]			Ja (Position bzw. Distance) [Units]
Profile Velocity (Sollgeschwindigkeit)	0x6081 / 0x00 [U/min]			Ja (Velocity) [Units/s]
Profile Acceleration (Sollbeschleunigung)	0x6083 / 0x00 [U/min/s]			Ja (Accel) [Units/s^2]
Profile Deceleration (Sollverzögerung)	0x6084 / 0x00 [U/min/s]			Ja (Decel) [Units/s^2]
Motion Profile Type (Geschwindigkeitsprofil)	0x6086 / 0x00	Ja 0=lineare Rampen 1=sin^2-Rampen		

MC_Stop_E2 (FB)

Mit dem Baustein MC_Stop_E2 wird die Achse gestoppt. Ein Stoppen ist nur bei bestromter Achse durchführbar.

ⓘ	Bei 0-1-Flanke am Enable-Eingang und bestromter Achse wird einmalig ein Stoppbefehl übertragen. Weitere Achsbewegungen (Neustarten) werden bei aktiviertem Stopp-Execute (=1) grundsätzlich geblockt. Es wird die QuickStopp-Funktion der Achse benutzt! Es ist nur eine Instanz dieses FB's sinnvoll und erlaubt.

Name	Variablenbereich	Typ	Funktion
Execute	IN	Bool	1 Stoppt Achse 0 Bewegungsfreigabe Achse
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Decel	IN	Dint	Stopprampe [Units/s^2]
Done	OUT	Bool	Achse gestoppt
Busy	OUT	Bool	FB-Funktion läuft
Error	OUT	Bool	Achse mit Fehler

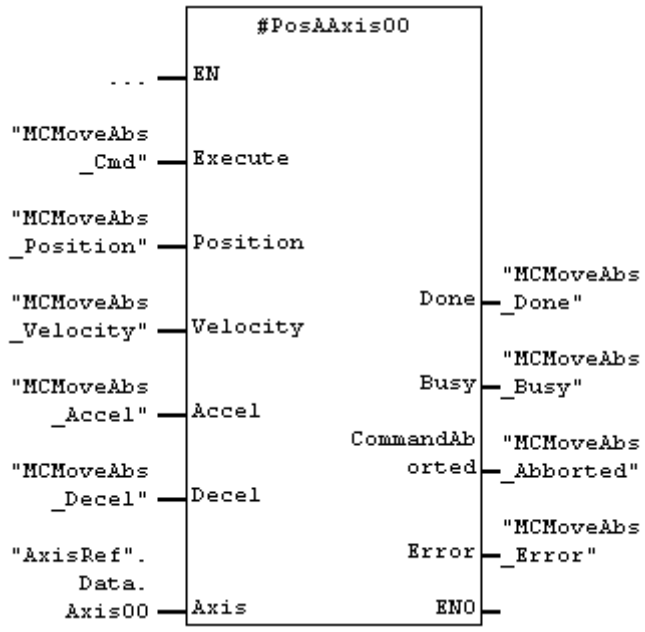


Im Stat-Bereich instanziierter MC_Stop_E2 mit dem Instanznamen StopAxis00.

MC_MoveAbsolute_E2 (FB)

Der Baustein MC_MoveAbsolute_E2 wird für absolute Positionierungen verwendet. Bezugspunkt der absoluten Position ist der durch eine Referenzfahrt festgelegte bzw. ermittelte absolute Bezugspunkt (auch mathematischer Nullpunkt).

Name	Variablenbereich	Typ	Funktion
Execute	IN	Bool	0-1-Flanke startet die Bewegung
Position	IN	Real	Absolute Position in Nutzereinheiten, z.B. „mm“
Velocity	IN	Real	Positioniergeschwindigkeit in Nutzereinheiten, z.B. „mm/s“
Accel	IN	Dint	Beschleunigung in Nutzereinheiten, z.B. „mm/s2“
Decel	IN	Dint	Verzögerung in Nutzereinheiten, z.B. „mm/s2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Done	OUT	Bool	Achse hat Zielposition erreicht
Busy	OUT	Bool	FB-Funktion läuft
CommandAbborted	OUT	Bool	FB-Kommando wurde abgebrochen durch neue Positionierung, Handfahren, Stromlosschalten, Stoppen etc.
Error	OUT	Bool	Achse mit Fehler



Im Stat-Bereich instanziierter MC_MoveAbsolute_E2 mit dem Instanznamen PosAAxis00.

MC_MoveRelative_E2 (FB)

Der Baustein MC_MoveRelative_E2 wird für relative Positionierungen (um eine Distanz) verwendet. Bezugspunkt für den Distanz ist die aktuelle Sollposition. Diese Art der Positionierung wird auch als Kettenpositionierung bezeichnet.

Name	Variablenbereich	Typ	Funktion
Execute	IN	Bool	0-1-Flanke startet die Bewegung
Distance	IN	Real	Distanz in Nutzereinheiten, z.B. „mm“
Velocity	IN	Real	Positioniergeschwindigkeit in Nutzereinheiten, z.B. „mm/s“
Accel	IN	Dint	Beschleunigung in Nutzereinheiten, z.B. „mm/s2“
Decel	IN	Dint	Verzögerung in Nutzereinheiten, z.B. „mm/s2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Done	OUT	Bool	Achse hat Zielposition erreicht (Distanz abgefahren)
Busy	OUT	Bool	FB-Funktion läuft
CommandAbborted	OUT	Bool	FB-Kommando wurde abgebrochen durch neue Positionierung, Handfahren, Stromlosschalten, Stoppen etc.
Error	OUT	Bool	Achse mit Fehler

Funktionsbausteine im Profile Velocity Mode

Die Bausteine MC_Stop_E2, MC_MoveVelocity_E2 und MC_Jog_E2 werden im Profile Velocity Mode ausgeführt. Neben den an den FB-Parametern verfügbaren Vorgaben sind folgende CANopen-

Paramter über CANopen-SDO oder einfacher über das EPOS-Studio zu beschreiben.

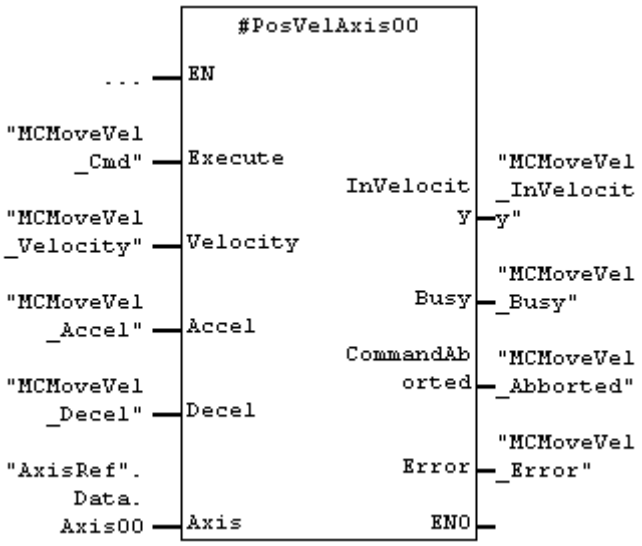
EPOS-Parameter	Objekt-Index bzw. Subindex	Einstellung EPOS-Studio bzw. SDO-Transfer erforderlich	MC_Stop	MC_MoveVelocity MC_Jog
Velocity Window (Geschwindigkeitsfenster für DONE-Meldung)	0x606D / 0x00 [U/min]	Ja		
Position Window Time (Zeit im Geschwindigkeitsfenster für DONE-Meldung)	0x606E / 0x00 [ms]	Ja		
Maximal Profile Velocity (Maximale Geschwindigkeit)	0x607F / 0x00 [U/min]	Ja		
QuickStop Deceleration (Schnellstopprampe)	0x6085 / 0x00 [U/min/s]		Ja (Decel) [Units/s^2]	
Max Acceleration (Maximale Beschleunigung/Verzögerung)	0x60C5 / 0x00 [U/min/s]	Ja		
Target Velocity (Zielgeschwindigkeit)	0x60FF / 0x00 [U/min]			Ja (Geschwindigkeit) [Units/s]
Profile Acceleration (Sollbeschleunigung)	0x6083 / 0x00 [U/min/s]			Ja (Accel) [Units/s^2]
Profile Deceleration (Sollverzögerung)	0x6084 / 0x00 [U/min/s]			Ja (Decel) [Units/s^2]
Motion Profile Type (Geschwindigkeitsprofil)	0x6086 / 0x00	Ja 0=lineare Rampen 1=sin^2-Rampen		

MC_MoveVelocity_E2 (FB)

Der Baustein MC_MoveVelocity_E2 wird für Endlosbewegungen verwendet.

ⓘ	Die Bewegung wird mit MC_Stop_E2 gestoppt. Wird am MC_MoveVelocity die Geschwindigkeit 0.0 Units/s vorgegeben, „fährt“ der Antrieb mit der Sollgeschwindigkeit 0.0 Units/s, d.h. die interne Sollwertvorgabe ist weiterhin aktiv. Soll sich die Geschwindigkeit fliegend ändern, kann die Geschwindigkeit über eine weitere Instanz von MC_MoveVelocity geändert werden oder das Execute an der „einen“ Instanz kann nachgetriggert werden (erneute 0-1-Flanke).

Name	Variablenbereich	Typ	Funktion
Execute	IN	Bool	0-1-Flanke startet die Bewegung
Velocity	IN	Real	Geschwindigkeit in Nutzereinheiten, z.B. „mm/s“
Accel	IN	Dint	Beschleunigung in Nutzereinheiten, z.B. „mm/s2“
Decel	IN	Dint	Verzögerung in Nutzereinheiten, z.B. „mm/s2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
InVelocity	OUT	Bool	Achse hat Zielgeschwindigkeit erreicht
Busy	OUT	Bool	FB-Funktion läuft
CommandAbborted	OUT	Bool	FB-Kommando wurde abgebrochen durch neue Positionierung, Handfahren, Stromlosschalten, Stoppen etc.
Error	OUT	Bool	Achse mit Fehler



Im Stat-Bereich instanziierter MC_MoveVelocity_E2 mit dem Instanznamen PosVelAxis00.

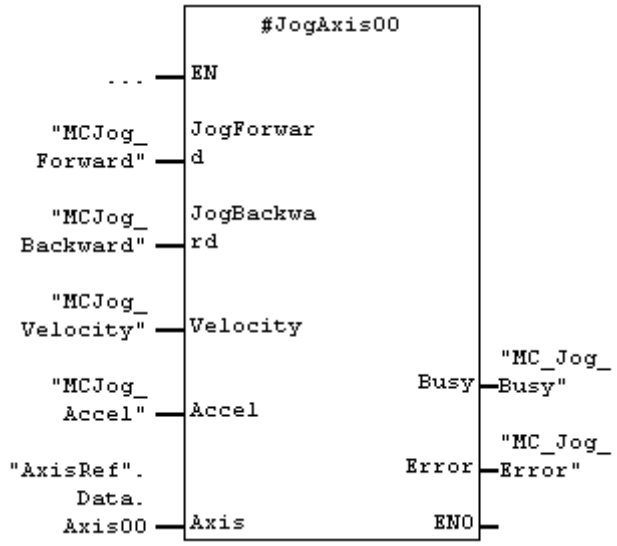
MC_Jog_E2 (FB)

Der Baustein MC_Jog_E2 wird verwendet, um die Achse „manuell“ zu bewegen (auch tippen genannt). Beide Richtungen sind möglich. Der FB funktioniert wie der MC_MoveVelocity, nur das richtungsabhängig gefahren werden kann.

ⓘ	Wird der Jog-Betrieb beendet (JogForward und JogBackward beide False), wird intern ein QuickStop ausgelöst, wobei jedoch im Unterschied zum „normalen“ Stoppen über MC_Stop_E2 die Accel-Rampe vom FB-Eingang verwendet wird, gegebenenfalls also „weicher“ gestoppt wird.

	Man beachte auch, dass der Accel-Wert gleichzeitig für die Beschleunigung und Verzögerung gilt.

Name	Variablenbereich	Typ	Funktion
JogForward	IN	Bool	0-1-Flanke startet Joggen im Uhrzeigersinn, 1-0-Flanke stoppt die Achse
JogBackward	IN	Bool	0-1-Flanke startet Joggen entgegen dem Uhrzeigersinn, 1-0-Flanke stoppt die Achse
Velocity	IN	Real	Handgeschwindigkeit in Nutzereinheiten, z.B. „mm/s“
Accel	IN	Dint	Beschleunigung/Verzögerung in Nutzereinheiten, z.B. „mm/s2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Busy	OUT	Bool	FB-Funktion läuft
Error	OUT	Bool	Achse mit Fehler



Im Stat-Bereich instanziierter MC_Jog_E2 mit dem Instanznamen JogAxis00.

Funktionsbausteine im Homing Mode

Die Bausteine MC_Home_E2 wird im Homing Mode ausgeführt. Neben den an den FB-Parametern verfügbaren Vorgaben sind folgende CANopen-Paramter über CANopen-SDO oder einfacher über das EPOS-Studio zu beschreiben.

EPOS-Parameter	Objekt-Index bzw. Subindex	Einstellung EPOS-Studio bzw. SDO-Transfer erforderlich	MC_Home
Homing Method	0x6098 / 0x00	Ja	

EPOS-Parameter	Objekt-Index bzw. Subindex	Einstellung EPOS-Studio bzw. SDO-Transfer erforderlich	MC_Home
(Methode des Referenzierens über Endschalter, mechanische Anschläge, Position setzen usw.)			
Homing Speeds (Suchgeschwindigkeit beim Referenzieren)	0x6099 / 0x00 [U/min]	Ja	
Homing Acceleration (Rampen beim Referenzieren)	0x609A / 0x00 [U/min/s]	Ja	
Home Offset (Nach dem eigentlichen Referenzieren wird diese Strecke noch abgefahren, dann der Wert des Parameters Home position als neue Istposition gesetzt)	0x607C / 0x00 [Inkremente]	Ja	
Current Threshold for homing modes -1 bis -4 (Wenn beim Referenzieren gegen den Anschlag gefahren werden soll, Stromgrenze)	0x2080 [mA]	Ja	
Home position (Setzposition nach erfolgreichem Homing und Abfahren der Strecke im Home Offset)	0x2081 / 0x00 [Inkremente]		Ja (Setzposition nach erfolgter Homing-Prozedur) [Units]
Motion Profile Type (Geschwindigkeitsprofil)	0x6086 / 0x00	Ja 0=lineare Rampen 1=sin^2-Rampen	

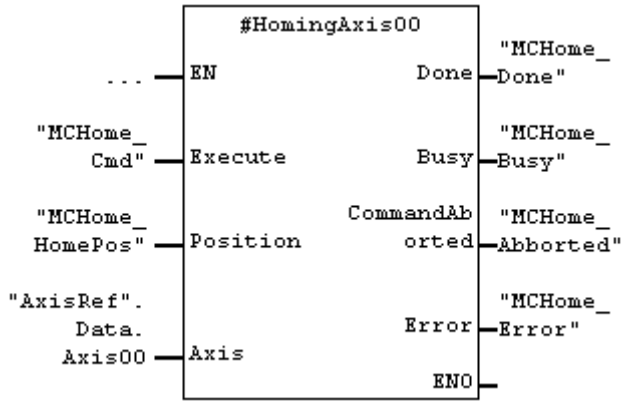
MC_Home_E2 (FB)

Der Baustein MC_Home_E2 wird verwendet, um den mathematischen Bezugspunkt für folgende Positionierungen der Achse zu definieren. Die Referenzierarten (Modi) sind der EPOS2-Beschreibung zu entnehmen.

Im Laboraufbau wurden die Modi „35“ und „-3“ getestet.

ⓘ	Es ist nur eine Instanz dieses FB's sinnvoll und erlaubt.

Name	Variablenbereich	Typ	Funktion
Execute	IN	Bool	0-1-Flanke startet die Referenzfahrt
Position	IN	REAL	Setzposition (neue Sollposition) in Nutzereinheiten, z.B. „mm“, der Wert des Parameter 0x2081/0x00 wird beschrieben
Axis	IN_OUT	AxisRefE2Type (UDT)	Achsreferenz (Achsverweis)
Done	OUT	Bool	Referenzfahrt (inkl. Abfahren der Distanz im Parameter 0x607C/0x00 Home Offset) beendet, Setzposition gesetzt, der mathematische Nullpunkt und gegebenenfalls die Software-Endgrenzen sind gültig
Busy	OUT	Bool	FB-Funktion läuft
CommandAbborted	OUT	Bool	FB-Kommando wurde abgebrochen durch neue Positionierung, Handfahren, Stromlosschalten, Stoppen etc.
Error	OUT	Bool	Achse mit Fehler oder Referenzierung (Homing) mit Fehler beendet



Im Stat-Bereich instanziierter MC_Home_E2 mit dem Instanznamen HomingAxis00.

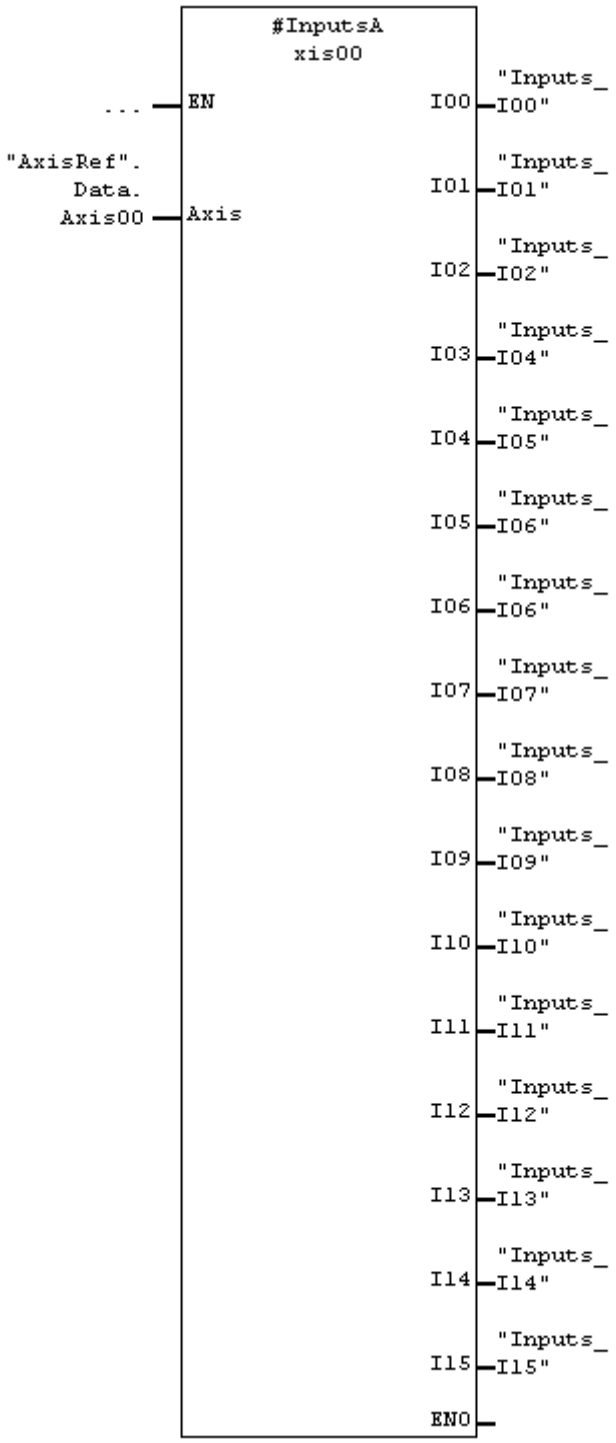
Spezial-FB's

E2_Input (FB)

Der Baustein E2_Input ist eine Spezialversion für die EPOS2-Achse, man beachte unterschiedliche Versionen der EPOS2-Familie.

Name	Variablenbereich	Typ	Funktion
Axis	IN_OUT	AxisRefE2Type	Achsreferenz (Achsverweis)

Name	Variablenbereich	Typ	Funktion
		(UDT)	
I00 bis I15	OUT	Bool	Eingänge 0 bis 15 als Status



Im Stat-Bereich instanziierter E2_Input mit dem Instanznamen InAxis00.

InDataE2Type (UDT)

Dieser Datentyp ist bei Verwendung in einem Datenbaustein mit einem Namen, z.B. Axis00 zu instantiieren. Pro Achse wird genau 1 Instanz benötigt. Die Instanz-Daten entsprechen T-PDO-Daten der EPOS2-Achse.

ⓘ	Am besten werden alle Instanzen (der verschiedenen Achsen) von InDataE2Type in einem separaten DB (z.B. „TPDODATA“) angelegt.

```
wStatusWord     :   WORD;       // TPDO1, async, 0x6041 + 0x00, Status word
wDigInWord      :   WORD;       // TPDO1, async, 0x2071 + 0x01, Input state
byActModeOfOp   :   BYTE;       // TPDO1, async, 0x6061 + 0x00, Actual mode of operation
diActPosition   :   DINT;       // TPDO2, async, 0x6064 + 0x00, Actual position [pq Enc.]
diActVelocity   :   DINT;       // TPDO2, async, 0x606C + 0x00, Actual velocity [rev/min]
```

OutDataE2Type (UDT)

Dieser Datentyp ist bei Verwendung in einem Datenbaustein mit einem Namen, z.B. Axis00 zu instanziieren. Pro Achse wird genau 1 Instanz benötigt. Die Instanz-Daten entsprechen R-PDO-Daten der EPOS2-Achse.

ⓘ	Am besten werden alle Instanzen (der verschiedenen Achsen) von OutDataE2Type in einem separaten DB (z.B. „RPDODATA“) angelegt.

```
wControlWord        :   WORD;       // RPDO1, async, 0x6040 + 0x00, Control word
diTarget            :   DINT;       // RPDO1, async, 0x607a + 0x00, Target [pq Enc.]
byModeOfOp          :   BYTE;       // RPDO1, async, 0x6060 + 0x00, Mode of operation
diProfVelocity      :   DINT;       // RPDO2, async, 0x6081 + 0x00, Profile velocity [rev/min]
diHomePosition      :   DINT;       // RPDO2, async, 0x2081 + 0x00, Home position [pq Enc.]
diProfAccel         :   DINT;       // RPDO3, async, 0x6083 + 0x00, Prof. acceleartion [rev/min/s]
diProfDecel         :   DINT;       // RPDO3, async, 0x6084 + 0x00, Prof. deceleration [rev/min/s]
diQuickStopDecel    :   DINT;       // RPDO4, async, 0x6085 + 0x00, Prof. deceleration [rev/min/s]
diSpeed             :   DINT;       // RPDO4, async, 0x60FF + 0x00, Prof. deceleration [rev/min]
```

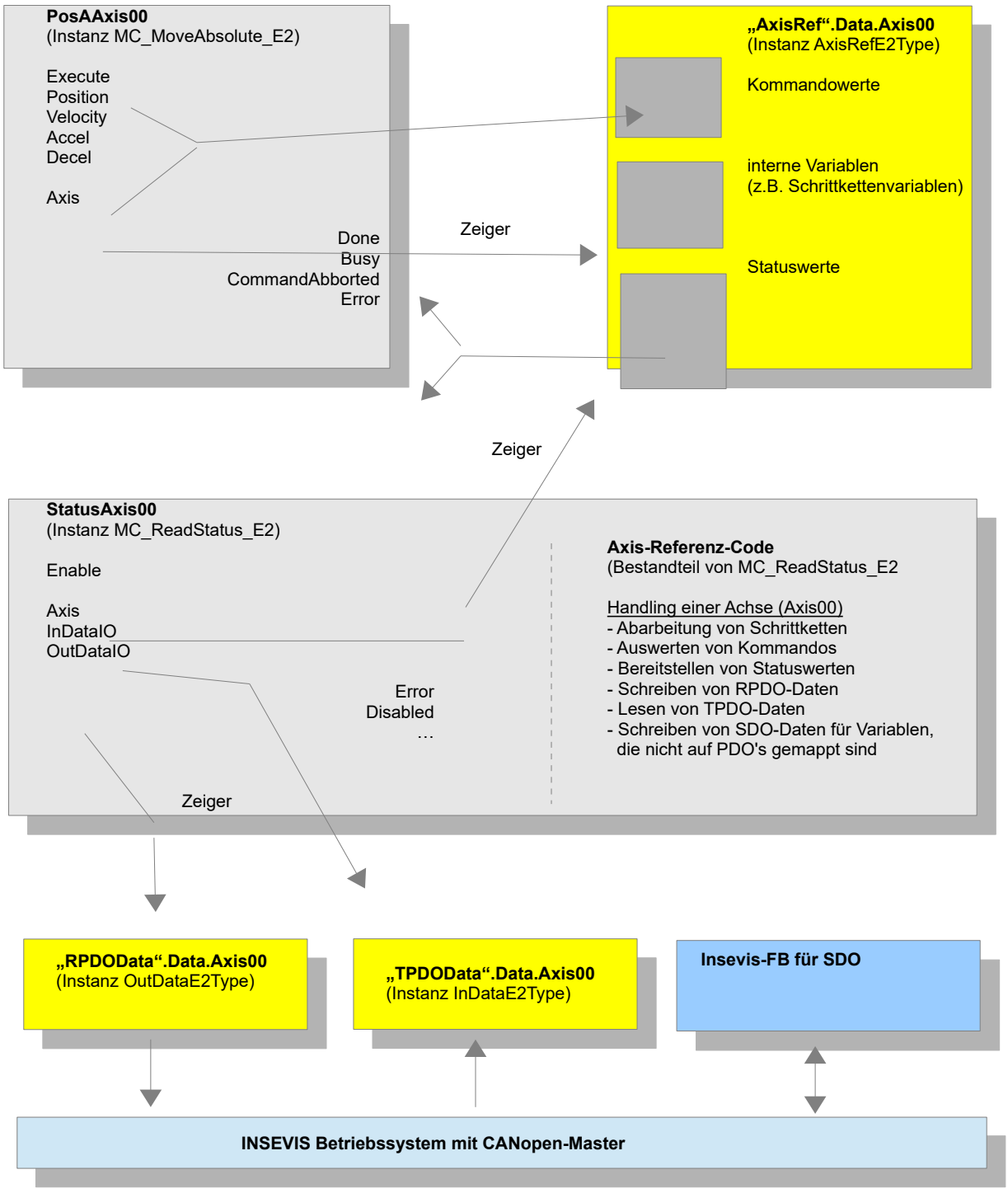
AxisRefE2Type

Dieser Datentyp wird intern zur Referenzierung der Achse benötigt. Alle Instanzen von MC-Bausteinen bestimmten damit die verbundene Achse. Da die instantiierten Variablen als IN_OUT übergeben werden, ist der Kopieraufwand gering. Der Baustein MC_ReadStatus_E2 verwendet die mit der Achsreferenz übergebenen Werte zur Abarbeitung der Schrittketten.

ⓘ	Am besten werden alle Instanzen (der verschiedenen Achsen) von AxisrefE2Type in einem separaten DB (z.B. „AXISREF“) angelegt.

Datenfluss am Beispiel einer MC-Block-Instanz

Folgende Grafik illustriert die Verwendung und den Datenfluss eines MC-Bausteines für eine Achse mit dem Namen Axis00.

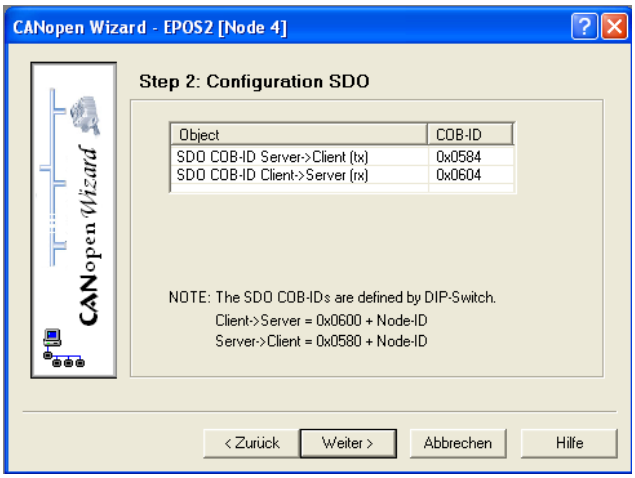
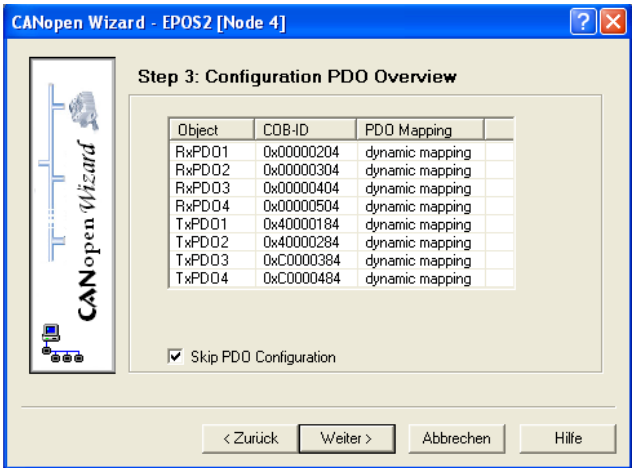


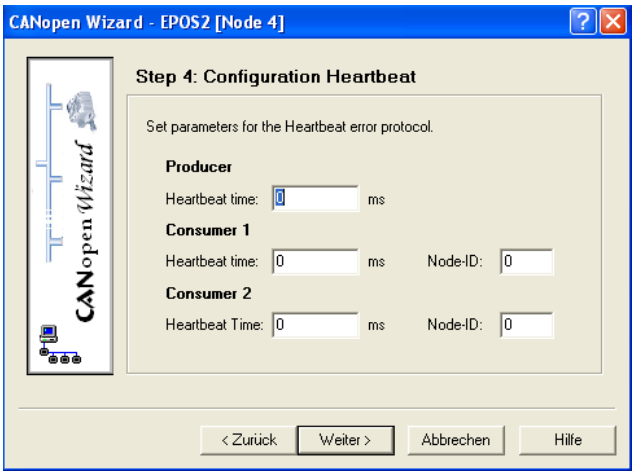
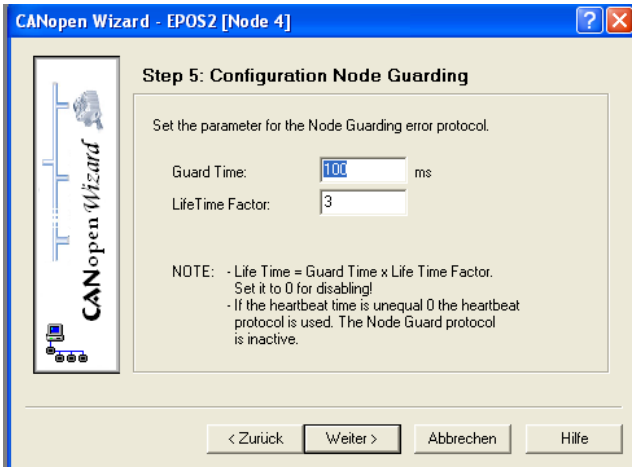
CANopen-Konfiguration mit dem EPOS-Studio

Grundsätzlich kann hier nicht auf die allgemeine Konfiguration eines EPOS2-Servoantriebs eingegangen werden. Wichtig für den CANopen-Teil ist hier lediglich, dass über den CANopen-Wizard die eigentliche Konfiguration geprüft (nach Konfiguration mit der ConfigStage und Anlauf der INSEVIS-SPS) werden sollte, da die Einstellungen der CANopen-Slaves durch die INSEVIS-SPS selbst erfolgen.

Zudem ist die CANopen-Node-ID am entsprechenden Jumper (CAN-ID) einzustellen. Die CAN-Bitrate (0x2001) steht per default auf dem Wert „9“ (Automatische Erkennung), diesen Wert sollte man der Einfachheit halber unverändert lassen.

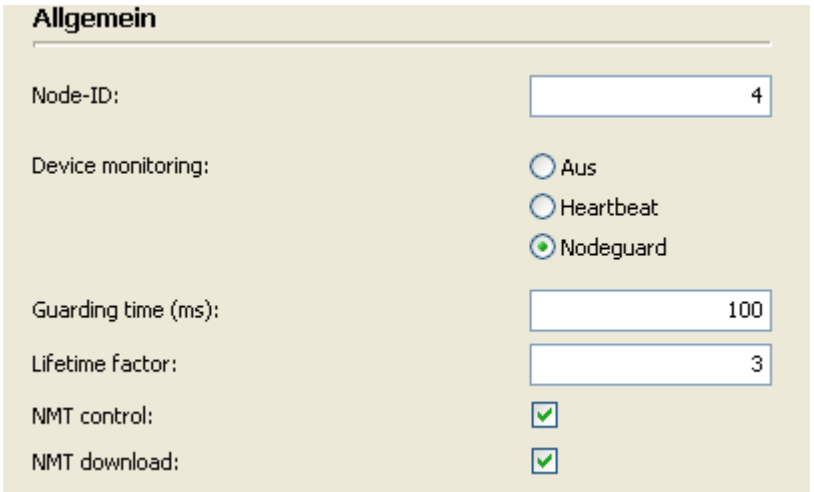
ⓘ	Verwenden Sie das „Object Dictionary“, um aktuelle Werte zu prüfen bzw. zu verändern.

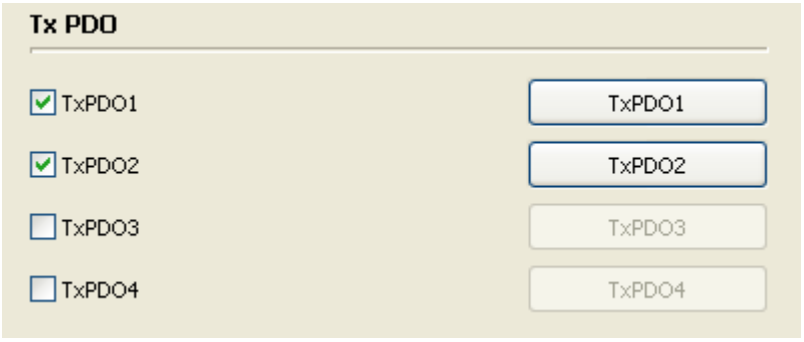
	Überprüfen der COB-ID's für die SDO's, i.d.R. Keine Veränderungen notwendig.
	Die PDO-Konfiguration kann komplett übersprungen werden.

	Eine Heartbeat-Überwachung wird nicht verwendet.
	Nodeguarding wird durch die INSEVIS-SPS konfiguriert und aktiviert.

Slave-Konfiguration mit ConfigStage

Mit der ConfigStage-Software werden unter anderem der CANopen-Master und jeder CANopen-Slave konfiguriert. Zudem wird die Verbindung von SPS-Daten (z.B. Datenbaustein und Offset im Datenbaustein) zu den CANopen-Daten (R-PDO's, T-PDO's) definiert.

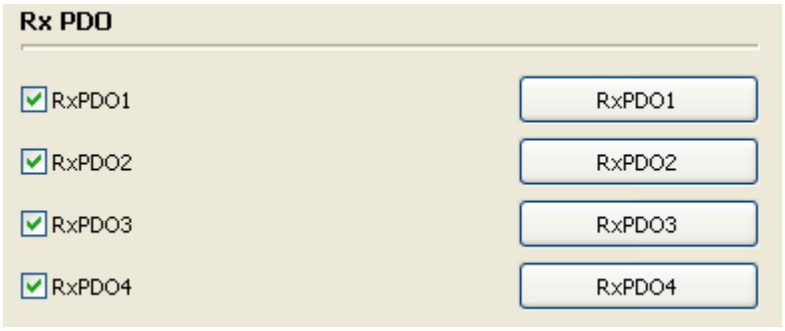
	Festlegen der Node-ID und des Guardings EPOS2 unterstützt Nodeguarding CANopen-Einstellungen (wie COB-ID's) zum EPOS2 laden



TPDO (EPOS2→ CANopen-Master)
Es werden für die Empfangsrichtung 2 T-PDO's benötigt. Download Kommunikationsparameter und des Mappings sind zu aktivieren.

Übertragungsverhalten TPDO1
Typ: 254
keine Sperrzeit

Übertragungsverhalten TPDO2
Typ: 254
Sperrzeit von z.B. 100ms definieren!



RPDO (CANopen-Master → EPOS2)

Es werden für die Senderichtung 4 R-PDO's benötigt. Download Kommunikationsparameter und des Mappings sind zu aktivieren.

Übertragungsverhalten RPDO1
Typ: 254
keine Sperrzeit

Übertragungsverhalten RPDO2
Typ: 254
keine Sperrzeit

Übertragungsverhalten RPDO3
Typ: 254
keine Sperrzeit

Übertragungsverhalten RPDO4
Typ: 254
keine Sperrzeit

Zusätzliche Konfiguration über SDO

Es werden keine zusätzlichen SDO's benötigt. Gegebenenfalls kann man hier CANopen-Daten übertragen, die man ansonsten über das EPOS-Studio übertragen müsste.

Mapping T-PDO1

Offset im Datenbereich (z.B. Datenbaustein) einer Instanz vom Typ „InDataE2Type“: **0** (Byte-Offset)

Nummer	Index	Subindex	Größe	Erklärung
1	0x6041	0x00	16 Bit/Word	Statuswort DS402
2	0x2071	0x01	16 Bit /Word	Eingangswort der DE
3	0x6061	0x00	16 Bit/Word	Aktuelle Betriebsart DS402

Mapping T-PDO2

Offset im Datenbereich (z.B. Datenbaustein) einer Instanz vom Typ „InDataE2Type“: **6** (Byte-Offset)

Nummer	Index	Subindex	Größe	Erklärung
1	0x6064	0x00	32 Bit/DWord	Aktuelle Position [Post quadrature encoder]
2	0x606C	0x00	32 Bit/DWord	Aktuelle Geschwindigkeit [U/min]

Mapping R-PDO1

Offset im Datenbereich (z.B. Datenbaustein) einer Instanz vom Typ „OutDataE2Type“: **0** (Byte-Offset)

Nummer	Index	Subindex	Größe	Erklärung
1	0x6040	0x00	16 Bit/Word	Steuerwort DS402
2	0x607A	0x00	32Bit/DWord	Target [Post quadrature encoder]
3	0x6060	0x00	16 Bit/Word	Betriebsart DS402

Mapping R-PDO2

Offset im Datenbereich (z.B. Datenbaustein) einer Instanz vom Typ „OutDataE2Type“: **8** (Byte-Offset)

Nummer	Index	Subindex	Größe	Erklärung
1	0x6081	0x00	32 Bit/DWord	Profilgeschwindigkeit [U/min]
2	0x2081	0x00	32 Bit/DWord	Home-Position [Post quadrature encoder]

Mapping R-PDO3

Offset im Datenbereich (z.B. Datenbaustein) einer Instanz vom Typ „OutDataE2Type“: **16** (Byte-Offset)

Nummer	Index	Subindex	Größe	Erklärung
1	0x6083	0	32	Profilbeschleunigung [U/min/s]

Nummer	Index	Subindex	Größe	Erklärung
			Bit/DWord	
2	0x6084	0	32 Bit/DWord	Profilverzögerung [U/min/s]

Mapping R-PDO4

Offset im Datenbereich (z.B. Datenbaustein) einer Instanz vom Typ „OutDataE2Type“: **24** (Byte-Offset)

Nummer	Index	Subindex	Größe	Erklärung
1	0x6085	0	32 Bit/DWord	Schnellstopprampe [U/min/s]
2	0x60FF	0	32 Bit/DWord	Geschwindigkeit im Profile Velocity Mode [U/min]

Zusätzliche SDO-Übertragung nach PDO-Mapping

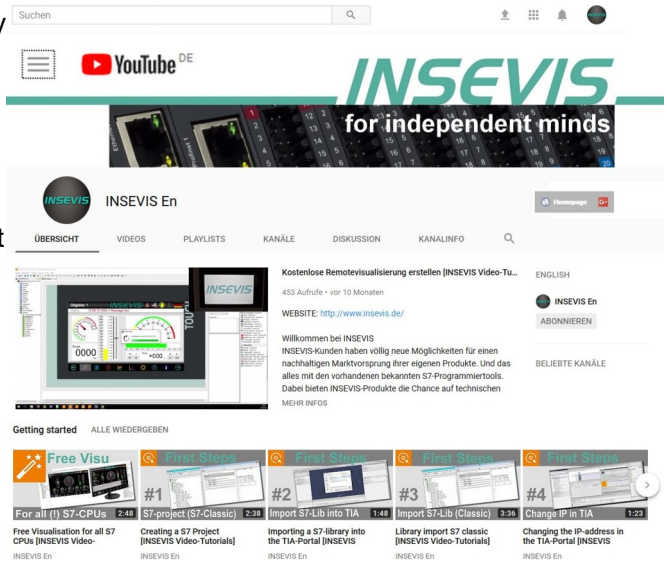
Nummer	Index	Subindex	Größe	Wert	Erklärung

S7-Beispiel-Programm

Das Beispielprojekt besteht aus einem S7-Programm, das die Verwendung der MC-Blöcke veranschaulicht.

Hint for better understanding by additional information

In the English YouTube-channel INSEVIS EN we supply different playlists with handling videos for single details. This will help you to get familiar with INSEVIS much faster.



Please download the referring manual from the download area of our English website insevis.com to get familiar with INSEVIS technology in detail.

Do you want to inform us about necessary increments or errors or do you want to provide us with your sample programs to offer it for free to all customers? Gladly we would provide your program -if you wish with the authors name- to all other customers of INSEVIS.

Hint to different versions of the sample programs

There could be older versions in delivery scope of the sample programs too. These were not updated and converted to the newest programming tool versions to allow access by older programming tools too. INSEVIS sample programs will be created in the present newest Siemens-programming tool always.

SAMPLE DESCRIPTION

Index of content

Motivation

Manufacturer-specific S7-blocks will be offered from different vendors for an easy implementation of their own drive technology into the world of Simatic- and Simatic-compatible PLCs since years. This is often done by a very effective S7-block, adapted to the specials of the vendors drive, containing a monolithic and mostly customized interface and specialized in a certain bus system (normally Profibus DP, also Interbus S and CANopen based on fieldbus-master modules of other manufacturers).

The PLCopen (http://www.plcopen.org) as an international organisation is dedicated to reduce the effords for engineering by using general software interfaces. In the area of drive technology standards were defined, a certification of drives with implemented interfaces is possible. By using bus systems like CANopen with drive interfaces (DS402 drive profile) the effords for the adaption onto a certain bus protocol is unimportant.

In the following the operation on a servo drive Maxon EPOS2 24/5 (http://www.maxonmotor.de) is described. The software was created for INSEVIS-PLC and is based on the PLCopen-standard

On following devices the software test was done:

EPOS2

Testing device	:	EPOS2 24/5
Software-version	:	0x2122
Hardware-version	:	0x6220
EPOS-Studio	:	1.44 revision 1

INSEVIS

Testing device	:	CC300V
operating system	:	2.0.23
S7-Library	:	Insevis_S7-library_from_2_0_22

Actually there will not be supported all possible modes, like e.g. the „Master Encoder Mode“, because the hardware platforms are to assortetly, to work with te same encoder-type. On demand a MC_Gear_E2 could be implemented generally later on.

Company inmotec Automation GmbH (support@inmotec.de) creates and expands drive specific software for INSEVIS-S7-controllers.

General principles of the software-design

1. All drive functions (so called Motion-Control-Blocks MC_) will be implemented as single function blocks, e.g. the function block „MC_Power_E2“, a S7-FB, is used to enable the servo drive. Because the motor does need not only to be enabled but also has to do motion functions, more function blocks are neccessary. Of course multiple axes were supported too. To prevent a various number of instances of an function block with separate instance blocks, an instanciation of function blocks in the STAT area of the variables definiton of the „container“-funktion block is recommended.
2. The MC-Blocks use no global resources as M-merker, T-times or Z-counter, but their instanciable IEC-variantes.
3. All drive functions of the INSEVIS-PLC communicate via asynchronous CANopen-PDO's reg. DS301, so that the efford for communication (bus load) is reduced. At the drive profile DS402 will be used operating modes only, what do not require equidistant transfers of demand values. The so called „interpolated mode“ will not be used.
4. The function blocks will be created in origin with SCL (Structured Control Language), an engineering-option to Step7 of Siemens. The use of these function blocks does not need a preinstalled SCL-package on the programming PC of the user.
5. To absorb diversities of the drives and name conflicts of already existing blocks from custom libraries (e.g. at the technology- PLC of Siemens), the MC-Blocks get a postfix like „_C3“ in reference to the regarding drive. There needs to be notified, that the instance name (in the sample „Axis00“) is not touched while swapping drives.
6. Because blocks do not reference each other, block-addresses (absolut numbers) can be adapted to the demands of the users program.

Drive functions (MC-blocks and -types)

An operation with encoders is recommended generally, if positioning applications are to do. If the measurement of rotation speed by Hall sensors, no positioning applications can be made. Only an operation by rotation speed is worthwile. It is recommended to drive the motor at least with 1000 r/min.

MC-Block/Symbol	Address	Function
MC_ReadStatus_E2	FB40	Visualization of drive states (currentless, stopping, remaining idle, profil based motion functions active, endlessmove active, synchronized motion functions active, reference move active)
MC_ReadAxisError_E2	FB41	Visualization of the error code of the drive
MC_ReadActualPosition_E2	FB42	Visualization of the actual postion of the motor
MC_ReadActualVelocity_E2	FB43	Visualization of actual velocity of the motor
MC_Reset_E2	FB44	Reset error in the drive
MC_Power_E2	FB45	Enable power stage to the motor or stop/disable fast as possible
MC_Stop_E2	FB46	Stop the motor
MC_MoveAbsolute_E2	FB47	Move to an absolute position
MC_MoveRelative_E2	FB48	Move a relative distance
MC_MoveVelocity_E2	FB50	Endless move (rotation speed assign)
MC_Home_E2	FB52	Execute homing move
MC_Jog_E2	FB53	Jog+/- move, stops on the software-end-delimiters
E2_Input	FB54	Read out inputs
InDataE2Type	UDT100	Data type for input data CANopen, instanciate once per axis
OutDataE2Type	UDT101	Data type for output data CANopen, instanciate once per axis
SWPosE2Type	UDT102	Data type state word CANopen, INTERNAL USE ONLY
CWPosE2Type	UDT103	Data type control word CANopen, , INTERNAL USE ONLY
AxisRefE2Type	UDT104	Data type axis reference, instanciate once per axis

Settings for EPOS2

Because EPOS2 uses fixed formates for positioning (post-quad-increments), velocities (r/min) and ramps (r/min/s), an implementation is worthwile, what allows an assign of better to understand engineer values.

The user has to write following parameters into the axis reference, so that it is possible to program directly in user units, user units/s and user units/s[2:]

```
      L     1.000000e+001                  // e.g. 10 mm feed/turns
      T     "AxisRef".Data.Axis00.fMechanicPitch
       L     4.000000e+003                     // increments Encoder
after
                                             // quadruplication,e.g. 4000
at
```

```
                                        // a 1000-impuls-Encoder
      T     "AxisRef".Data.Axis00.fEncPQPerMotRev
```

Also the CANopen-Node-ID is to assign, e.g. foracyclic requests (error codes).

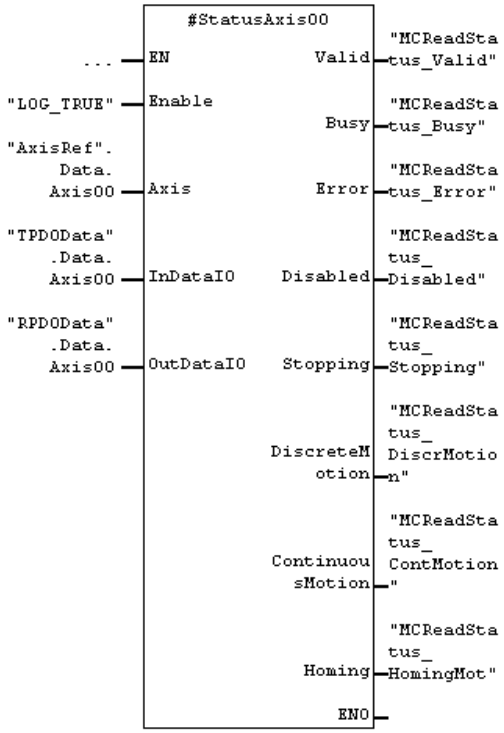
```
          L     4
      T     "AxisRef".Data.Axis00.iNode
```

MC_ReadStatus_E2 (FB)

The MC_ReadStatus_E2 will be used for visualization (State generation) of different drive states. With these information the PLC-programm can watch all activities of the drive.

ⓘ	This block MUST be implemented into the PLC-program, because beside the state generation the complete axis reference will be processed. That´s why this block size is larger than at the other MC-blocks. Explanations are made in the chapter axis reference. It is only one instance of this FB allowed and reasonable.

Name	Variables area	Type	Function
Enable	IN	Bool	Activate state generation For the processing of the axis reference the enable-input is not important, but anyway the FB MUST be called.
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
InDataIO	IN_OUT	InDataE2Type (UDT)	Reference to IO-data (input-data CANopen)
OutDataIO	IN_OUT	OutDataE2Type (UDT)	Reference to IO-data (output data CANopen)
Valid	OUT	Bool	FB-data are valid (as long Enable = True)
Busy	OUT	Bool	FB-function runs (always False, because created from PDO-data)
Error	OUT	Bool	Axis with error
Disabled	OUT	Bool	Axis is disabled
Stopping	OUT	Bool	Axis is stopping
DiscreteMotion	OUT	Bool	Axis is positioning
ContinuousMotion	OUT	Bool	Axis is positioning endless
Homing	OUT	Bool	Axis executes „homing-move“



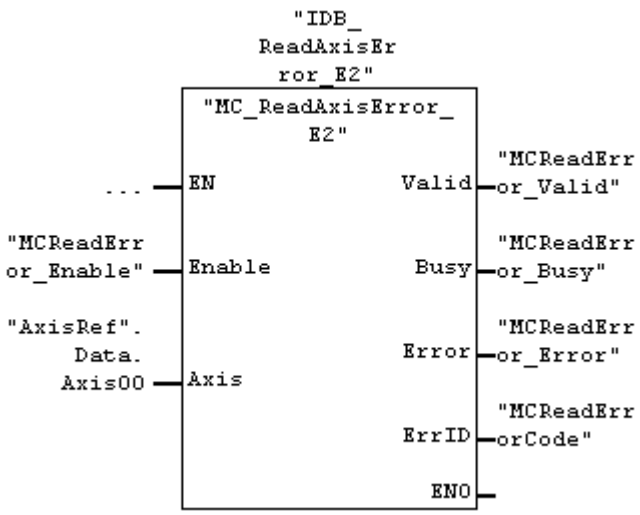
In the STAT-area of a Container-FB instanciated MC_ReadStatus_E2 with the instance name StatusAxis00.

MC_ReadAxisError_E2 (FB)

The MC_ReadAxisError_E2 will be used for visualization of the error code of the axis.
The meaning of the error code is mentioned in the drives help manual.

Name	Variables area	Type	Function
Enable	IN	Bool	Read error code
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Valid	OUT	Bool	FB-data are valid (as long Enable = True)
Busy	OUT	Bool	FB-function (SDO-Transfer) runs (actally)
Error	OUT	Bool	Axis with error
ErrorID	OUT	DWORD	Error code of axis (here 32-bit)

ⓘ	The EPOS2-CANopen-object 0x1003 will be red on subindex 1, what contains the actual error code. The object will be red by CANopen-SDO, when the error bit in the stat word will be set on TRUE (edge) or the enable-input will be set on TRUE (edge). Occures after a TIME-OUT-time of 150ms no answer from the device or an error will be reported at the SDO-transfer, an error code DW#16#FFFFFFFF (SDO-transfer-error) will be displayed. **The FB in the actual version can not be instanciated in the STAT-area of an instance-DB's of an higher ranked FB's! It must be created a seperate instance for the MC_ReadAxisError_E2.** It is only one instance of this FB allowed and reasonable.

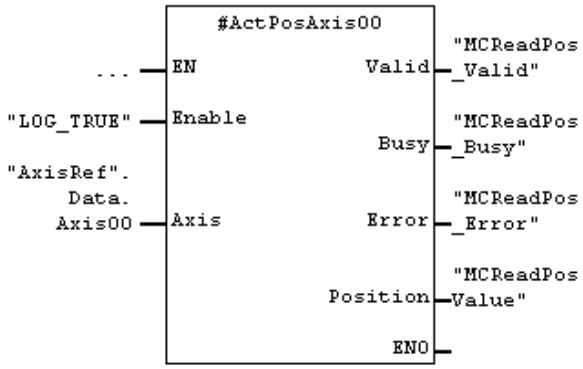


Separate instanciated MC_ReadAxisError_E2 with the instance name ErrorAxis00.

MC_ReadActualPosition_E2 (FB)

The MC_ReadActualPosition_E2 provides the absolute position of the axis.

Name	Variables area	Type	Function
Enable	IN	Bool	Read actual position
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Valid	OUT	Bool	FB-data are valid (as long Enable = True)
Busy	OUT	Bool	FB-function runs (always False, because PDO-date)
Error	OUT	Bool	FB-error (always False, because PDO-date)
Position	OUT	REAL	Axis position in user units, e.g. „mm“

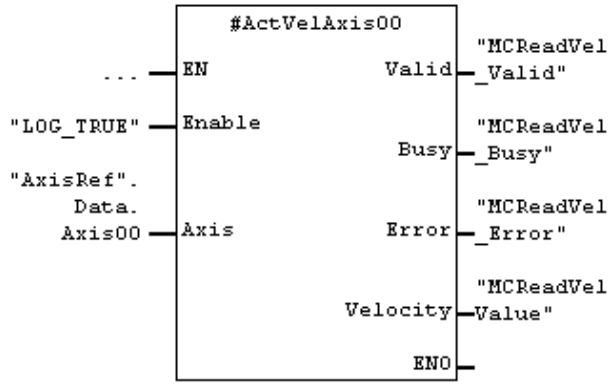


In the STAT-area of a Container-FB instanciated MC_ReadActualPosition_E2 with the instance name ActPosAxis00.

MC_ReadActualVelocity_E2 (FB)

The MC_ReadActualVelocity_E2 provides the actual velocity of the axis.

Name	Variables area	Type	Function
Enable	IN	Bool	Read actual velocity
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Valid	OUT	Bool	FB-data are valid (as long Enable = True)
Busy	OUT	Bool	FB-function runs (always False, because PDO-date)
Error	OUT	Bool	FB-error (always False, because PDO-date)
Velocity	OUT	REAL	Axis velocity in user units. e.g. „mm/s“



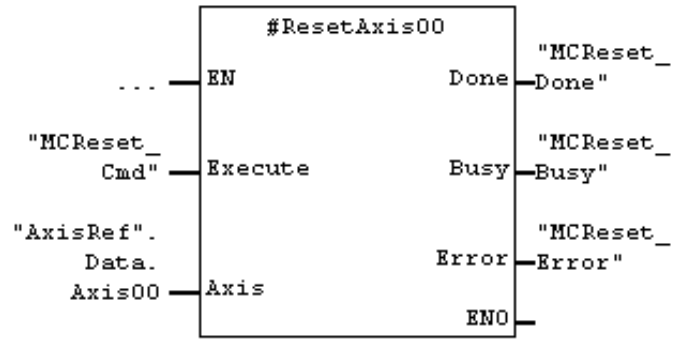
In the STAT-area of a Container-FB instanciated MC_ReadActualVelocity_E2 with the instance name ActVelAxis00.

MC_Reset_E2 (FB)

With the block MC_Reset_E2 the servo axis error will be reset.

ⓘ	It is only one instance of this FB allowed and reasonable.

Name	Variables area	Type	Function
Execute	IN	Bool	0-1-edge receipes axis
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Done	OUT	Bool	FB-function ready and axis without error
Busy	OUT	Bool	FB-function is executed/runs
Error	OUT	Bool	Axis with error



In the STAT-area instanciated MC_Reset_E3 with the instance name ResetAxis00.

MC_Power_E2 (FB)

With the block MC_Power_E2 the axis will be enabled or disabled

ⓘ	It is only one instance of this FB allowed and reasonable.

Name	Variables area	Type	Function
Enable	IN	Bool	0-1-edge enables power stage axis, 1-0-edge executes an immediate stop

Name	Variables area	Type	Function
			with following switching to currentless
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Status	OUT	Bool	1 power stage enabled 0 disabled
Busy	OUT	Bool	Function enable power stage even active
Error	OUT	Bool	Axis with error

Function blocks in the profile Position Mode

The blocks MC_Stop_E2, MC_MoveAbsolute_E2, MC_MoveRelative_E2 will be processed in the profile „Position Mode“. Beside the assigns on the FB-parameters following CANopen-paramters are to assign via CANopen-SDO or easier via the EPOS-Studio.

EPOS-Parameter	Object-index resp. subindex	Setting EPOS-Studio resp. SDO-transfer necessary	MC_Stop	MC_MoveAbsolute MC_MoveRelative
Position Window (positioning window for DONE-message)	0x6067 / 0x00 [increments]	Yes		
Position Window Time (time in the positioning window for DONE-message)	0x6067 / 0x00 [ms]	Yes		
Software Position Limit	0x607D / 0x01 minimal 0x607D / 0x02 maximal [increments] The software-position-control can be dectivated with -2147483648 resp. +2147483647	Yes		
maximal Profile Velocity	0x607F / 0x00 [r/min]	Yes		
QuickStop Deceleration	0x6085 / 0x00 [r/min/s]		Yes (Decel) [Units/s^2]	
Max Acceleration/ Deceleration	0x60C5 / 0x00 [r/min/s]	Yes		
Target Position (or distance from theactual nominal position)	0x607A / 0x00 [increments]			Yes (position resp. distance) [Units]
Profile Velocity (nominal velocity)	0x6081 / 0x00 [r/min]			Yes (Velocity) [Units/s]
Profile Acceleration (nominal acceleration)	0x6083 / 0x00 [r/min/s]			Yes (Accel) [Units/s^2]
Profile Deceleration (nominal deceleration)	0x6084 / 0x00 [r/min/s]			Yes (Decel) [Units/s^2]

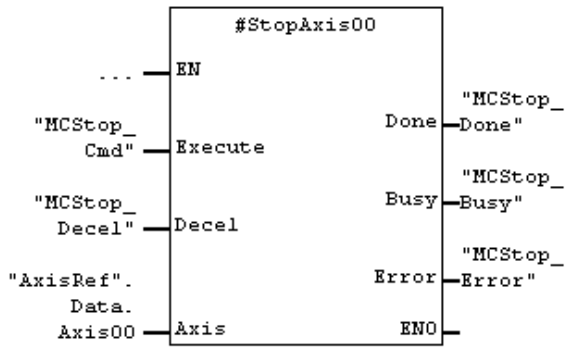
EPOS-Parameter	Object-index resp. subindex	Setting EPOS-Studio resp. SDO-transfer necessary	MC_Stop	MC_MoveAbsolute MC_MoveRelative
Motion Profile Type	0x6086 / 0x00	Yes 0 = linear ramps 1 = sin^2-ramps		

MC_Stop_E2 (FB)

With the block MC_Stop_E2 the axis will be stopped. Stopping is only possible with a power stage enabled axis.

ⓘ	At an 0-1-edge the axis motion is stopped. Axis moves (new requests) will be blocked at activated Stop-Execute (=1) generally. The QuickStop-Function of the axis will be used! It is only one instance of this FB allowed and reasonable.

Name	Variables area	Type	Function
Execute	IN	Bool	1 Stops axis 0 Moves enabled
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Decel	IN	Dint	Stop ramp [Units/s^2]
Done	OUT	Bool	Axis stopped
Busy	OUT	Bool	Function is executed/runs
Error	OUT	Bool	Axis with error

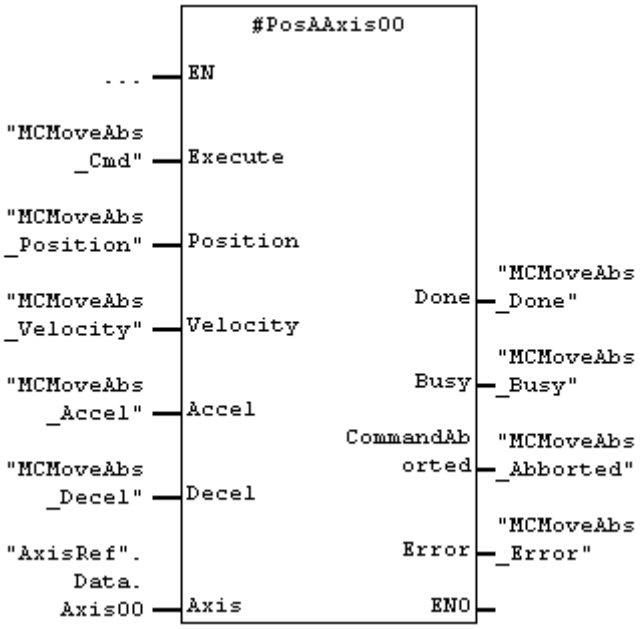


In the STAT-area instanciated MC_Stop_E2 with the instance name StopAxis00.

MC_MoveAbsolute_E2 (FB)

The block MC_MoveAbsolute_E2 will be used for absolute positioning. Reference point of the absolute position is defined by homing reference travel or by determined absolute point of origin (mathematical zero-point).

Name	Variables area	Type	Function
Execute	IN	Bool	0-1-edge starts the move
Position	IN	Real	Absolute position in user units, e.g. „mm“
Velocity	IN	Real	Positioning velocity in user units, e.g. „mm/s“
Accel	IN	Dint	Acceleration in user units, e.g. „mm/s^2“
Decel	IN	Dint	Deceleration in user units, e.g. „mm/s^2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Done	OUT	Bool	Axis has reached target position
Busy	OUT	Bool	Function is executed/runs
CommandAbborted	OUT	Bool	Command was cancelled by new positioning, jogging disabled motor, stopping, etc.
Error	OUT	Bool	Axis with error



In the STAT-area instanciated MC_MoveAbsolute_E2 with the instance name PosAAxis00.

MC_MoveRelative_E2 (FB)

The block MC_MoveRelative_E2 will be used for relative positioning (for a distance). Reference point for the distance is the actual target position. This kind of positioning is referred as chain positioning.

Name	Variables area	Type	Function
Execute	IN	Bool	0-1-edge startes the move
Distance	IN	Real	Distance in user units e.g. „mm“
Velocity	IN	Real	Positioning velocity in user units e.g. „mm/s“
Accel	IN	Dint	Acceleration in user units, e.g. „mm/s^2“
Decel	IN	Dint	Deceleration in user units, e.g. „mm/s^2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Done	OUT	Bool	Axis has reached target postion
Busy	OUT	Bool	Function is executed/runs
CommandAbborted	OUT	Bool	Command was cancelled by new positioning, jogging disabled motor, stopping, etc.
Error	OUT	Bool	Axis with error

Function blocks in the profile Velocity Mode

The blocks MC_Stop_E2, MC_MoveVelocity_E2 and MC_Jog_E2 will be processed in the profile „Velocity Mode“. Beside the assigns on the FB-parameters following CANopen-parameters are to assign via CANopen-SDO or easier via the EPOS-Studio.

EPOS-Parameter	Object-index resp. subindex	Setting EPOS-Studio resp. SDO-transfer necessary	MC_Stop	MC_MoveVelocity MC_Jog
Position Window (positioning window for DONE-message)	0x606D / 0x00 [r/min]	Yes		
Position Window Time (time in the positioning window for DONE-message)	0x606E / 0x00 [ms]	Yes		
Maximal Profile Velocity	0x607F / 0x00 [r/min]	Yes		
QuickStop Deceleration	0x6085 / 0x00 [r/min/s]		Yes (Decel) [Units/s^2]	
Max Acceleration / Deceleration	0x60C5 / 0x00 [r/min/s]	Yes		

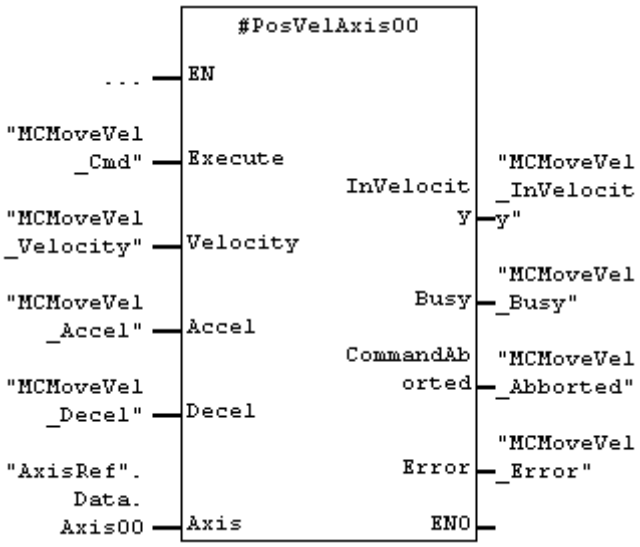
EPOS-Parameter	Object-index resp. subindex	Setting EPOS-Studio resp. SDO-transfer necessary	MC_Stop	MC_MoveVelocity MC_Jog
Target Velocity ((nominal velocity)	0x60FF / 0x00 [r/min]			Yes (Velocity) [Units/s]
Profile Acceleration (nominal acceleration)	0x6083 / 0x00 [r/min/s]			Yes (Accel) [Units/s^2]
Profile Deceleration (nominal deceleration)	0x6084 / 0x00 [r/min/s]			Yes (Decel) [Units/s^2]
Motion Profile Type	0x6086 / 0x00	Yes 0 = linear ramps 1 = sin^2-ramps		

MC_MoveVelocity_E2 (FB)

The block MC_MoveVelocity_E2 will be used for endless moves.

ⓘ	The move MUST be stopped with the block MC_Stop_E2. Will be assigned the velocity 0.0 units/s on the block MC_MoveVelocity, the drives moves with the profile- (nominal-) velocity 0.0 units/s. It means, the assigned profile-(nominate) values are still active. If the velocity shell be changed „flying“, the velocity can be changed by an additional instance of MC_MoveVelocity or the Execute of the „one“ instance can be re-triggered (another 0-1-edge).

Name	Variables area	Type	Function
Execute	IN	Bool	0-1-edge startes the move
Velocity	IN	Real	Positioning velocity in user units e.g. „mm/s“
Accel	IN	Dint	Acceleration in user units, e.g. „mm/s^2“
Decel	IN	Dint	Deceleration in user units, e.g. „mm/s^2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
InVelocity	OUT	Bool	Axis has reached profile (target-) postion (distance driven)
Busy	OUT	Bool	Function is executed/runs
CommandAbborted	OUT	Bool	Command was cancelled by new positioning, jogging disabled motor, stopping, etc.
Error	OUT	Bool	Axis with error



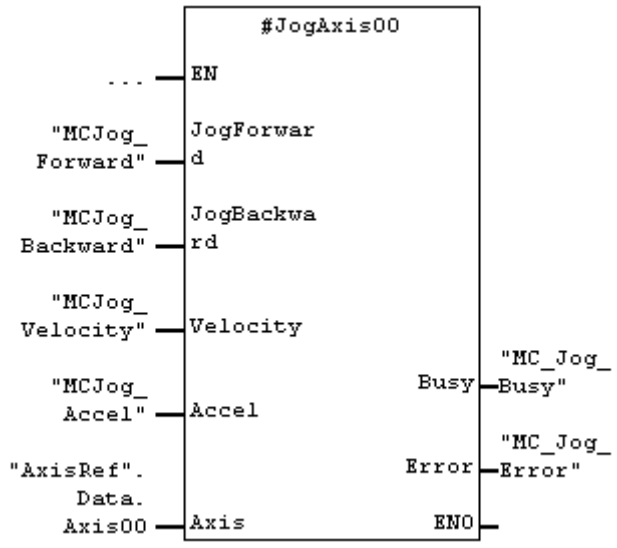
In the STAT-area instanciated MC_MoveVelocity_E2 with the instance name PosVelAxis00.

MC_Jog_E2 (FB)

The block MC_Jog_E2 will be used to move the axis „manually“ (also named „inching“). Both directions are possible. The FB wors like the MC_MoveVelocity, but ist can be moved directional.

ⓘ	If the Jog-process will be finished (JogForward and JogBackward both FALSE), than internal will be triggered QuickStop, where in difference wobei to the „normal“ stopping via MC_Stop_E2 the Accel-ramp from FB-input is used, so can be stopped more „soft“ Note, that the Accel-value is valid for acceleration as well as for deceleration.

Name	Variables area	Type	Function
JogForward	IN	Bool	0-1-edge startes jogging clockwise, 1-0-edge stops the axis
JogBackward	IN	Bool	0-1-edge startes jogging counter-clockwise, 1-0-edge stops the axis
Velocity	IN	Real	Manual velocity in user units, e.g. „mm/s“
Accel	IN	Dint	Acceleration / deceleration in user units, e.g. „mm/s2“
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Busy	OUT	Bool	FB-Function is executed/runs
Error	OUT	Bool	Axis with error



In the STAT-area instanciated MC_Jog_E2 with the instance name JogAxis00.

Function blocks in the Homing Mode

The blocks MC_Home_E2 will be executed in the homing mode. Beside the assign on the FB-parameters following CANopen-paramters are to assign via CANopen-SDO or easier via the EPOS-Studio.

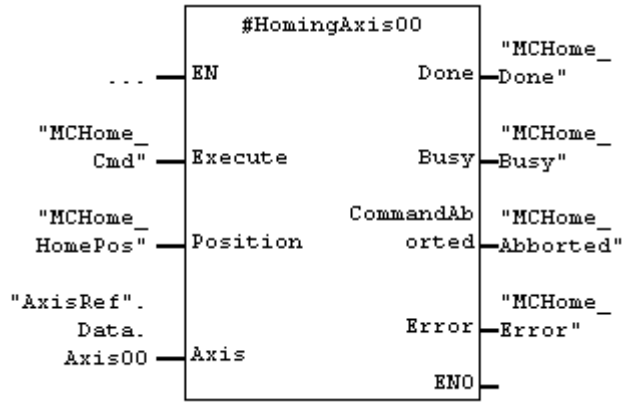
EPOS-Parameter	Object-index resp. subindex	Settings EPOS-Studio resp. SDO-transfer neccessary	MC_Home
Homing Method (Method of referenciation by position switches, mechanical arresters, set position,...)	0x6098 / 0x00	Yes	
Homing Speeds (search velocity at the referencation)	0x6099 / 0x00 [U/min]	Yes	
Homing Acceleration (ramps at the referencation)	0x609A / 0x00 [U/min/s]	Yes	
Home Offset (after the referenc-ation, this course will be driven too, than the value of the parameter Home position will be set as new actual position.	0x607C / 0x00 [Inkremente]	Yes	
Current Threshold for homing modes -1 to -4 (If it should be driven against an arrester while referencing, current limit)	0x2080 [mA]	Yes	
Home position (Set position after sucessfully Homing and driving the course in the Home Offset)	0x2081 / 0x00 [increments]		Yes (Set position after finished homing-procedure) [Units]
Motion Profile Type	0x6086 / 0x00	Yes 0 = linear ramps 1 = $\sin^2$-ramps	

MC_Home_E2 (FB)

Ther block MC_Home_C3 will be used to define the mathematical reference- (zero-) point of the axis. The reference kinds (Modi) are to be find in the EPOS2-manual. The modi „35“ and „-3“ were tested in the labaratories.

ⓘ	It is only one instance of this FB allowed and useful.

Name	Variables area	Type	Function
Execute	IN	Bool	0-1-edge startes the reference move
Position	IN	REAL	Set position (new target position) in user units, eg.g. „mm“, the value of the parameter 0x2081/0x00 will be described
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
Done	OUT	Bool	Reference move (incl. Driving the distance in the parameter 0x607C/0x00 Home Offset) finished, Set position was set, the mathematical zero point and if so, the Software-End-Limits are valid
Busy	OUT	Bool	Function is executed/runs
CommandAbborted	OUT	Bool	Command was cancelled by new positioning, jogging disabled motor, stopping, etc.
Error	OUT	Bool	Axis with error or referencation (Homing) finished with error



In the STAT-area instanciated MC_Home_E2 with the instance name HomingAxis00.

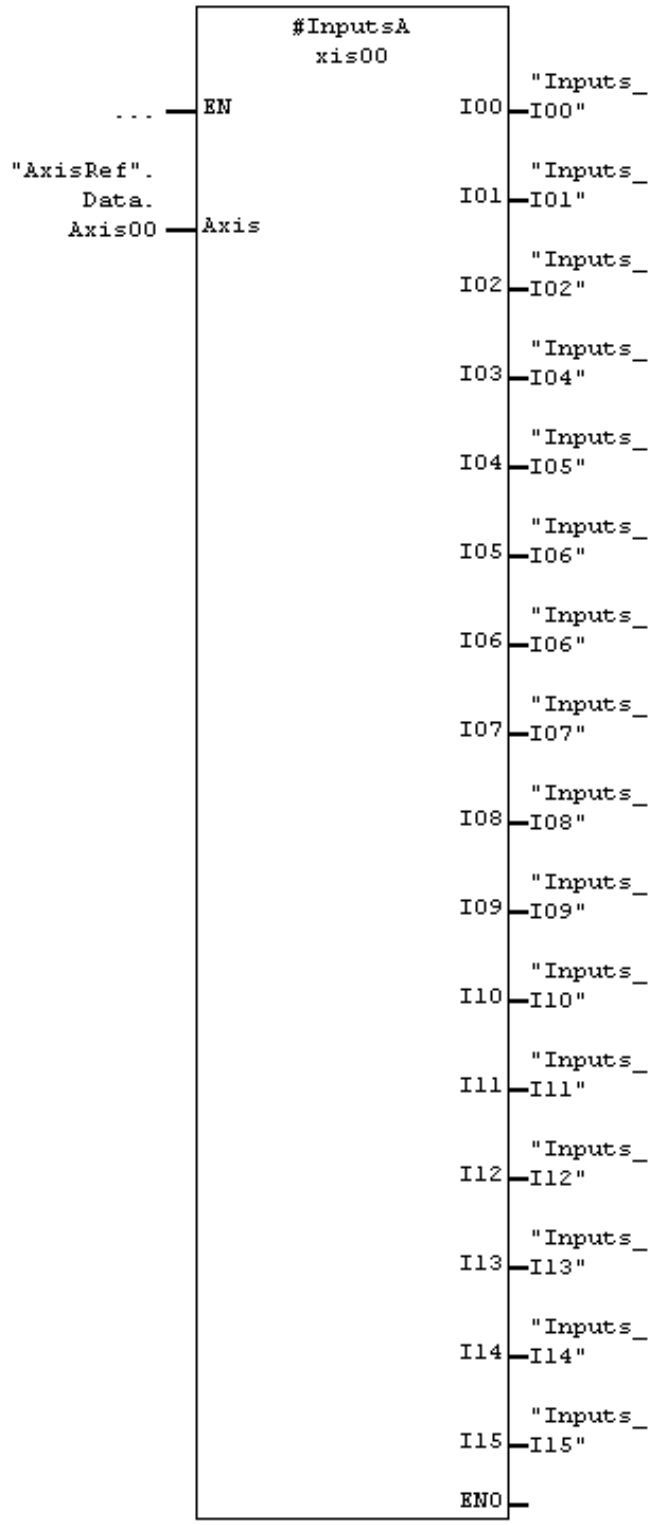
Special-FB's

E2_Input (FB)

The block E2_Input is a special version for the EPOS2-axis (Note the different versions of the EPOS2-family.).

Name	Variables area	Type	Function
Axis	IN_OUT	AxisRefE2Type (UDT)	Axis refererence (axis pointer)
I00 bis I15	OUT	Bool	Inpots 0 to 15 as state

In the STAT-area instanciated E2_Input with the instance name InAxis00.



InDataE2Type (UDT)

This data type is to instanciate while using in a data block with a name, e.g.. Axis00. Exactly 1 instance per axis will be needed. The instance data correlate to the T-PDO-data of the EPOS2-axis.

ⓘ	The best solution is to set up all instances (of the different axis´) of InDataE2Type in a separate DB (e.g. „TPDODATA“).

```
wStatusWord     :   WORD;       // TPDO1, async, 0x6041 + 0x00, Status word
wDigInWord      :   WORD;       // TPDO1, async, 0x2071 + 0x01, Input state
byActModeOfOp   :   BYTE;       // TPDO1, async, 0x6061 + 0x00, Actual mode of operation
diActPosition   :   DINT;       // TPDO2, async, 0x6064 + 0x00, Actual position [pq Enc.]
diActVelocity   :   DINT;       // TPDO2, async, 0x606C + 0x00, Actual velocity [rev/min]
```

OutDataE2Type (UDT)

This data type is to instanciate while using in a data block with a name, e.g.. Axis00. Exactly 1 instance per axis will be needed. The instance data correlate to the R-PDO-data of the EPOS2-axis.

ⓘ	The best solution is to set up all instances (of the different axis´) of OutDataE2Type in a separate DB (e.g. „RPDODATA“).

```
wControlWord        :   WORD;       // RPDO1, async, 0x6040 + 0x00, Control word
diTarget            :   DINT;       // RPDO1, async, 0x607a + 0x00, Target [pq Enc.]
byModeOfOp          :   BYTE;       // RPDO1, async, 0x6060 + 0x00, Mode of operation
diProfVelocity      :   DINT;       // RPDO2, async, 0x6081 + 0x00, Profile velocity [rev/min]
diHomePosition      :   DINT;       // RPDO2, async, 0x2081 + 0x00, Home position [pq Enc.]
diProfAccel         :   DINT;       // RPDO3, async, 0x6083 + 0x00, Prof. acceleration [rev/min/s]
diProfDecel         :   DINT;       // RPDO3, async, 0x6084 + 0x00, Prof. deceleration [rev/min/s]
diQuickStopDecel    :   DINT;       // RPDO4, async, 0x6085 + 0x00, Prof. deceleration [rev/min/s]
diSpeed             :   DINT;       // RPDO4, async, 0x60FF + 0x00, Prof. deceleration [rev/min]
```

AxisRefE2Type

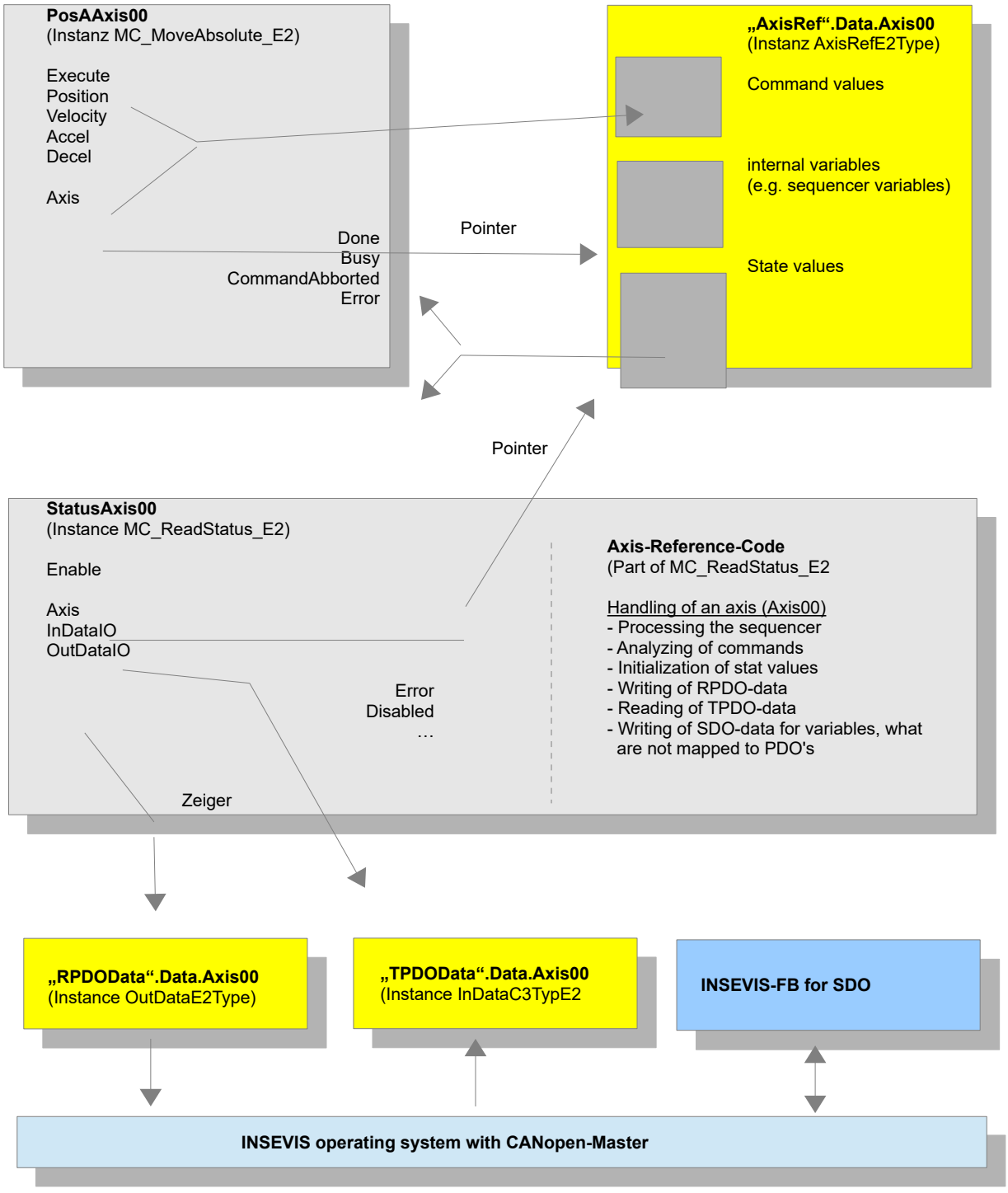
This data type will be used internally as axis working data of the axis. Because the instanciated variables were handled over as IN_OUT, the efford for copying is low.

The block MC_ReadStatus_E2 uses the variables, defined by the axis reference for processing the sequencers.

ⓘ	The best solution is to set up all instances (of the different axis´) of AxisRefE2Type in a separate DB (e.g. „AXISREF“)

Sample of a MC-block-instance

Following figure shows the use and the data flow of a MC-block for one axis with the name Axis00.

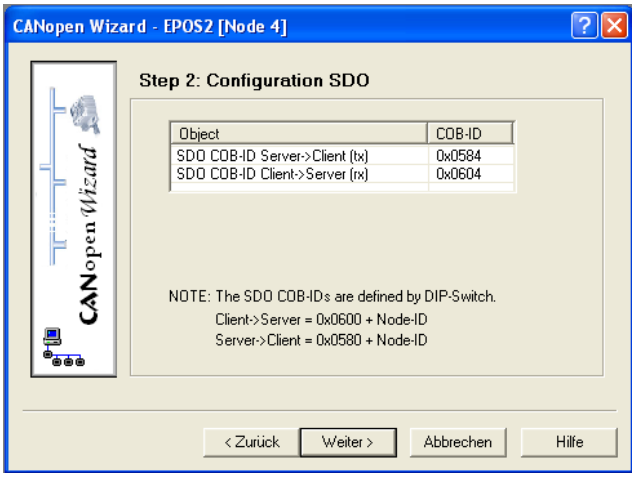
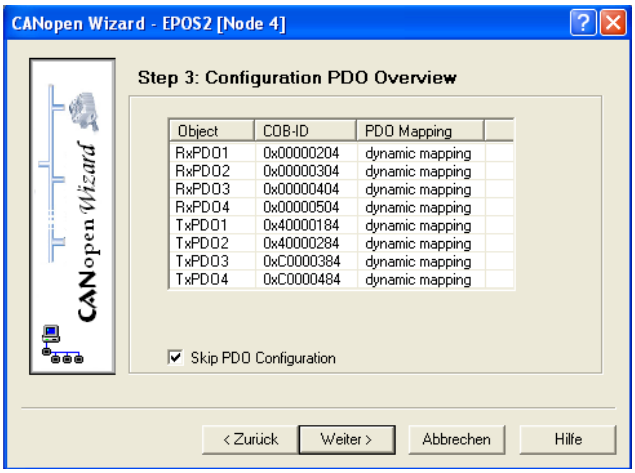


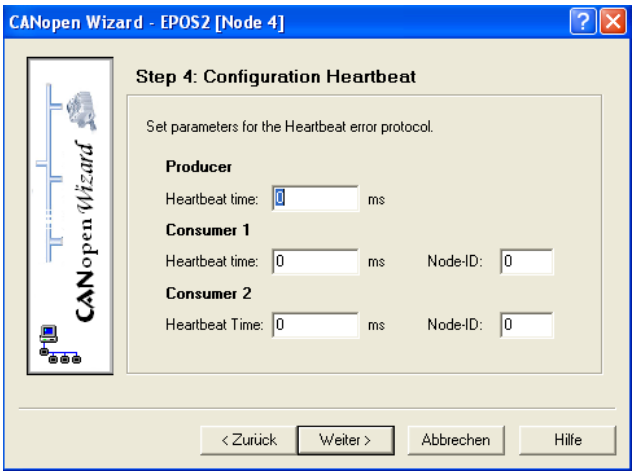
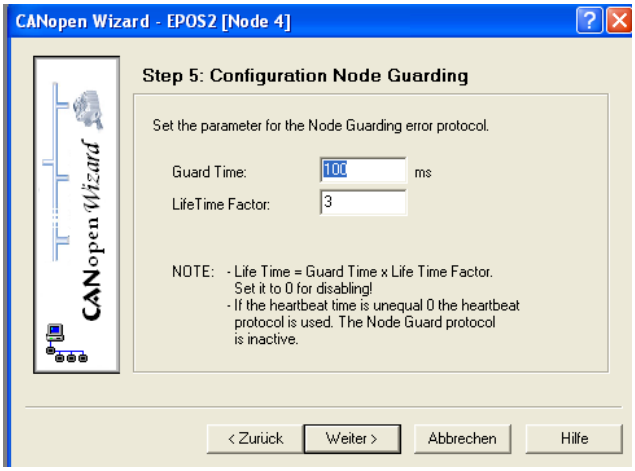
CANopen-configuration with the EPOS-Studio

Gererally there can not payed attention for the common configuration of a EPOS2-servo drive. Important for the CANopen-part is only, that actual configuration should be tested via the CANopen-Wizard (after configuration with the ConfigStage and start of the INSEVIS-PLC), because the setting of the CANopen-Slaves are carried out by the INSEVIS-PLC thereselfe.

The CANopen-Node-ID is to be set on the referring jumper (CAN-ID) too. The CAN-Bitrate (0x2001) is default assigned to the value „9“ (automatic detection). It is recommended to keep this value unchanged for an easier parameterization.

ⓘ	Use the „Object Dictionary“, to test or to change actual values.

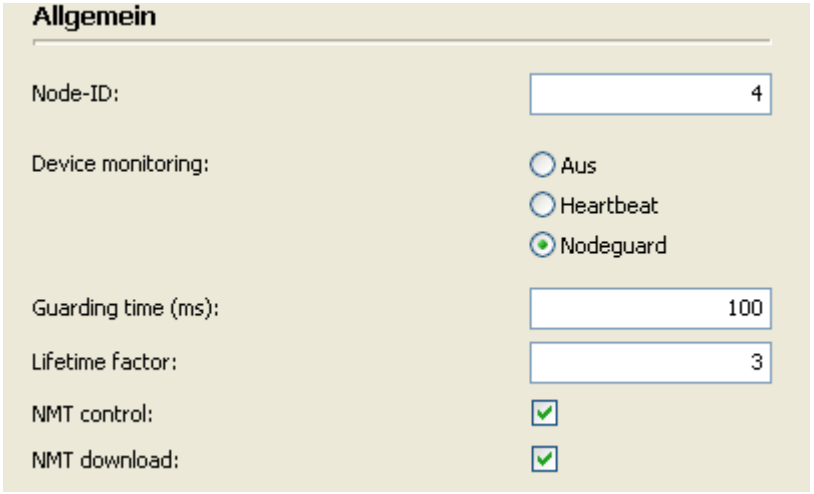
	Check of the COB-ID's for the SDO's, (normally no changes needed).
	The PDO-configuration can be skipped completey.

	A heartbeat-monitoring is not used.
	Nodeguarding will be configurated and activated by the INSEVIS-PLC.

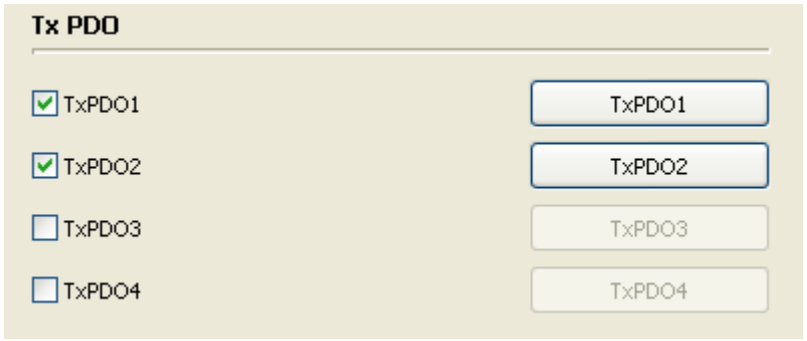
Slave-configuration with the ConfigStage

With the ConfigStage-software will amongst others configurated the CANopen-Master and each CANopen-Slave. Also the connection from PLC-data (e.g. DataBlock and offset in the DataBlock) to the CANopen-data (R-PDO's, T-PDO's) will bedefined.

The axis can be taken over as type into the library of the ConfigStage!

<table>
<tr>
<td> 
</td>
<td><u>Definition of Node-ID and guardings</u>

EPOS2 supports node guarding

CANopen-settings (like COB-ID's) to load to EPOS2</td>
</tr>
<tr>
<td> 
</td>
<td><u>TPDO (EPOS2→ CANopen-Master)</u>
2 T-PDO's are necessary for the receive direction.
activate the download of the communication parameter and mapping

Response characteristics TPDO1
Typ: 254, no blocking time

Response characteristics TPDO2
Typ: 254, define a blocking time of e.g. 100ms !</td>
</tr>
<tr>
<td> 
</td>
<td><u>RPDO (CANopen-Master → EPOS2)</u>

4 R-PDO's are necessary for the send direction.
activate the download of the communication parameter and mapping

Response characteristics RPDO1
Typ: 254, no blocking time

Response characteristics RPDO2
Typ: 254, no blocking time</td>
</tr>
</table>

	Response characteristics RPDO3 Typ: 254, no blocking time Response characteristics RPDO3 Typ: 254, no blocking time
	additional configuration via SDO No additional SDOs are neccessary. (If you want, you can transfere here CANopen-data, what otherwise must be transferred by the EPOS-Studio.)

Mapping T-PDO1

Offset in data area (e.g. data block) of an instance from type „InDataE2Type“: **0** (Byte-Offset)

Number	Index	Subindex	Size	Explanation
1	0x6041	0x00	16 Bit/Word	State word DS402
2	0x2071	0x01	16 Bit /Word	Input word of the digital inputs
3	0x6061	0x00	16 Bit/Word	Actual operation mode DS402

Mapping T-PDO2

Offset in data area (e.g. data block) of an instance from type „InDataE2Type“: **6** (Byte-Offset)

Number	Index	Subindex	Size	Explanation
1	0x6064	0x00	32 Bit/DWord	Actual position [Post quadrature encoder]
2	0x606C	0x00	32 Bit/DWord	Actual velocity [U/min]

Mapping R-PDO1

Offset in data area (e.g. data block) of an instance from type „OutDataE2Type“: **0** (Byte-Offset)

Number	Index	Subindex	Size	Explanation
1	0x6040	0x00	16 Bit/Word	Control word DS402
2	0x607A	0x00	32Bit/DWord	Target [Post quadrature encoder]
3	0x6060	0x00	16 Bit/Word	Operation mode DS402

Mapping R-PDO2

Offset in data area (e.g. data block) of an instance from type „OutDataE2Type“: **8** (Byte-Offset)

Number	Index	Subindex	Size	Explanation
1	0x6081	0x00	32 Bit/DWord	Profile velocity [r/min]
2	0x2081	0x00	32 Bit/DWord	Home-position [Post quadrature encoder]

Mapping R-PDO3

Offset in data area (e.g. data block) of an instance from type „OutDataE2Type“: **16** (Byte-Offset)

Number	Index	Subindex	Size	Explanation
1	0x6083	0	32 Bit/DWord	Profile acceleration [r/min/s]
2	0x6084	0	32 Bit/DWord	Profile deceleration [r/min/s]

Mapping R-PDO4

Offset in data area (e.g. data block) of an instance from type „OutDataE2Type“: **24** (Byte-Offset)

Number	Index	Subindex	Size	Explanation
1	0x6085	0	32 Bit/DWord	Quick-stop-ramp [r/min/s]
2	0x60FF	0	32 Bit/DWord	Velocity in the profile Velocity Mode [r/min]

Additional SDO-transfers after PDO-mapping

Number	Index	Subindex	Size	Value	Explanation

S7-Sample-program

The sample project consists of an S7-program, what demonstrates the application of the MC-blocks.

INSEVIS

INSEVIS Vertriebs GmbH

Am Weichselgarten 7
D - 91058 Erlangen

Fon: +49(0)9131-691-440
Fax: +49(0)9131-691-444
Web: www.insevis.de
E-Mail: info@insevis.de