



Funktionsbausteine für PSx-3__ mit EtherNet/IP-Schnittstelle

halstrup-walcher GmbH

Stegener Straße 10
D-79199 Kirchzarten

Phone: +49 (0) 76 61/39 63-0
Fax: +49 (0) 76 61/39 63-99

E-Mail: info@halstrup-walcher.de
Internet: www.halstrup-walcher.de

Inhaltsverzeichnis

1	Sicherheitshinweise	4
1.1	Bestimmungsgemäße Verwendung	4
1.2	Symbolerklärung	4
2	Datenstruktur DRIVE_DATA	5
3	Fehlerbeschreibung (Error ID).....	8
4	Beschreibung der Funktionsbausteine.....	10
4.1	Gemeinsamkeiten aller Funktionsbausteine.....	10
4.2	MC_Move	12
4.3	MC_Error	15
4.4	MC_ReadParameter	16
4.5	MC_WriteParameter	18
4.6	MC_Parametrization	20
4.7	MC_PositionParametrization.....	23
5	Hinzufügen von Antrieben zu einem Projekt.....	28
5.1	Betreffende EDS-Datei mit Hilfe von RSNetWorx registrieren	28
5.2	Betreffende EDS-Datei mit Hilfe von Logix Designer registrieren	30
5.3	Einen Antrieb hinzufügen	34
6	Hinzufügen von Funktionsblöcken zu einem Projekt	38
6.1	Einen Funktionsblock importieren	38
6.2	Darstellung der Antriebe und FBs im Controller Organizer	41
7	Hinzufügen von FBs und Controller Tags in ein Programm	42

Bedeutung der Betriebsanleitung

Diese Betriebsanleitung erläutert die Funktionsbausteine für die Positioniersysteme PSx 3__ EI (mit EtherNet/IP-Schnittstelle).

Von diesen Geräten können für Personen und Sachwerte Gefahren durch nicht bestimmungsgemäße Verwendung und durch Fehlbedienung ausgehen. Deshalb muss jede Person, die mit der Handhabung der Geräte betraut ist, eingewiesen sein und die Gefahren kennen. Die Betriebsanleitung und insbesondere die darin gegebenen Sicherheitshinweise müssen sorgfältig beachtet werden. **Wenden Sie sich unbedingt an den Hersteller, wenn Sie Teile davon nicht verstehen.**

Der Hersteller behält sich das Recht vor, diese Funktionsbausteine weiterzuentwickeln, ohne dies in jedem Einzelfall zu dokumentieren. Über die Aktualität dieser Betriebsanleitung gibt Ihnen Ihr Hersteller gerne Auskunft.

© 2015

Das Urheberrecht an dieser Betriebsanleitung verbleibt beim Hersteller. Sie darf weder ganz noch in Teilen vervielfältigt oder Dritten zugänglich gemacht werden.

1 Sicherheitshinweise

1.1 Bestimmungsgemäße Verwendung

Die Positioniersysteme PSx-3__-EI eignen sich besonders zur automatischen Einstellung von Werkzeugen, Anschlägen oder Spindeln bei Holzverarbeitungsmaschinen, Verpackungsmaschinen, Druckmaschinen, Abfüllanlagen und bei Sondermaschinen.

Die PSx-3__-EI sind nicht als eigenständige Geräte zu betreiben, sondern dienen ausschließlich zum Anbau an eine Maschine.

1.2 Symbolerklärung

In dieser Betriebsanleitung wird mit folgenden Hervorhebungen auf die darauf folgend beschriebenen Gefahren bei der Handhabung der Anlage hingewiesen:



WARNUNG!

Sie werden auf eine Gefährdung hingewiesen, die zu Körperverletzungen bis hin zum Tod führen kann, wenn Sie die gegebenen Anweisungen missachten.



ACHTUNG!

Sie werden auf eine Gefährdung hingewiesen, die zu einem erheblichen Sachschaden führen kann, wenn Sie die gegebenen Anweisungen missachten.



INFORMATION!

Sie erhalten wichtige Informationen zum sachgemäßen Betrieb.

2 Datenstruktur DRIVE_DATA

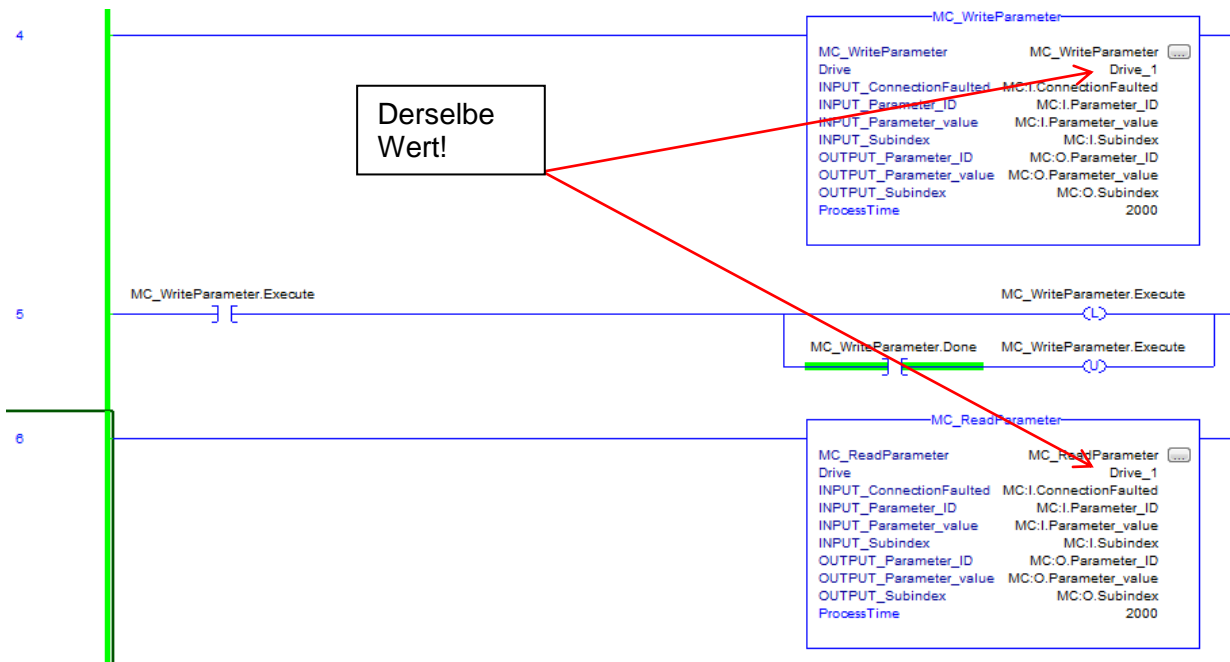
Für jeden Antrieb gibt es eine Datenstruktur, in der einige Daten dieses Antriebs abgelegt sind. Dies sind der Antriebsstatus, der Name des Antriebs (optional) sowie eine Beschreibung der Funktion des Antriebs (optional). Für jeden Antrieb wird eine Instanz dieser Struktur benötigt. Diese Instanz muss jedem FB übergeben werden, der auf den betr. Antrieb wirkt. Pro Antrieb stehen sechs FBs zur Verfügung. Mit Hilfe des Antriebsstatus wird sichergestellt, dass nicht gleichzeitig zwei Zugriffe aus unterschiedlichen FBs auf den Parameterkanal durchgeführt werden können.

Parametername	Datentyp	geschrieben von	Beschreibung
Name	STRING[16]	Benutzer (optional)	Name der Achse
Beschreibung	STRING[32]	Benutzer (optional)	Beschreibung (z.B. Funktion, Aufgabe dieser Achse)
State	DINT	Funktionsbausteine	Actual state

Diese Bibliothek beinhaltet Funktionen zum Ausführen der folgenden Aufgaben (anwendbar auf jeden im Projekt beinhalteten Antrieb):

- Ein Fahrauftrag kann mit Hilfe des FBs „MC_Move“ erteilt werden.
- Fehlermeldungen können mit Hilfe des FBs „MC_Error“ abgerufen werden.
- Einzelne Parameter können mit dem FB „MC_ReadParameter“ gelesen werden.
- Einzelne Parameter können mit dem FB „MC_WriteParameter“ geschrieben werden.
- Mehrere Parameter können gleichzeitig mit dem FB „MC_Parametrization“ geschrieben werden.
- Alle Parameter, die für die Berechnung der Istposition relevant sind, können gleichzeitig mit dem FB „MC_PosParametrization“ geschrieben werden.

Das untenstehende Beispiel beinhaltet zwei unterschiedliche FB-Aufrufe, die für denselben Antrieb gelten sollen. Der Parameter „Drive“ muss daher bei beiden FB-Aufrufen denselben Wert haben (in diesem Falle „Drive_1“).



Abhängig davon, welche FBs bei einem bestimmten Antrieb gerade aktiv sind, hat der Status dieses Antriebs den folgenden Wert:

- MC_Move → bit 0 von „state“ (läuft: bit 0 gesetzt; läuft nicht: bit 0 zurückgesetzt)
- MC_Error → kein Einfluss auf „state“
- MC_ReadParameter → state = 30
- MC_WriteParameter → state = 40
- MC_Parametrization → state = 50
- MC_PosParametrization → state = 60

Wenn ein FB bei einem bestimmten Antrieb aktiv wird, setzt er die Variable „state“ für diesen Antrieb. Bei jedem Antrieb zeigt die Variable „state“, welche FBs aktuell bei diesem Antrieb aktiv sind.

Beispiele:

- state = 0 → Alle FBs sind inaktiv.
- state = 30 → Nur FB „MC_ReadParameter“ läuft.
- state = 41 → Die FBs „MC_WriteParameter“ und „MC_Move“ laufen.

Die folgenden Regeln gelten für das gleichzeitige Aktivieren mehrerer FBs bei demselben Antrieb:

- „MC_Error“ kann gleichzeitig mit allen anderen FBs aktiv sein.
- „MC_Move“ kann nur gestartet werden, wenn „MC_Parametrization“ und „MC_PositionParametrization“ inaktiv sind. („MC_ReadParameter“ oder „MC_WriteParameter“ können gleichzeitig mit „MC_Move“ aktiv sein.)
- „MC_ReadParameter“ kann nur gestartet werden, wenn „MC_WriteParameter“, „MC_Parametrization“ und „MC_PositionParametrization“ inaktiv sind. („MC_Move“ kann aktiv sein.)

- „MC_WriteParameter“ kann nur gestartet werden, wenn „MC_ReadParameter“, „MC_Parametrization“ und „MC_PositionParametrization“ inaktiv sind. („MC_Move“ kann aktiv sein.)
- „MC_Parametrization“ kann nur gestartet werden, wenn „MC_Move“, „MC_ReadParameter“, „MC_WriteParameter“ und „MC_PositionParametrization“ inaktiv sind.
- „MC_PositionParametrization“ kann nur gestartet werden, wenn „MC_Move“, „MC_ReadParameter“, „MC_WriteParameter“ und „MC_Parametrization“ inaktiv sind.

Beispiel:

Im Projekt befinden sich drei Antriebe. Jeder Antrieb soll über einen FB MC_Move angesteuert werden, außerdem soll mit MC_Error der Status jedes Antriebs ermittelt werden können und pro Antrieb sollen beliebige Schreib- und Lesezugriffe ausgeführt werden können

Dazu sind insgesamt drei globale Variablen des Typs DRIVE_DATA notwendig, diese müssen bei den Controller Tags generiert werden:

- Drive_1
- Drive_2
- Drive_3

Für die Ausführung der benötigten Funktionen wird pro Antrieb auch je eine Instanz der FBs MC_Move, MC_Error, MC_ReadParameter und MC_WriteParameter benötigt:

- MC_Move_1, MC_Error_1, MC_Read_1, MC_Write_1:
VAR_IN_OUT „Drive“ → „Drive_1“ eintragen
- MC_Move_2, MC_Error_2, MC_Read_2, MC_Write_2:
VAR_IN_OUT „Drive“ → „Drive_2“ eintragen
- MC_Move_3, MC_Error_3, MC_Read_3, MC_Write_3:
VAR_IN_OUT „Drive“ → „Drive_3“ eintragen

3 Fehlerbeschreibung (Error ID)

Nachfolgend die Fehlercodes, die von den Funktionsbausteinen ausgegeben werden:

ErrorID (hex)	Beschreibung
16xF000 (mask)	FB
16#1xxx	Error in MC_Move
16#2xxx	Error in MC_Error
16#3xxx	Error in MC_ReadParameter
16#4xxx	Error in MC_WriteParameter
16#5xxx	Error in MC_Parametrization
16#6xxx	Error in MC_PositionParametrization
16#0F00 (mask)	Internal FB and PD errors
16#x1xx	Error in state machine or other AOI internal error
16#x6xx	Unallowed input data change
16#x7xx	Connection Faulted
16#00F0 (mask)	Parameter errors
16#xx1x	Parameter: communication timeout (1000 ms)
16#xx2x	Parameter: invalid parameter number
16#xx3x	Parameter: value is read only
16#xx4x	Parameter: lower or upper limit exceeded
16#xx5x	Parameter: faulty sub-index
16#xx6x	Parameter: not an array
16#xx7x	Parameter: incorrect data type
16#xx8x	Parameter: setting not allowed (resetting only)
16#xx9x	Parameter: request cannot be processed due to operating state
16#xxAx	Other error
16#000F (mask)	Drive errors
16#xxx1	Drag error
16#xxx2	Under- or overvoltage motor supply
16#xxx3	Positioning run aborted
16#xxx4	Temperature exceeded
16#xxx5	Absolute measuring system error
16#xxx6	Block or overcurrent error
16#xxx7	Manual displacement
16#xxx8	Incorrect target value
16#xxx9	Under- or overvoltage during run
16#xxxA	Lower position limit exceeded
16#xxxB	Upper position limit exceeded

Die Fehler „Drive errors“ sind eine Abbildung der Fehlerbits im Statuswort des PSx.

Beispiele:

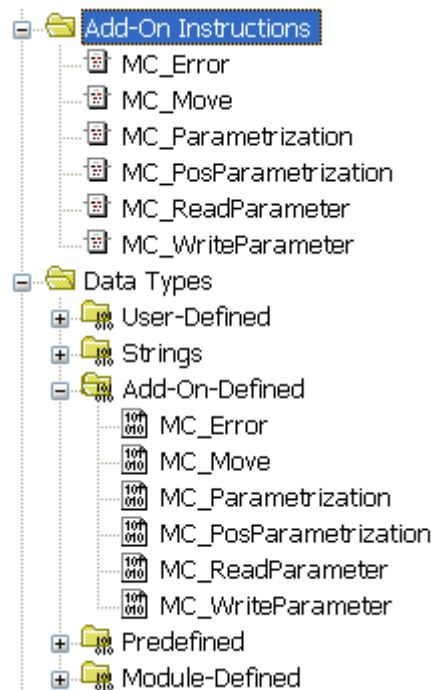
- Fahrauftrag (MC_Move) mit falschem Sollwert → ErrorID = 16#1008

- Parameter schreiben (MC_WriteParameter) mit ungültiger Parameternummer → ErrorID = 16#4020

4 Beschreibung der Funktionsbausteine

Zunächst müssen die Bausteine in ein eigenes RSLogix- oder Studio 5000-Projekt eingebunden werden. Dies geschieht im Controller Organizer mit Rechtsklick auf „Add-On Instructions“, dann „Import Add-On Instruction“ und die gewünschten Funktionsbausteine einzeln importieren.

Im Ergebnis präsentieren diese sich dann folgendermaßen im Controller Organizer:



4.1 Gemeinsamkeiten aller Funktionsbausteine

Die FBs nutzen jeweils einen Teil der folgenden Variablen, die alle vom Typ VAR_IN_OUT sind. Diese Ein- und Ausgänge müssen in jedem Fall beschaltet werden, ansonsten gibt es beim Download Fehlermeldungen.

Drive

Referenz auf den gewünschten Antrieb (s. auch Kap. 2)

- Typ: DRIVE_DATA
- Art: VAR_IN_OUT

ErrorDescription

Fehlerbeschreibung

- Typ: STRING
- Default value: „
- mit beliebiger String-Variable aus den Controller Tags verbinden

INPUT_ConnectionFaulted

Zeigt an, ob die Eingangsdaten des betr. Antriebs gültig sind

- Typ: BOOL
- connect with I.ConnectionFaulted of the corresp. drive

INPUT_Status_Word

Statuswort des betr. Antriebs

- Typ: INT
- mit I.status_word des betr. Antriebs verbinden

INPUT_Actual_Position

Istposition des betr. Antriebs

- Typ: DINT
- mit I.actual_position des betr. Antriebs verbinden

INPUT_Parameter_ID

Parameternummer des Parameterinterface des betr. Antriebs

- Typ: INT
- mit I.Parameter_ID des betr. Antriebs verbinden

INPUT_Parameter_value

Parameterwert des Parameterinterface des betr. Antriebs

- Typ: DINT
- mit I.Parameter_value des betr. Antriebs verbinden

INPUT_Subindex

Array-Subindex des Parameterinterface des betr. Antriebs

- Typ: INT
- mit I.Subindex des betr. Antriebs verbinden

OUTPUT_Control_Word

Steuerwort des betr. Antriebs

- Typ: INT
- mit O.control_word des betr. Antriebs verbinden

OUTPUT_Target_Position

Sollposition des betr. Antriebs

- Typ: DINT
- mit O.target_position des betr. Antriebs verbinden

OUTPUT_Parameter_ID

Parameternummer des Parameterinterface des betr. Antriebs

- Typ: INT
- mit O.Parameter_ID des betr. Antriebs verbinden

OUTPUT_Parameter_value

Parameterwert des Parameterinterface des betr. Antriebs

- Typ: DINT
- mit O.Parameter_value des betr. Antriebs verbinden

OUTPUT_Subindex

Array-Subindex des Parameterinterface des betr. Antriebs

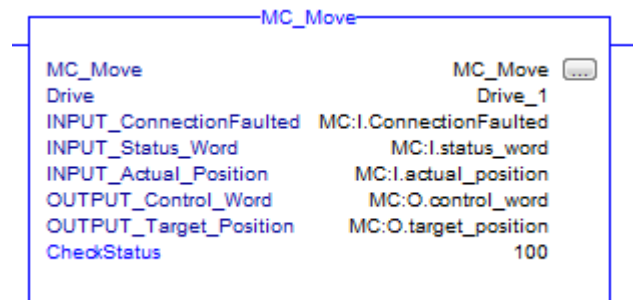
- Typ: INT

- mit O.Subindex des betr. Antriebs verbinden

Diese Variablen vom Typ VAR_IN_OUT sind in den folgenden Beschreibungen der einzelnen FBs nicht mehr gesondert aufgeführt.

4.2 MC_Move

Dieser FB dient der Positionierung des Antriebs.



Release

Freigabe des Antriebs

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Beschreibung:

- Ein Sollwert wird erst angefahren, wenn dieser Eingang gesetzt ist.
- Dieser Eingang wirkt direkt auf das Freigabebit (Bit 4) im Steuerwort. Bleibt der Eingang gesetzt und ist z.B. das Nachregeln im Antrieb aktiv, so regelt der Antrieb automatisch nach.
- Ist der Eingang gesetzt und wird der Sollwert geändert, so fährt der Antrieb diesen sofort an. Eine Flanke ist nicht erforderlich.
- Wird der Eingang während der Fahrt zurückgesetzt, stoppt der Antrieb.

Position

Anzufahrender Sollwert

- Typ: DINT
- Anfangswert: 0
- Art: VAR_INPUT

Beschreibung:

- Wird während einer Fahrt eine neue Sollposition übertragen, wird diese sofort angefahren.
- Ist nach Fahrtende das Release-Bit noch gesetzt und wird der Sollwert geändert, so fährt der Antrieb diesen sofort an.



INFORMATION!

Um den gleichen Sollwert z.B. nach einem Blockieren anzufahren, muss die Freigabe „Release“ zurückgesetzt und erneut gesetzt werden.

ManualRunToLargerValues

Handfahrt zu größeren Werten

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Beschreibung:

- Handfahrt zu größeren Werten bis zum oberen Endschalter.
- Der Eingang „Release“ muss zusätzlich gesetzt sein/werden.



ACHTUNG!

Beim Zurücksetzen des Eingangs „ManualRunToLargerValues“ muss auch der Release-Eingang zurückgesetzt werden, da der Antrieb ansonsten den Sollwert (FB-Eingang „Position“) anfährt.

ManualRunToSmallerValues

Handfahrt zu kleineren Werten

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Beschreibung:

- Handfahrt zu kleineren Werten bis zum unteren Endschalter.
- Der Eingang „Release“ muss zusätzlich gesetzt sein/werden.



ACHTUNG!

Beim Zurücksetzen des Eingangs „ManualRunToSmallerValues“ muss auch der Release-Eingang zurückgesetzt werden, da der Antrieb ansonsten den Sollwert (FB-Eingang „Position“) anfährt.

CheckStatus

Verzögerungszeit beim Abfragen des Statusworts

- Typ: DINT
- Anfangswert: 100
- Art: VAR_INPUT

Beschreibung:

- Logische Verzögerungszeit (in [ms]) für das Abfragen der Statuswort-Bits 6 und 0. Beim Erteilen eines Fahrauftrags wird erst nach Ablauf dieser Verzögerungszeit geprüft, ob der Antrieb fährt. Es soll verhindert werden, dass der FB das Statuswort-Bit 6 („Antrieb läuft“) inaktiv und das Statuswort-Bit 0 („Sollposition ist erreicht“) aktiv sieht, bevor der Antrieb den neuen Fahrauftrag empfangen hat.
- Der Wert 100 ist für die meisten Anwendungen ausreichend. Falls der Wert RPI für die genutzte Verbindung höher als der Wert von „CheckStatus“ sein sollte, besteht allerdings die Gefahr, dass nach Ablauf der Verzögerungszeit der FB den Fahrauftrag als beendet betrachtet, bevor er überhaupt begonnen hat.

Active

Fahrauftrag bzw. Fahrt ist aktiv

- Typ: BOOL
- Art: VAR_OUTPUT

Dieser Ausgang wird gesetzt, wenn:

- die Freigabe („Release“) von 0 auf 1 gesetzt wird
- die Freigabe („Release“) schon vorhanden ist und sich der Sollwert ändert

- das Bit „Antrieb läuft“ im Status des Antriebs gesetzt ist (z.B. beim Nachregeln des Antriebs)

Dieser Ausgang wird zurückgesetzt, wenn:

- am Ende einer Fahrt das Bit „Antrieb läuft“ im Status des Antriebs nicht mehr gesetzt ist
- ein Kommunikationsfehler auftritt

InPosition

Sollposition erreicht

- Typ: BOOL
- Art: VAR_OUTPUT

Dieser Ausgang ist eine Abbildung des Statusbits „Sollposition erreicht“. Falls ein Kommunikationsfehler auftritt, wird er zurückgesetzt.

Actual position

Istwert der Position

- Typ: DINT
- Art: VAR_OUTPUT

Dieser Wert ist eine Abbildung der Istposition. Falls ein Kommunikationsfehler auftritt, wird der Wert auf 0 gesetzt.

Error

Fehler bei der Ausführung des FB oder Fehler im Antrieb

- Typ: BOOL
- Art: VAR_OUTPUT

Das Fehlerbit kann auch gesetzt sein, während der Antrieb fährt (z.B. Schleppfehler).

ErrorID

Fehler-ID

- Typ: INT
- Art: VAR_OUTPUT

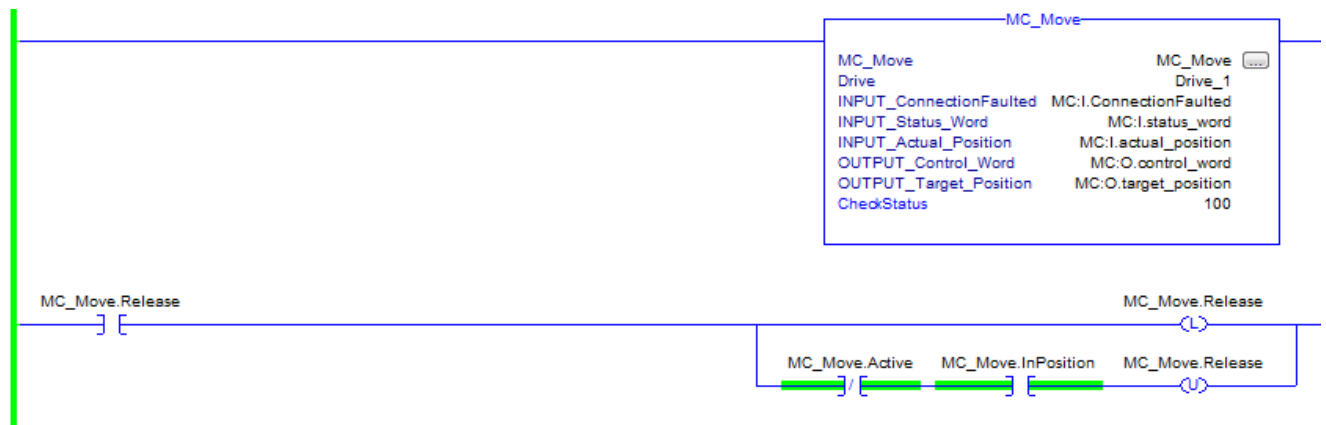
Die ErrorID kann auch gesetzt sein, während der Antrieb fährt (z.B. Schleppfehler). Falls kein Fehler vorliegt, wird 0 ausgegeben.

Falls der Antrieb mehrere Fehler meldet, wird die ErrorID mit der höchsten Priorität ausgegeben. Die Priorität der Ausgabe entspricht der Reihenfolge in der folgenden Tabelle (höchste Prio hat 16#x1xx):

ErrorID	Beschreibung
16#x1xx	FB internal error
16#x2xx	Invalid PD input address
16#x3xx	Invalid PD output address
16#x4xx	Error while reading PD
16#x5xx	Error while writing PD
16#xxx2	Under- or overvoltage motor supply
16#xxx4	Temperature exceeded
16#xxx5	Absolute measuring system error
16#xxx8	Incorrect target value
16#xxx9	Under- or overvoltage during run
16#xxx6	Block or overcurrent error
16#xxx7	Manual displacement
16#xxxA	Lower position limit exceeded

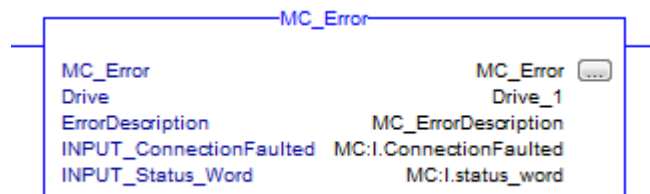
16#xxxB	Upper position limit exceeded
16#xxx3	Positioning run aborted
16#xxx1	Drag error

Einfaches Kontaktplan-Programm mit dem FB „MC_Move“:



4.3 MC_Error

Dieser FB gibt den Status des Antriebs und des FBs als Fehlerbit, Fehler-ID („ErrorID“) und als Text aus.



Enable

Die Ausgänge Error, ErrorID und ErrorDescription werden ständig vom Antrieb aktualisiert, solange Enable gesetzt ist. Wird das Enable zurückgesetzt, so nehmen diese Ausgänge die angegebenen Defaultwerte an.

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Error

Fehler bei der Ausführung des FB oder Fehler im Antrieb

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

ErrorID

Fehler-ID (siehe folgende Tabelle „ErrorID“)

- Typ: INT
- Defaultwert: 0
- Art: VAR_OUTPUT

ErrorDescription

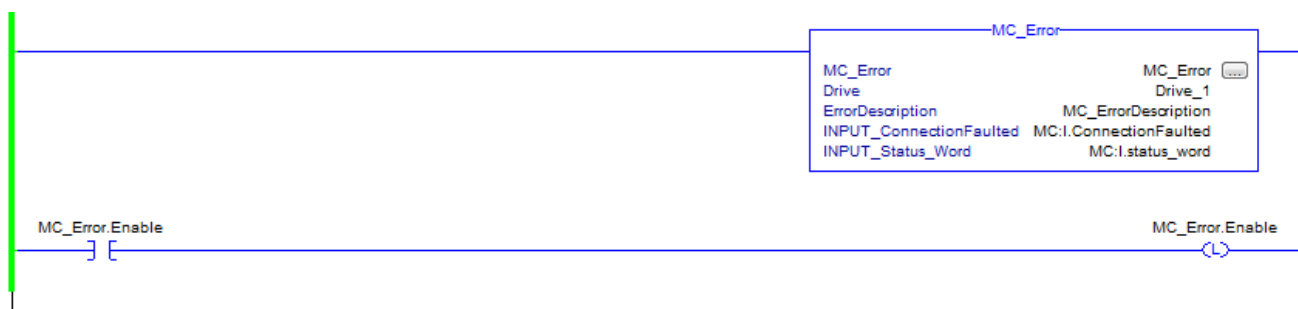
Fehlerbeschreibung als Text

- Typ: STRING
- Defaultwert: „“
- Art: VAR_IN_OUT

Die Priorität der Ausgabe entspricht der Reihenfolge in der folgenden Tabelle (höchste Prio hat 16#x1xx).

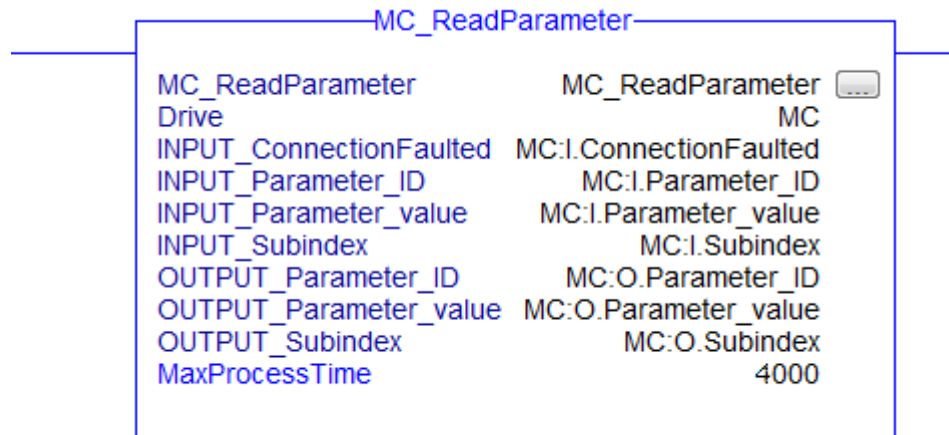
ErrorID	ErrorDescription
16#x1xx	FB internal error
16#x2xx	Invalid PD input address
16#x4xx	Error while reading PD
16#xxx2	Under- or overvoltage motor supply
16#xxx4	Temperature exceeded
16#xxx5	Absolute measuring system error
16#xxx8	Incorrect target value
16#xxx9	Under- or overvoltage during run
16#xxx6	Block or overcurrent error
16#xxx7	Manual displacement
16#xxxA	Lower position limit exceeded
16#xxxB	Upper position limit exceeded
16#xxx3	Positioning run aborted
16#xxx1	Drag error

Einfaches Kontaktplan-Programm mit dem FB „MC_Error“:



4.4 MC_ReadParameter

Mit diesem FB können Werte von Parametern aus dem Antrieb ausgelesen werden. Alle Parameter außer Par. 23 („Gerätetyp als String“) können gelesen werden.



Execute

Start eines Lesevorgangs

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Beschreibung:

Bei einer steigenden Flanke wird ein Lesevorgang des mit „ParameterNumber“ und „Subindex“ spezifizierten Parameters gestartet. Für einen erneuten Lesevorgang muss erneut eine steigende Flanke generiert werden. Wird das Bit zurückgesetzt, so nehmen die Ausgänge die angegebenen Defaultwerte an.

ParameterNumber

Parameternummer des auszulesenden Parameters

- Typ: INT
- Anfangswert: 0
- Art: VAR_INPUT

SubIndex

Subindex des Parameters

- Typ: INT
- Anfangswert: 0
- Art: VAR_INPUT

MaxProcessTime

Maximale Zeit, nach der der FB die Kommunikation mit dem Antrieb beenden haben muss (in [ms]).

- Typ: DINT
- Anfangswert: 4000
- Art: VAR_INPUT

Beschreibung:

Der FB muss innerhalb der spezifizierten Zeit beendet sein, andernfalls gibt der FB einen Fehler zurück. Der Wert 4000 ist für die meisten Anwendungen ausreichend.

Active

Bit ist gesetzt, solange der Lesevorgang läuft

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

Das Bit wird zurückgesetzt, sobald der Wert gelesen wurde oder ein Fehler aufgetreten ist (→ Ausgänge „Done“ und „Error“ abprüfen).

Done

Bit ist gesetzt, sobald der Parameter erfolgreich gelesen wurde und an „Value“ anliegt

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

Das Bit wird beim Start eines Lesevorgangs zurückgesetzt.

Error

Bit ist gesetzt, wenn während der Ausführung des FBs ein Fehler aufgetreten ist

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

ErrorID

Fehler-ID (siehe Tabelle „ErrorID“ in Kap. 3)

- Typ: INT
- Defaultwert: 0
- Art: VAR_OUTPUT

Antriebsfehler („Drive errors“) werden beim Lesen eines Parameters nicht beachtet.

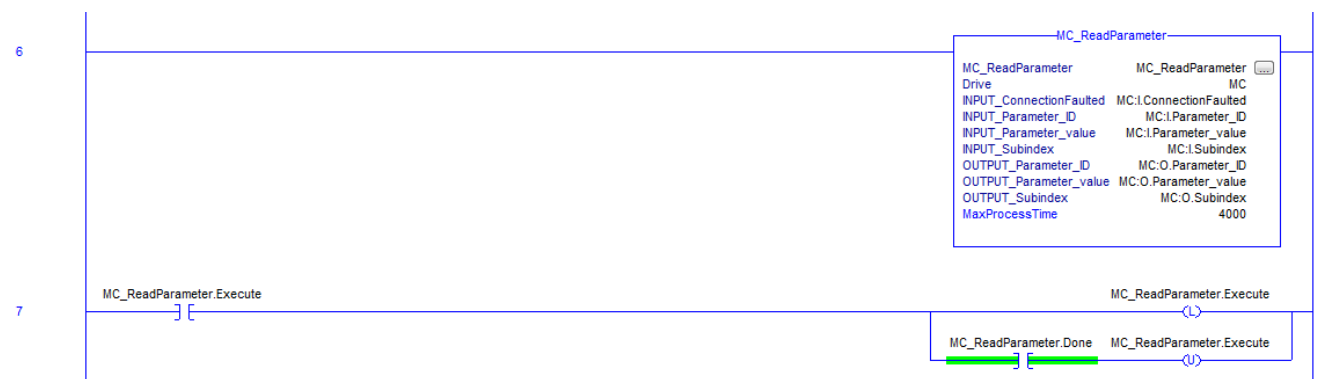
Value

Istwert des ausgelesenen Parameters

- Typ: DINT
- Defaultwert: 0
- Art: VAR_OUTPUT

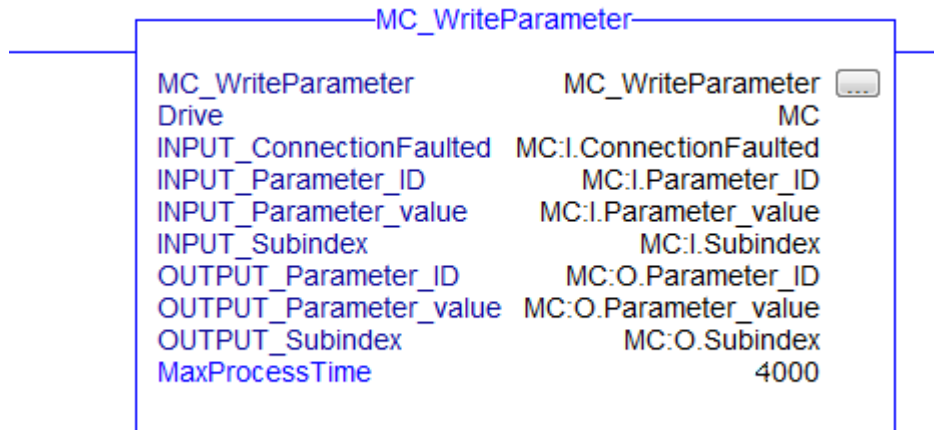
Bei einem Fehler wird 0 ausgegeben.

Einfaches Kontaktplan-Programm mit dem FB „MC_ReadParameter“:



4.5 MC_WriteParameter

Mit diesem FB können Parameterwerte in den Antrieb geschrieben werden.



Execute

Start eines Schreibvorgangs

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Beschreibung:

Bei einer steigenden Flanke wird ein Schreibvorgang des mit „ParameterNumber“ und „Subindex“ spezifizierten Parameters mit dem Wert aus dem Eingang „Value“ gestartet. Für einen erneuten Schreibvorgang muss erneut eine steigende Flanke generiert werden. Wird das Bit zurückgesetzt, so nehmen die Ausgänge die angegebenen Defaultwerte an.

ParameterNumber

Parameternummer des zu schreibenden Parameters

- Typ: INT
- Anfangswert: 0
- Art: VAR_INPUT

Value

zu schreibender Wert des betr. Parameters

- Typ: DINT
- Anfangswert: 0
- Art: VAR_INPUT

MaxProcessTime

Maximale Zeit, nach der der FB die Kommunikation mit dem Antrieb beenden muss (in [ms]).

- Typ: DINT
- Anfangswert: 4000
- Art: VAR_INPUT

Beschreibung:

Der FB muss innerhalb der spezifizierten Zeit beendet sein, andernfalls gibt der FB einen Fehler zurück. Der Wert 4000 ist für die meisten Anwendungen ausreichend.

Active

Bit ist gesetzt, solange der Schreibvorgang läuft

- Typ: BOOL
- Defaultwert: FALSE

- Art: VAR_OUTPUT

Das Bit wird zurückgesetzt, sobald der Wert geschrieben wurde oder ein Fehler aufgetreten ist (→ Ausgänge „Done“ und „Error“ abprüfen).

Done

Bit ist gesetzt, sobald der Parameter erfolgreich geschrieben wurde

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

Das Bit wird beim Start eines Schreibvorgangs zurückgesetzt.

Error

Bit ist gesetzt, wenn während der Ausführung des FBs ein Fehler aufgetreten ist

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

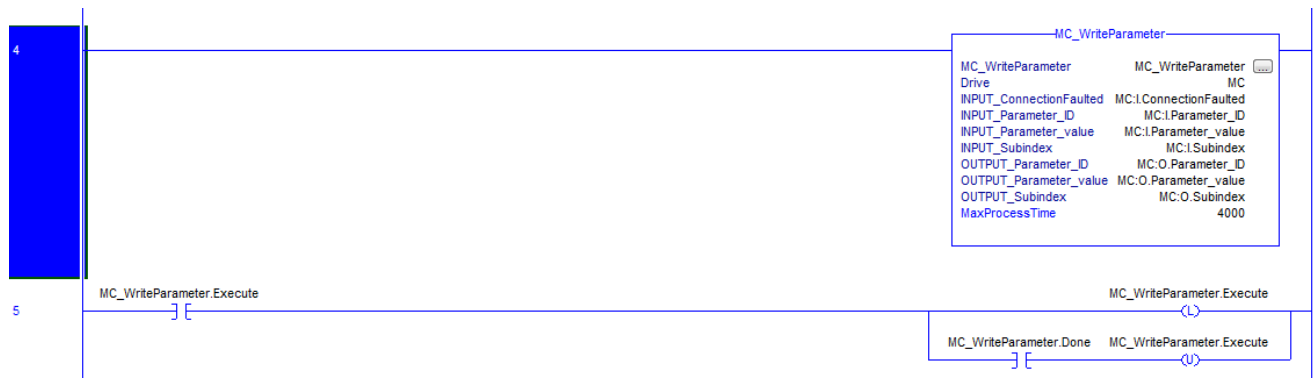
ErrorID

Fehler-ID (siehe Tabelle „ErrorID“ in Kap. 3)

- Typ: INT
- Defaultwert: 0
- Art: VAR_OUTPUT

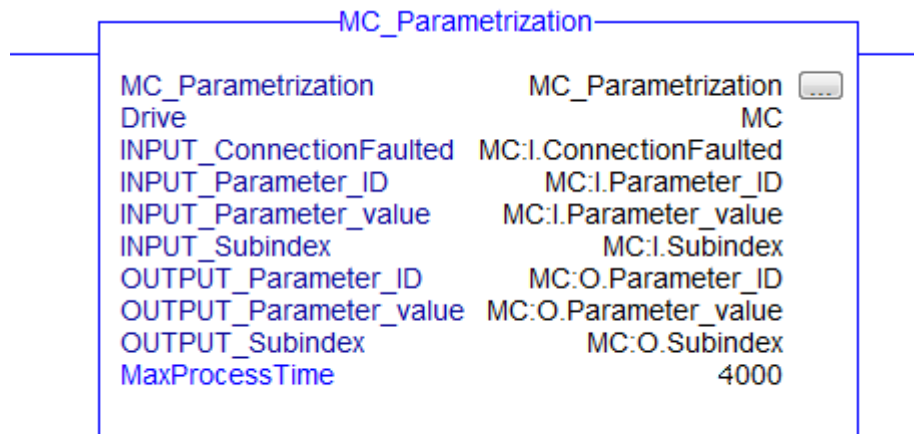
Antriebsfehler („Drive errors“) werden beim Schreiben eines Parameters nicht beachtet.

Einfaches Kontaktplan-Programm mit dem FB „MC_WriteParameter“:



4.6 MC_Parametrization

Mit diesem FB können die folgenden Parameter des Antriebs mit einem einzigen Funktionsaufruf geschrieben werden: 10, 26, 28, 30, 32, 40, 42, 44, 46, 48, 52, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78, 80, 84, 86, 108, 110 und 113



Folgendes ist bei der Nutzung des FBs zu beachten:

- Zu jedem Parameterwert gibt es zusätzlich ein „Enable Tag“, um festzulegen, ob der Parameter geschrieben werden soll oder nicht.
Bsp.: DirRotation_Enable = 1 → DirRotation_Value wird gesetzt
- Die Parameter werden in der folgenden Reihenfolge geschrieben:
 - „DeliveryState“ (Par. 113) →
 - „DirRotation“ (Par. 26) →
 - „PosScaleNumerator“ (Par. 28) →
 - ... →
 - „MaxTemperature“ (Par. 110) →
 - „SaveSettings“ (Par. 113)
 Dazwischen werden die Parameter in aufsteigender Reihenfolge geschrieben.
- Wahlweise kann vor dem Setzen einzelner Parameter ein Auslieferungszustand angefordert werden. Dazu muss vor der Ausführung des FBs der Eingang „DeliveryState_113“ auf TRUE gesetzt werden. Dadurch werden die Werte aller Parameter auf den Auslieferungszustand gesetzt (zunächst ohne zu speichern).
- Bei denjenigen Parametern, bei denen das Enable gesetzt ist (→ „x_Enable“ ist gesetzt), werden die Werte vom Auslieferungszustand mit demjenigen Wert überschrieben, der in „x_Value“ vorgeben wurde.
- Wahlweise können die geschriebenen Werte am Ende auch gespeichert werden. Dazu muss vor der Ausführung des FBs der Eingang „SaveSettings_113“ auf TRUE gesetzt werden.
- Bei einem Schreibfehler eines Parameters werden die nachfolgenden Parameter nicht mehr geschrieben und es erfolgt auch kein Speichern der Werte, falls der Eingang „SaveSettings“ gesetzt ist.

Execute

Start eines Parametriervorgangs

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Beschreibung:

Bei einer steigenden Flanke wird ein Parametriervorgang mit den angegebenen Werten gestartet. Für einen erneuten Parametriervorgang muss erneut eine steigende Flanke generiert werden. Wird das Bit zurückgesetzt, so nehmen die Ausgänge die angegebenen Defaultwerte an.

DeliveryState

Laden der Werkseinstellungen (zunächst ohne Speichern)

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

IP-Adresse und Adressvergabemethode bleiben jedoch unbeeinflusst.

x_Enable

Falls gesetzt, wird der betr. Parameter geschrieben

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

x_Value

Sollwert des Parameters

- Anfangswert: 0
- Art: VAR_INPUT

Die Parameternummer ist hinter dem Parameternamen angegeben. Der Datentyp, eine Beschreibung sowie der Wertebereich kann der Betriebsanleitung des PSx-3__-EI entnommen werden.

SaveSettings

Permanentes Speichern der Einstellungen

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

MaxProcessTime

Maximale Zeit, nach der der FB die Kommunikation mit dem Antrieb beendet haben muss (in [ms]).

- Typ: DINT
- Anfangswert: 4000
- Art: VAR_INPUT

Beschreibung:

Der FB muss innerhalb der spezifizierten Zeit beendet sein, andernfalls gibt der FB einen Fehler zurück. Der Wert 4000 ist für die meisten Anwendungen ausreichend.

Active

Bit ist gesetzt, solange die Parametrierung läuft

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

Das Bit wird zurückgesetzt, sobald die Parameterisierung erfolgreich beendet wurde oder ein Fehler aufgetreten ist (→ Ausgänge „Done“ und „Error“ abprüfen).

Done

Bit ist gesetzt, sobald die Parameterisierung erfolgreich beendet wurde

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

Das Bit wird beim Start einer Parametrierung zurückgesetzt.

Error

Bit ist gesetzt, wenn während der Ausführung des FBs ein Fehler aufgetreten ist

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

ErrorID

Fehler-ID (siehe Tabelle „ErrorID“ in Kap. 3)

- Typ: INT
- Defaultwert: 0
- Art: VAR_OUTPUT

Antriebsfehler („Drive errors“) werden bei einer Parametrierung nicht beachtet.

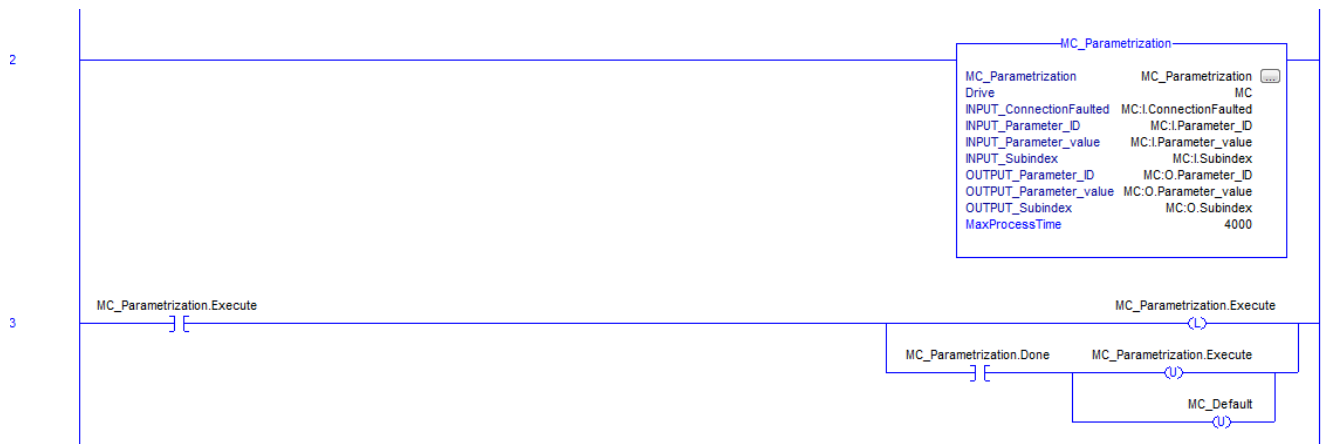
ErrorParameter

Parameternummer, bei dem im Fall eines Fehlers der Fehler aufgetreten ist

- Typ: INT
- Defaultwert: 0
- Art: VAR_OUTPUT

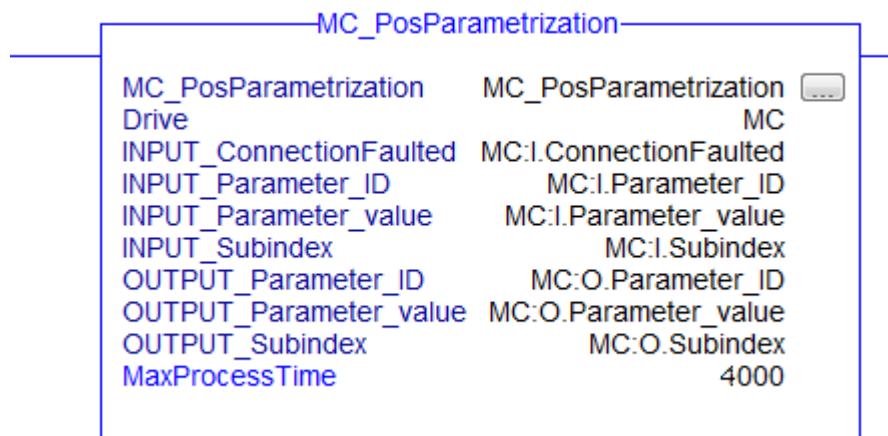
Falls es keinen Fehler gab, wird 0 ausgegeben.

Einfaches Kontaktplan-Programm mit dem FB „MC Parametrization“:



4.7 MC_PositionParametrization

Mit diesem FB kann die Parametrierung der Positionsdaten vorgenommen werden (Parameter, die die angezeigte Istposition beeinflussen).



Folgendes ist bei der Nutzung des FBs zu beachten:

- Es müssen alle Werte gesetzt werden und die Werte müssen in einem sinnvollen Bezug zueinander stehen. Alle Werte werden verarbeitet, danach werden die folgenden Parameter in der angeg. Reihenfolge geschrieben:
 - Drehsinn (Par. 26) = Direction
 - Istwertbewertung Zähler (Par. 28) = 400
 - Istwertbewertung Nenner (Par. 30) = StepsPerTurn
 - Istwert (Par. 10) = SetPoint
 - Falls (SetPoint > UpperLimit):
 - oberes Mapping-Ende (Par. 34) = SetPoint + (3 x StepsPerTurn)
 - sonst:
 - oberes Mapping-Ende (Par. 34) = UpperLimit + (3 x StepsPerTurn)
 - obere Endbegrenzung (Par. 36) = UpperLimit
 - untere Endbegrenzung (Par. 38) = LowerLimit
- Die Anzahl der Schritte pro Umdrehung „StepsPerTurn“ ergibt unmittelbar den Wert des Parameters „Istwertbewertung Nenner“ (Par. 30). Dabei wird angenommen, dass der Wert von „Istwertbewertung Zähler“ (Par. 28) im Auslieferungszustand ist, also auf 400.
- Vor dem Schreiben der Parameter werden die eingegebenen Werte auf Gültigkeit geprüft.

Nachfolgend die Bedingungen und die Fehlermeldungen, die bei nicht erfüllter Bedingung ausgegeben werden.

Bedingung	ErrorID	ErrorParameter
$\text{StepsPerTurn} \geq 1$	16#6140	39
$\text{StepsPerTurn} \leq 10000$	16#6140	39
$\text{LowerLimit} \leq \text{UpperLimit}$	16#6140	42
$(\text{UpperLimit} - \text{LowerLimit}) / \text{StepsPerTurn} \leq 250$	16#6140	43
Falls $\text{SetPoint} < \text{LowerLimit}$: $(\text{UpperLimit} - \text{SetPoint}) / \text{StepsPerTurn} \leq 250$	16#6140	3
Falls $\text{SetPoint} > \text{UpperLimit}$: $(\text{SetPoint} - \text{LowerLimit}) / \text{StepsPerTurn} \leq 250$	16#6140	3

- Wahlweise können die geschriebenen Werte am Ende auch gespeichert werden. Dazu muss vor der Ausführung des FBs der Eingang „SaveSettings“ auf TRUE gesetzt werden.

- Bei einem Schreibfehler eines Parameters werden die nachfolgenden Parameter nicht mehr geschrieben und es erfolgt auch kein Speichern der Werte, falls der Eingang „SaveSettings“ gesetzt ist.

Execute

Start eines Parametriervorgangs

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

Beschreibung:

Bei einer steigenden Flanke wird ein Parametriervorgang mit den angegebenen Werten gestartet. Für einen erneuten Parametriervorgang muss erneut eine steigende Flanke generiert werden. Wird das Bit zurückgesetzt, so nehmen die Ausgänge die angegebenen Defaultwerte an.

Direction

Richtung, in der der Antrieb bei größeren Werten drehen soll (bei Sicht auf die Abtriebswelle):

0 → CW, 1 → CCW

- Typ: INT
- Anfangswert: 0
- Art: VAR_INPUT

StepsPerTurn

Schritte pro Umdrehung an der Abtriebswelle (Auflösung)

- Typ: INT
- Anfangswert: 0
- Art: VAR_INPUT

LowerLimit

Untere Endbegrenzung

- Typ: DINT
- Anfangswert: 0
- Art: VAR_INPUT

UpperLimit

Obere Endbegrenzung

- Typ: DINT
- Anfangswert: 0
- Art: VAR_INPUT

SetPoint

Wert, auf den das Messsystem referenziert wird (neuer Istwert an der aktuellen Position)

- Typ: DINT
- Anfangswert: 0
- Art: VAR_INPUT

SaveSettings

Permanentes Speichern der Einstellungen

- Typ: BOOL
- Anfangswert: FALSE
- Art: VAR_INPUT

MaxProcessTime

Maximale Zeit, nach der der FB die Kommunikation mit dem Antrieb beendet haben muss (in [ms]).

- Typ: DINT
- Anfangswert: 4000
- Art: VAR_INPUT

Beschreibung:

Der FB muss innerhalb der spezifizierten Zeit beendet sein, andernfalls gibt der FB einen Fehler zurück. Der Wert 4000 ist für die meisten Anwendungen ausreichend.

Active

Bit ist gesetzt, solange die Parametrierung läuft

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

Das Bit wird zurückgesetzt, sobald die Parameterisierung erfolgreich beendet wurde oder ein Fehler aufgetreten ist (→ Ausgänge „Done“ und „Error“ abprüfen).

Done

Bit ist gesetzt, sobald die Parameterisierung erfolgreich beendet wurde

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

Das Bit wird beim Start einer Parametrierung zurückgesetzt.

Error

Bit ist gesetzt, wenn während der Ausführung des FBs ein Fehler aufgetreten ist

- Typ: BOOL
- Defaultwert: FALSE
- Art: VAR_OUTPUT

ErrorID

Fehler-ID (siehe Tabelle „ErrorID“ in Kap. 3)

- Typ: INT
- Defaultwert: 0
- Art: VAR_OUTPUT

Antriebsfehler („Drive errors“) werden bei einer Parametrierung nicht beachtet.

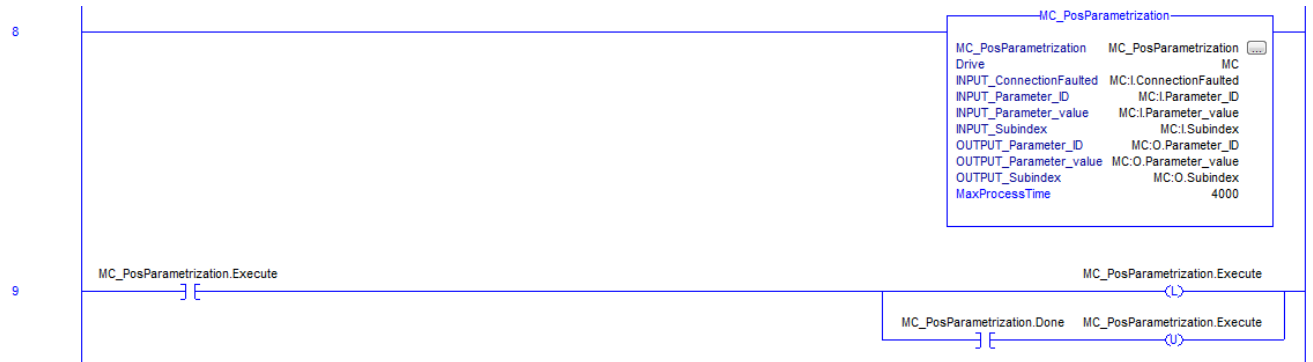
ErrorParameter

Parameternummer, bei dem im Fall eines Fehlers der Fehler aufgetreten ist

- Typ: INT
- Defaultwert: 0
- Art: VAR_OUTPUT

Falls es keinen Fehler gab, wird 0 ausgegeben.

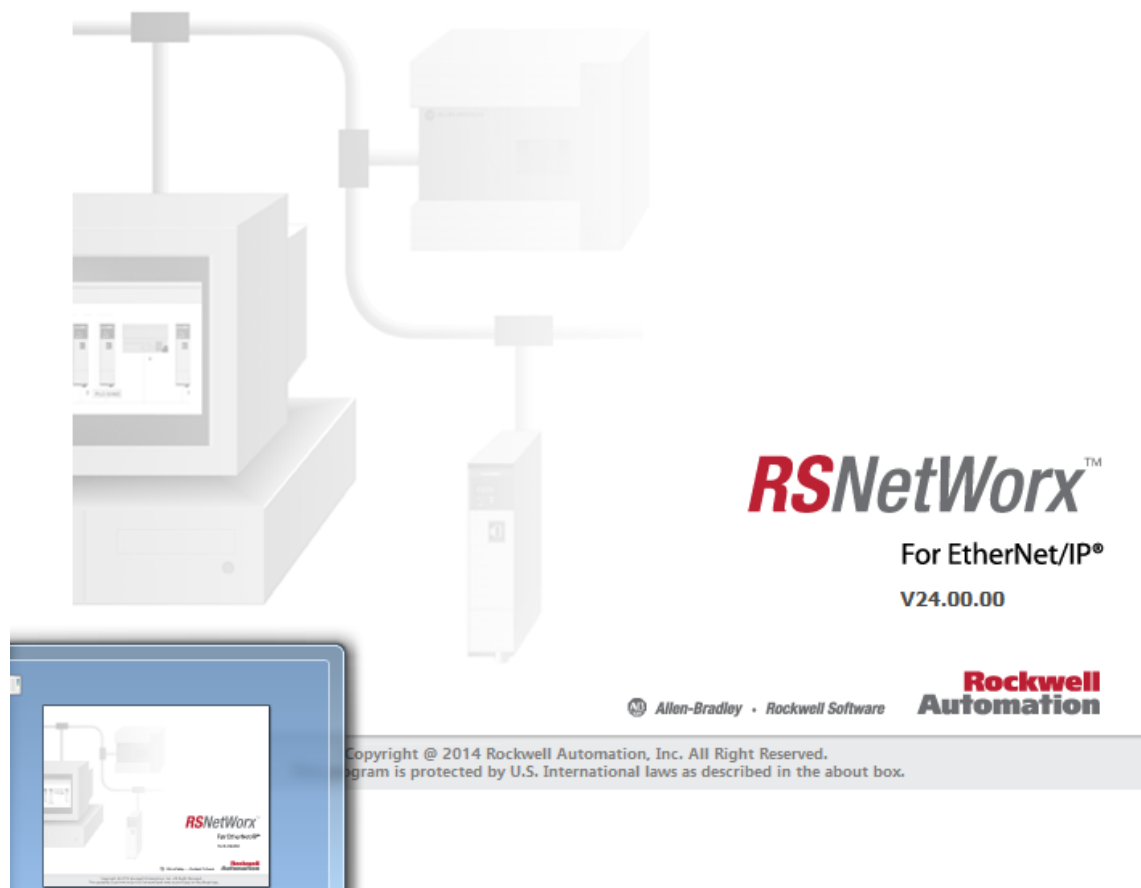
Einfaches Kontaktplan-Programm mit dem FB „MC_PositionParametrization“:



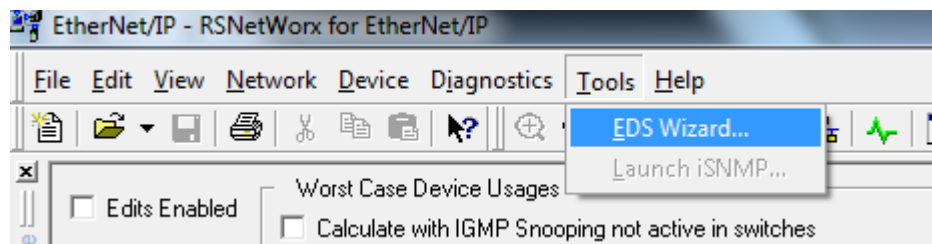
5 Hinzufügen von Antrieben zu einem Projekt

5.1 Betreffende EDS-Datei mit Hilfe von RSNetWorx registrieren

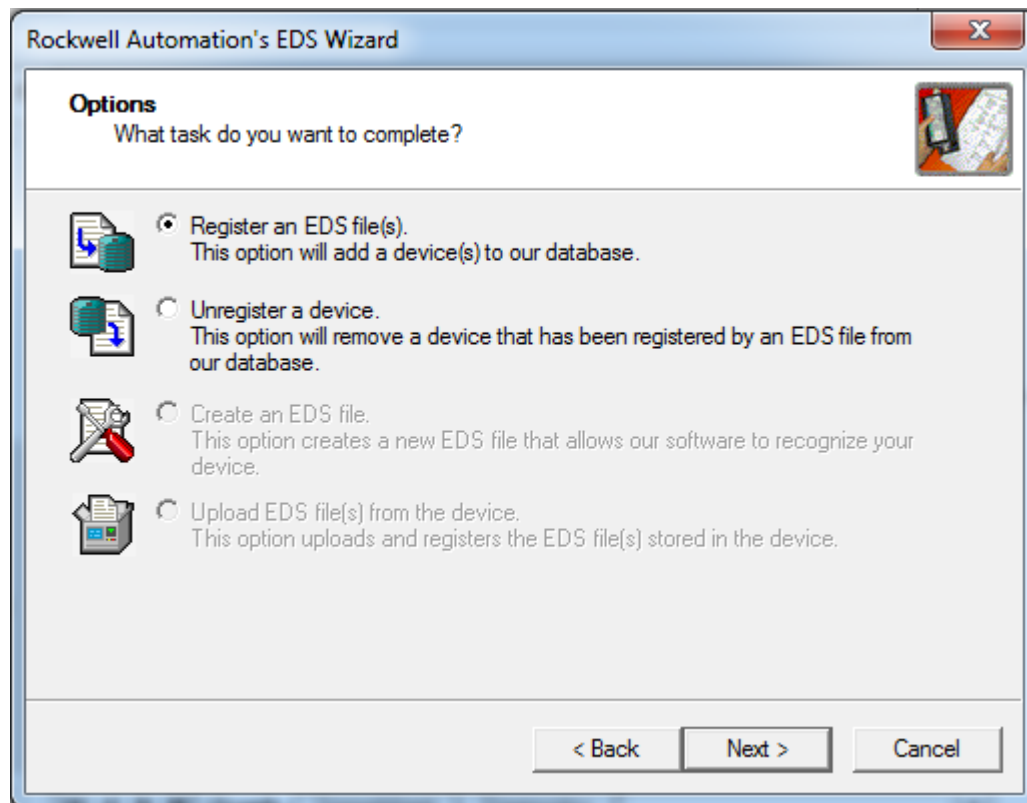
- Zunächst muss die betr. EDS-Datei registriert werden. Dazu die Software „RSNetWorx for Ethernet I/P“ starten:



- Dann „Tools → EDS Wizard“ anwählen:



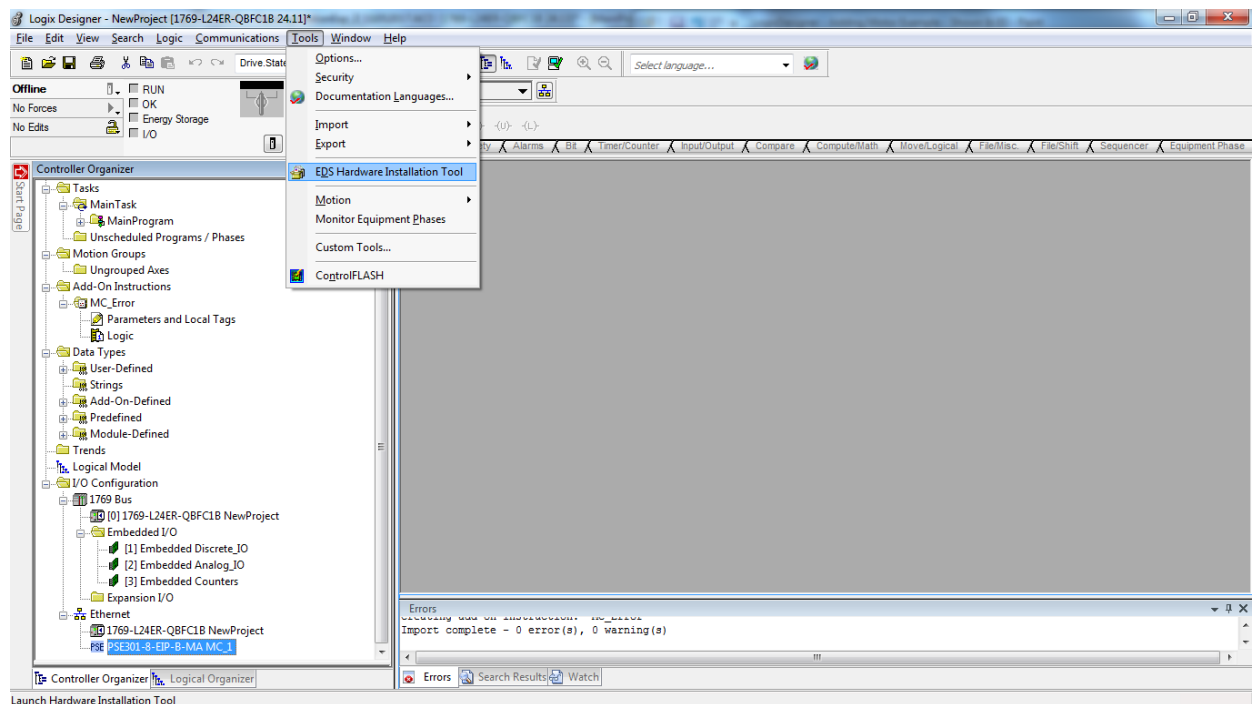
- Den Anweisungen des Assistenten folgen, um die Dateien zu registrieren:



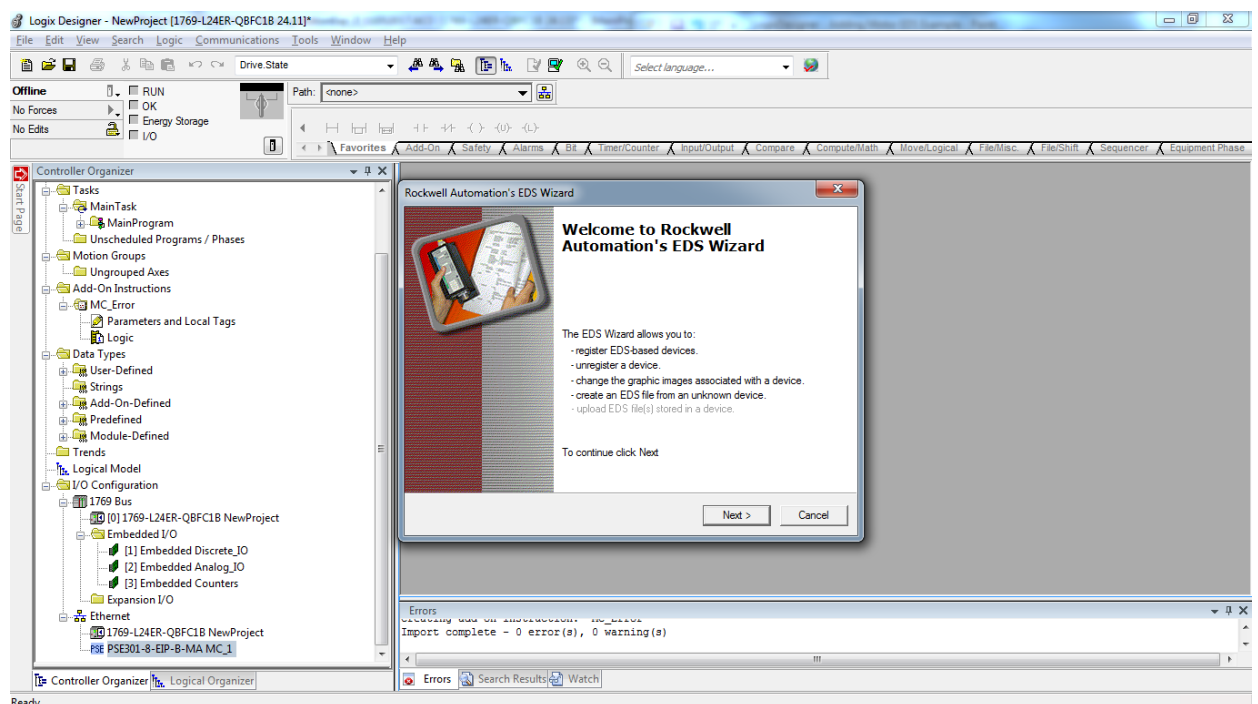
- Nach Abschluss der Registrierung RSNetWorx beenden.

5.2 Betreffende EDS-Datei mit Hilfe von Logix Designer registrieren

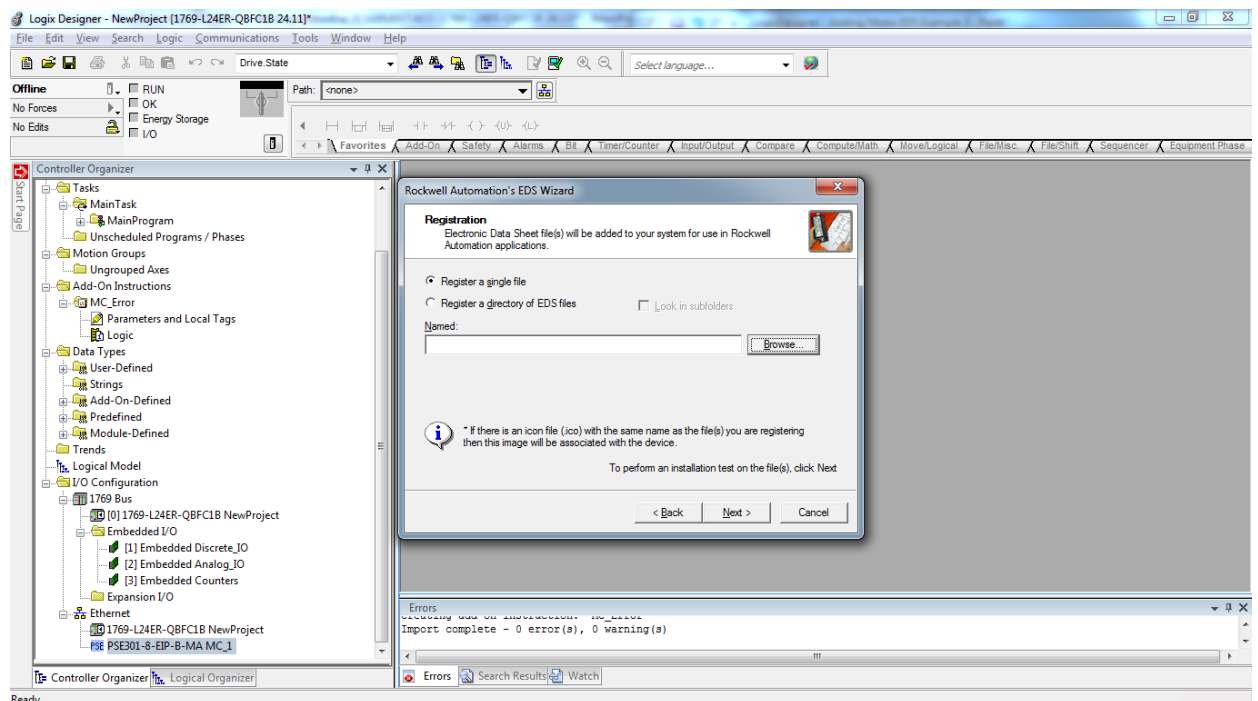
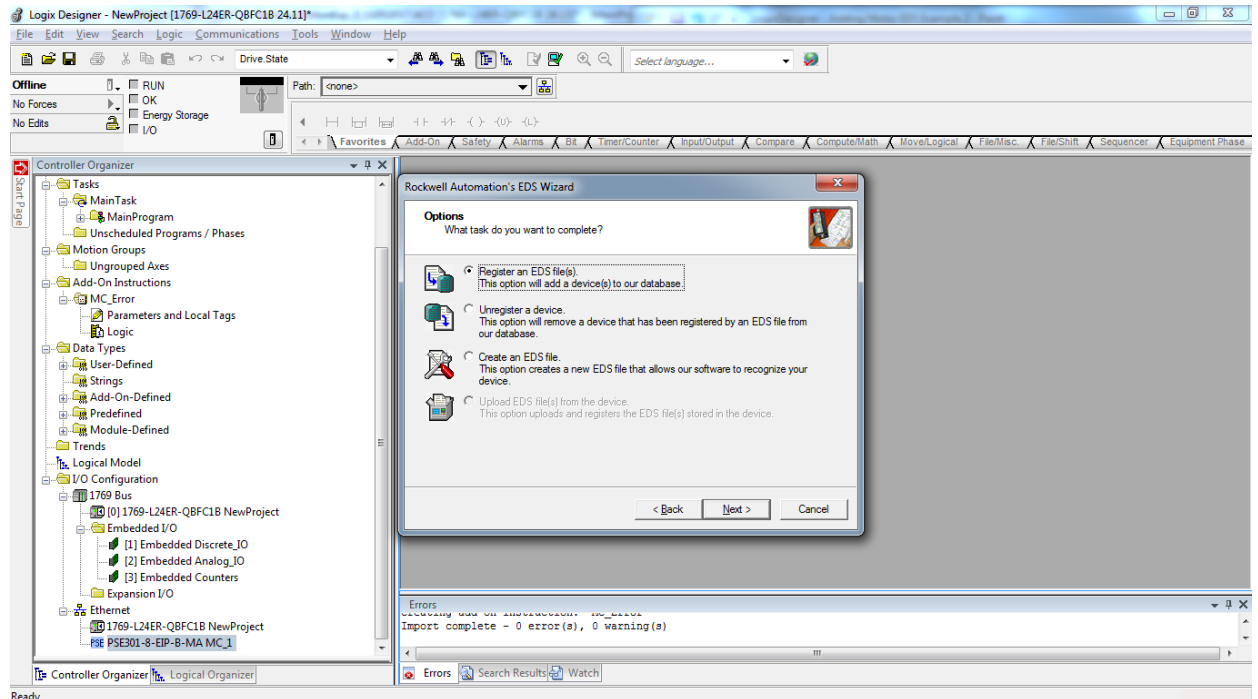
- Alternativ kann die EDS-Datei mit Hilfe des Logix Designers registriert werden. Zunächst „Tools → EDS Hardware Installation Tool“ anwählen:

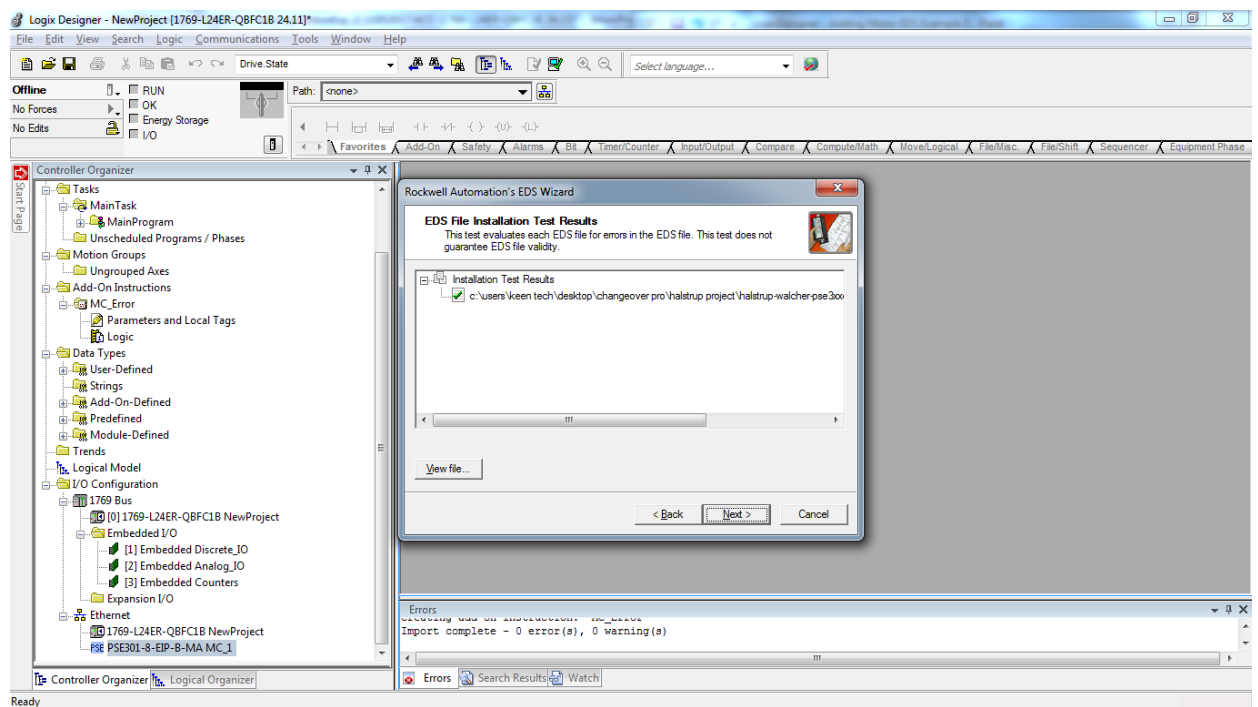
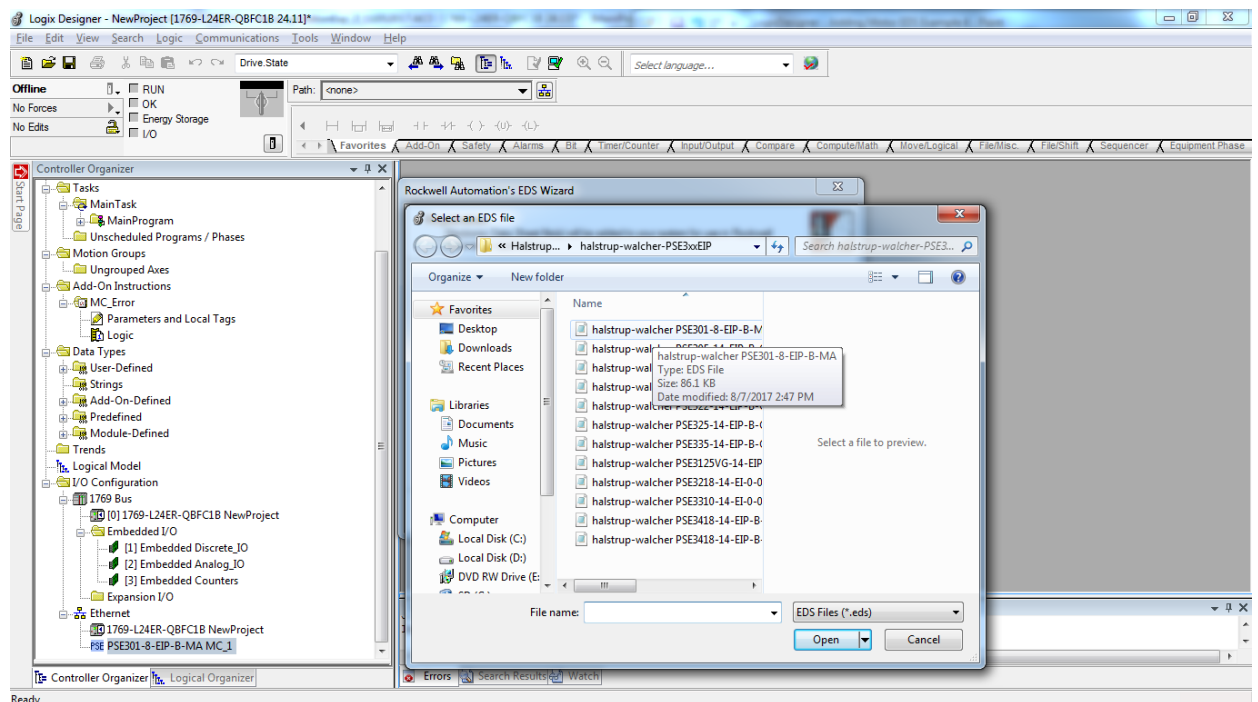


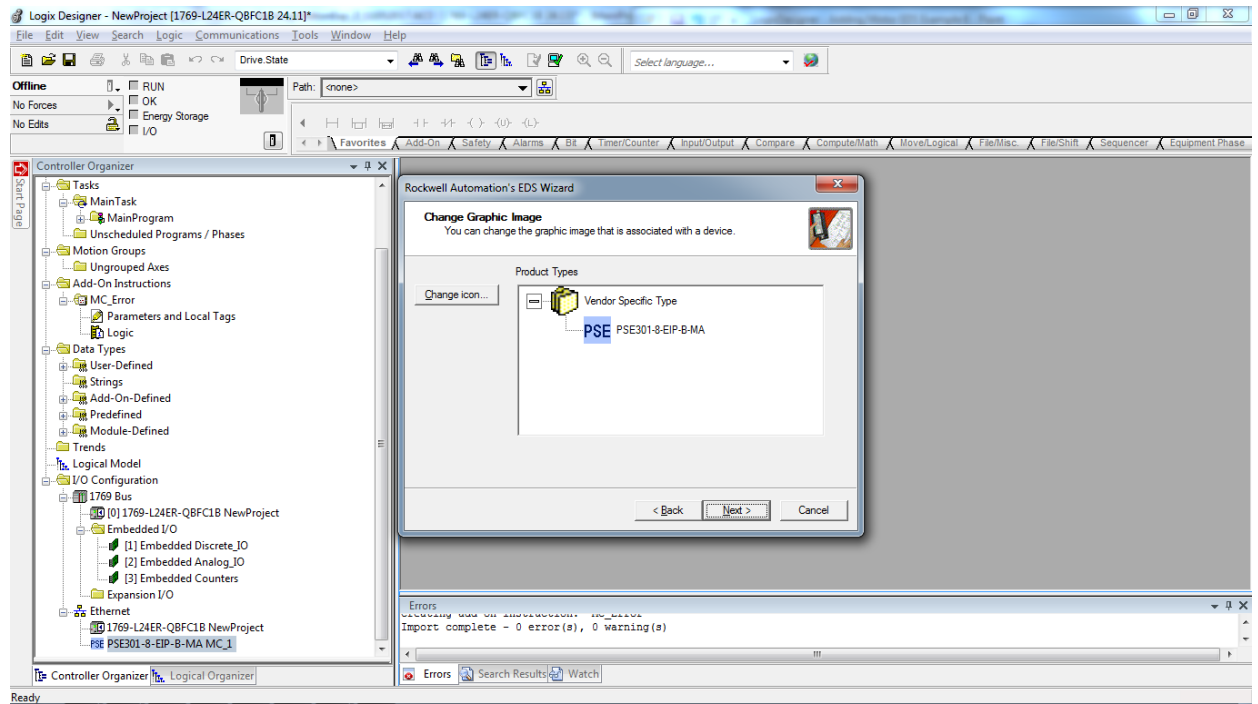
- Dann „Next“ anwählen, um den EDS Wizard zu starten:



- Den Anweisungen des Assistenten folgen, um die Dateien zu registrieren:

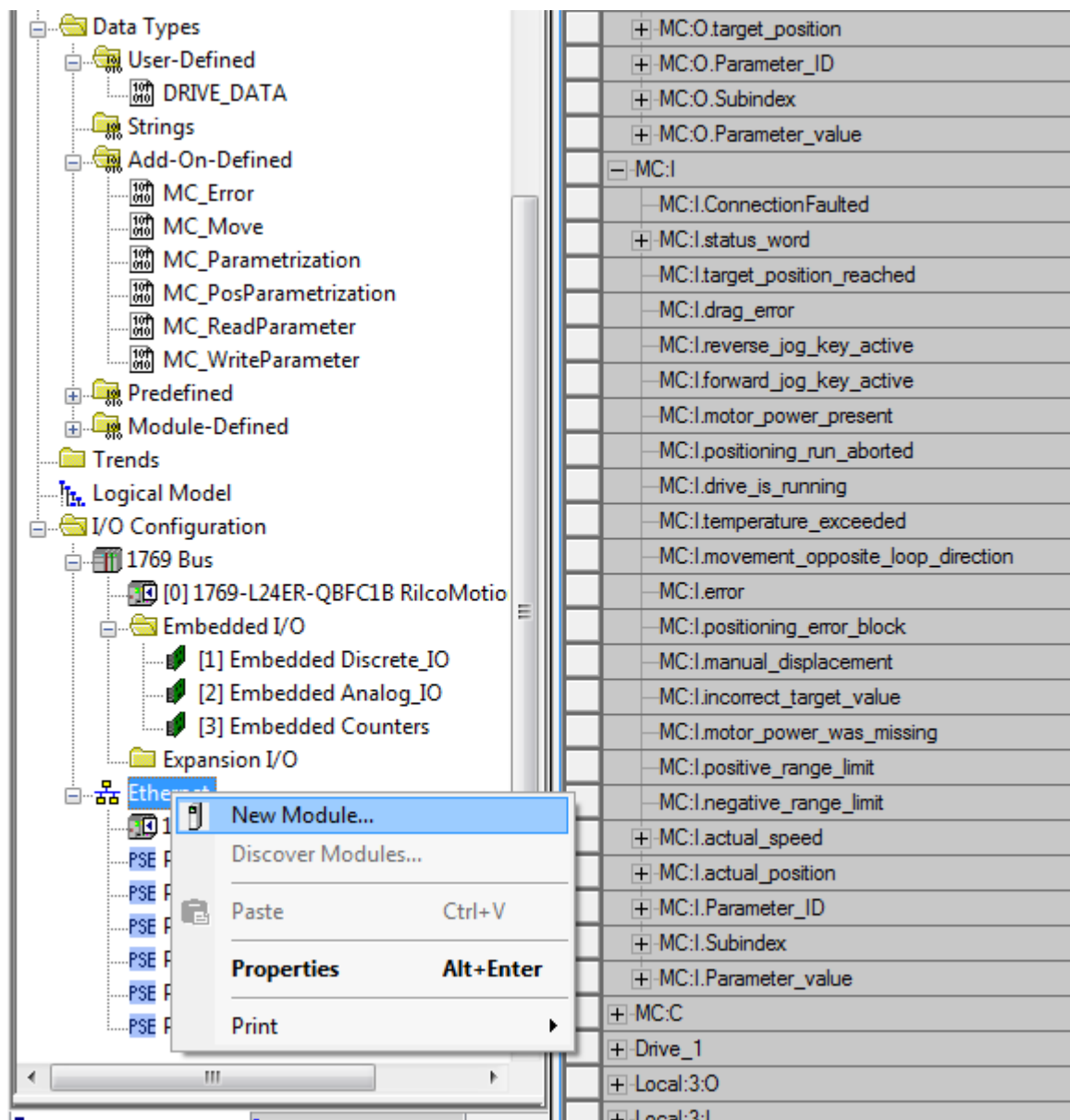




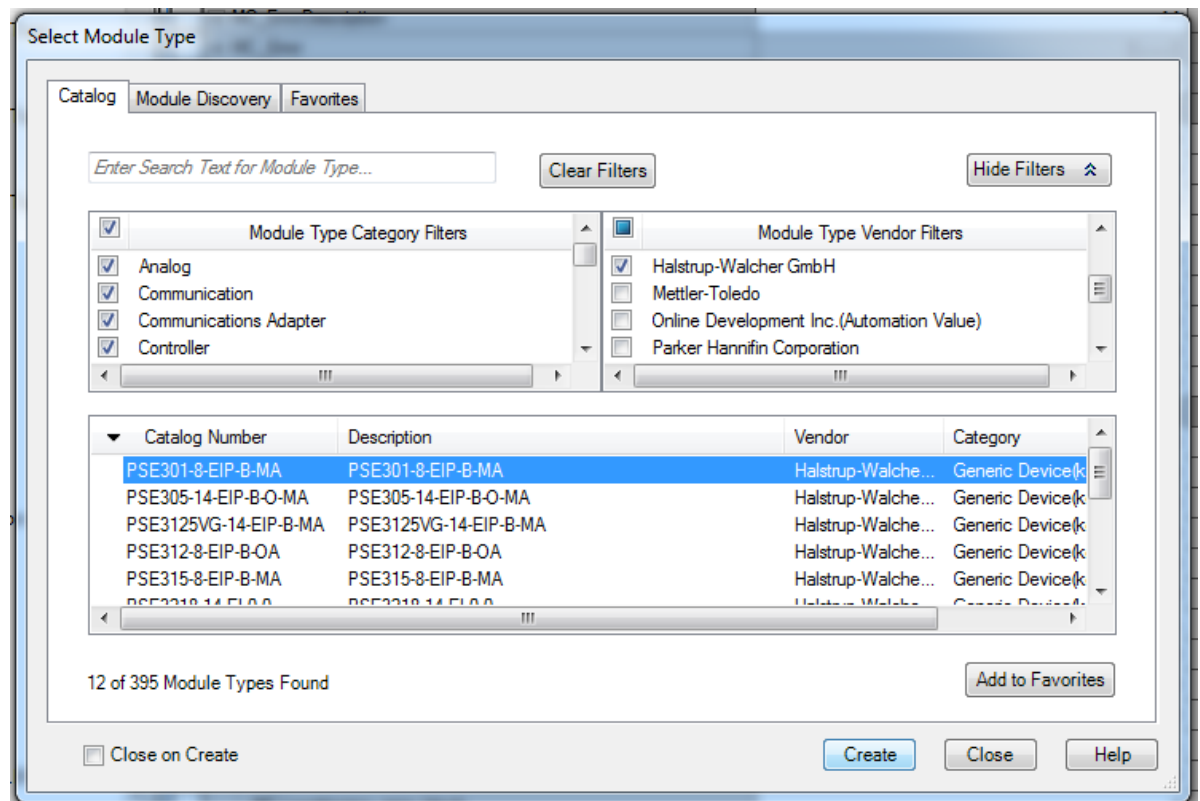


5.3 Einen Antrieb hinzufügen

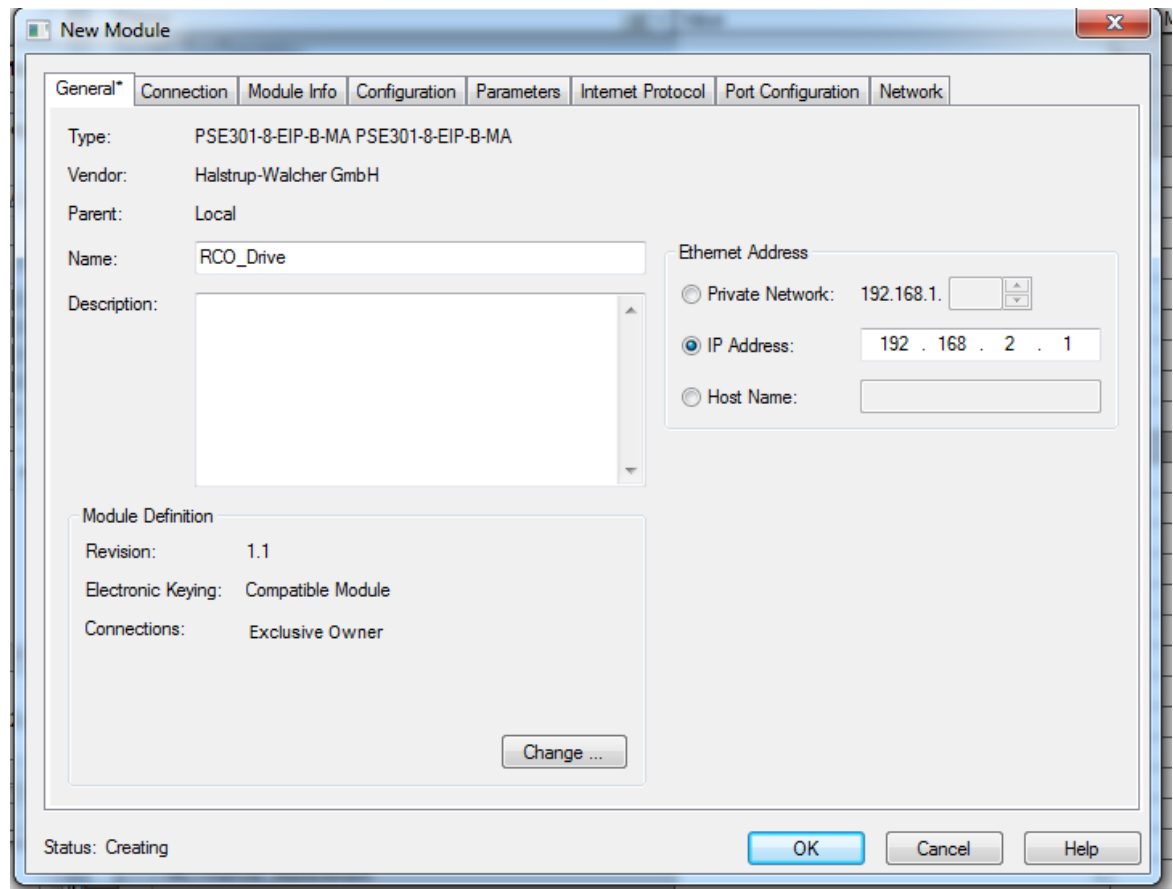
- Um einen Antrieb hinzuzufügen, ist das Studio 5000 oder RS Logix zu starten. Am unteren Ende des Gerätebaumes „Ethernet → New Module“ anwählen:



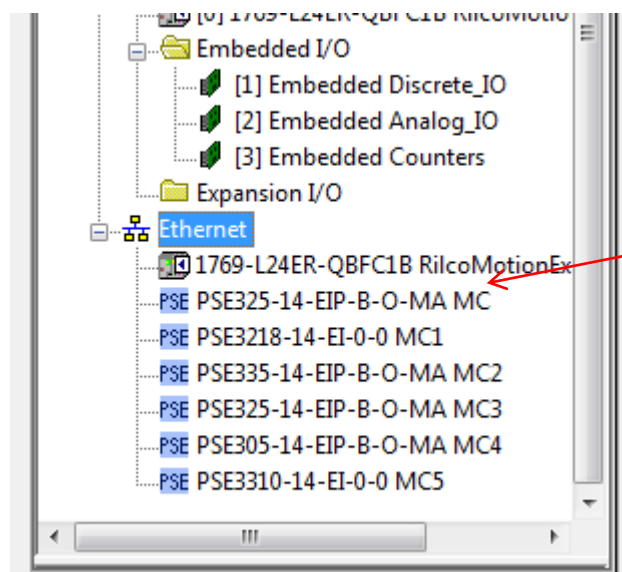
- Das Dialogfenster „Select Module Type“ erscheint. Das gewünschte Gerät anwählen und auf „Create“ klicken:



- Einen Namen und eine IP-Adresse eingeben und auf „OK“ klicken:



- Anschließend im Gerätebaum den Abschnitt „Ethernet“ beachten:



→ Das neue Gerät erscheint in der Modulliste.

- Anschließend die Controller tags beachten. Es wurden automatisch die relevanten Ein- und Ausgangsdaten angelegt.

Im Folgenden sind die Ausgangsdaten dargestellt (die Daten, die an den Antrieb gesendet werden):

MC:O	{...}	{...}		_03F2:P
MC:O.control_word	2#0000_0000_0000_0000		Binary	INT
MC:O.manual_run_to_larger_values	0		Decimal	BOOL
MC:O.manual_run_to_smaller_values	0		Decimal	BOOL
MC:O.transfer_target_position	0		Decimal	BOOL
MC:O.release_the_axle_will_only_run_if_the_bi	0		Decimal	BOOL
MC:O.target_position	12000		Decimal	DINT
MC:O.Parameter_ID	2#0000_0000_0000_0000		Binary	INT
MC:O.Subindex	16#0000		Hex	INT
MC:O.Parameter_value	0		Decimal	DINT

Im Folgenden sind die Eingangsdaten dargestellt (die Daten, die vom Antrieb empfangen werden):

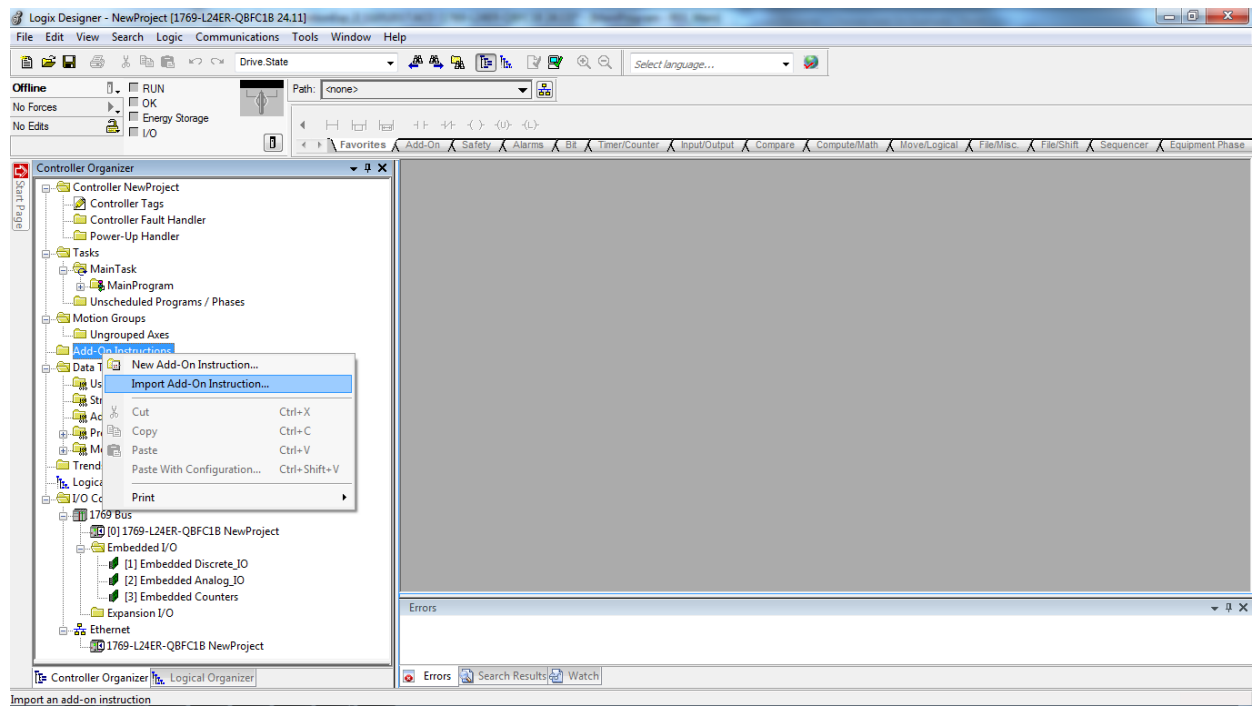
MC:I	{...}	{...}		_03F2:I
MC:I.ConnectionFaulted	0		Decimal	BOOL
MC:I.status_word	2#1000_1001_0001_0000		Binary	INT
MC:I.target_position_reached	0		Decimal	BOOL
MC:I.drag_error	0		Decimal	BOOL
MC:I.reverse_jog_key_active	0		Decimal	BOOL
MC:I.forward_jog_key_active	0		Decimal	BOOL
MC:I.motor_power_present	1		Decimal	BOOL
MC:I.positioning_run_aborted	0		Decimal	BOOL
MC:I.drive_is_running	0		Decimal	BOOL
MC:I.temperature_exceeded	0		Decimal	BOOL
MC:I.movement_opposite_jog_direction	1		Decimal	BOOL
MC:I.error	0		Decimal	BOOL
MC:I.positioning_error_block	0		Decimal	BOOL
MC:I.manual_displacement	1		Decimal	BOOL
MC:I.incorrect_target_value	0		Decimal	BOOL
MC:I.motor_power_was_missing	0		Decimal	BOOL
MC:I.positive_range_limit	0		Decimal	BOOL
MC:I.negative_range_limit	1		Decimal	BOOL
MC:I.actual_speed	0		Decimal	INT
MC:I.actual_position	167200		Decimal	DINT
MC:I.Parameter_ID	16#0000		Hex	INT
MC:I.Subindex	0		Decimal	INT
MC:I.Parameter_value	0		Decimal	DINT

(Die Anordnung und die Typen dieser Variablen sind durch die EDS-Datei definiert, die zu dem betr. Antrieb gehört.)

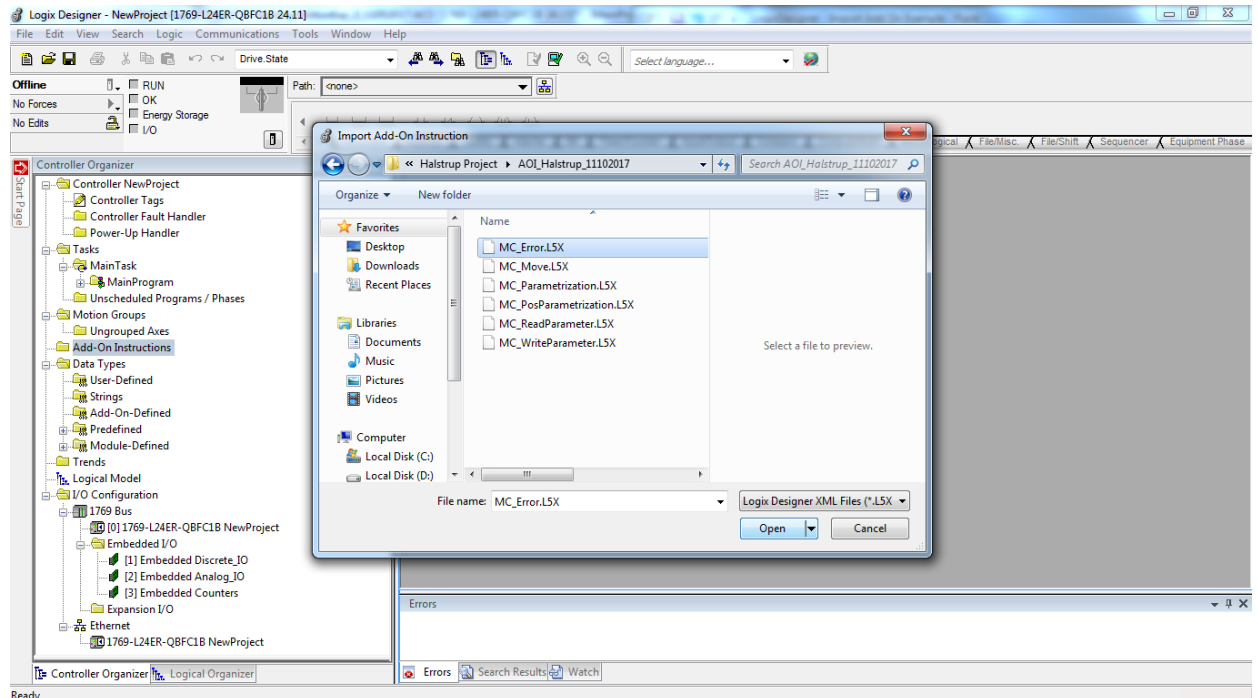
6 Hinzufügen von Funktionsblöcken zu einem Projekt

6.1 Einen Funktionsblock importieren

- Im Controller Organizer mit Rechtsklick auf „Add-On Instructions“, dann auf „Import Add-On Instruction“ klicken:

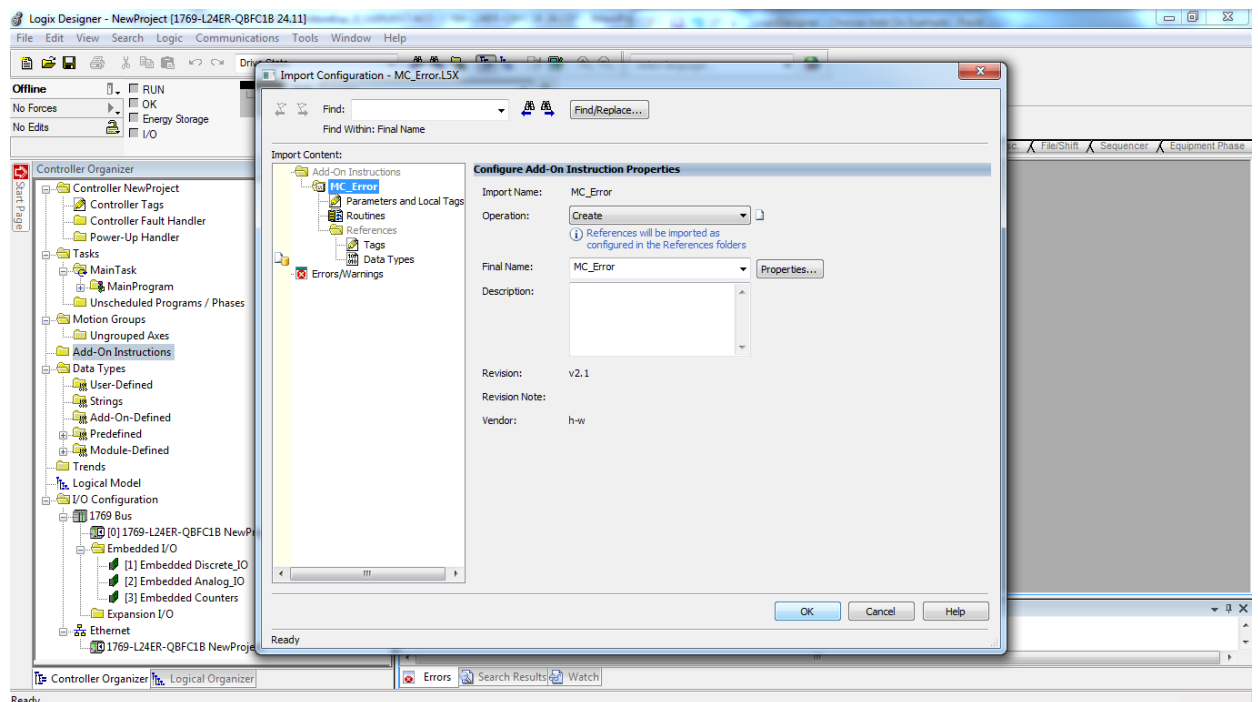


- Die FB-Bibliothek von halstrup-walcher anwählen und den gewünschten FB auswählen:



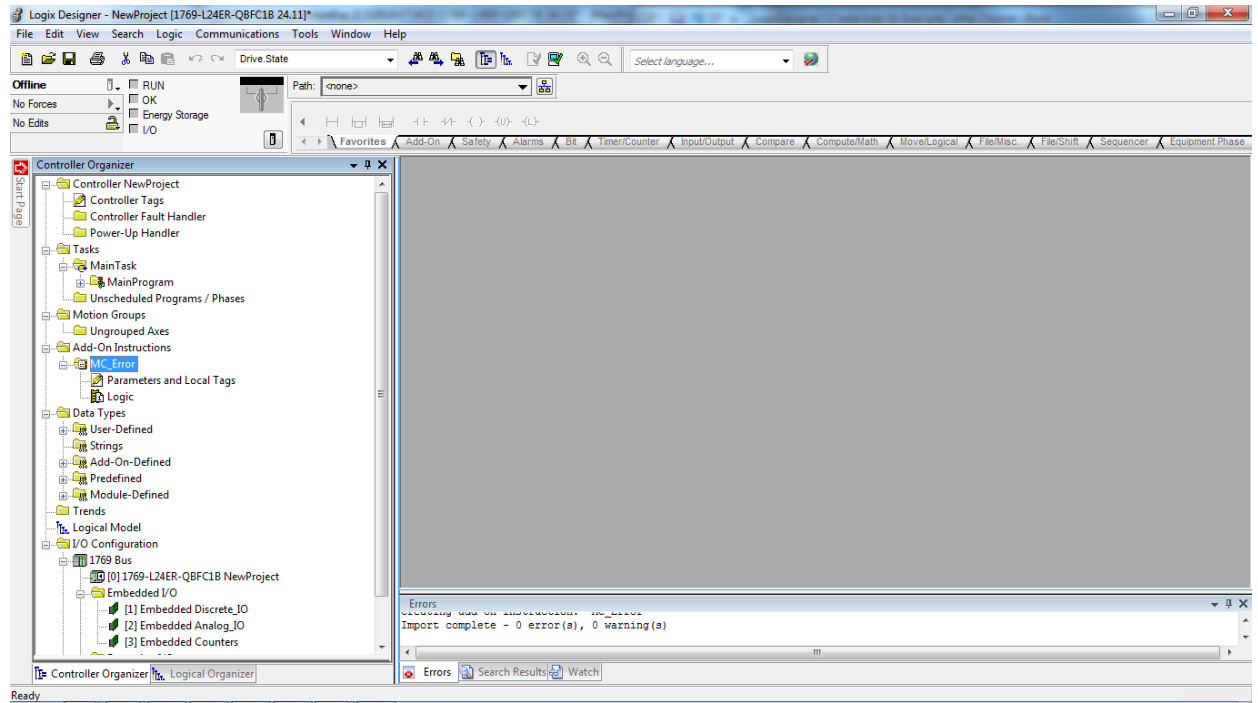
Dann auf „Open“ klicken.

- Anschließend den FB konfigurieren:



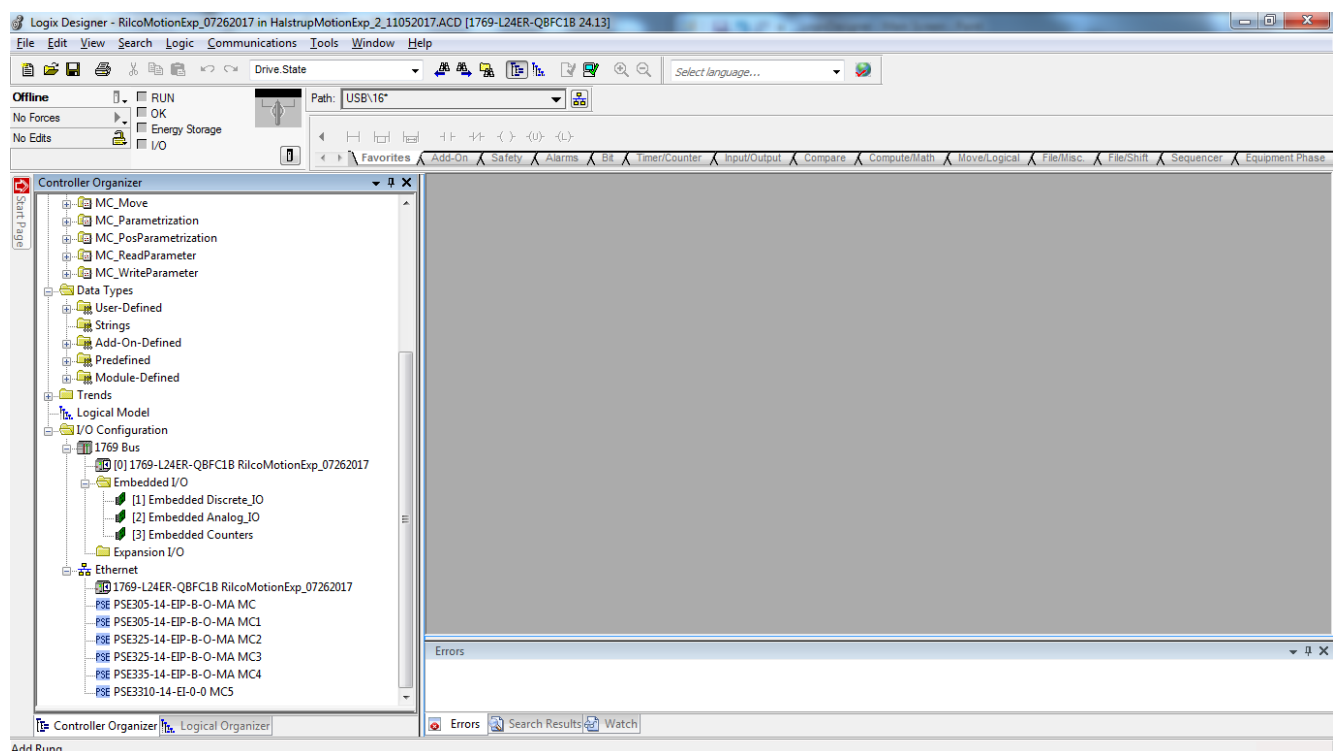
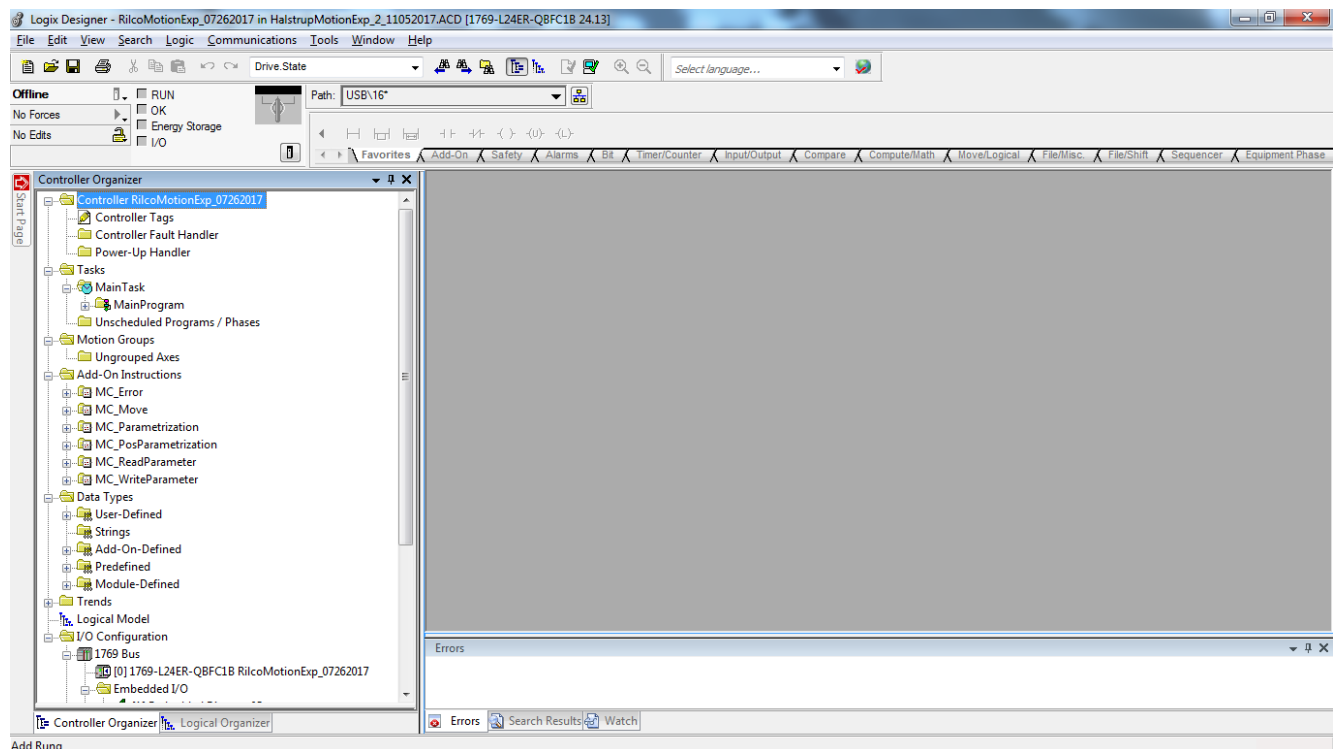
(Die Defaulteinstellungen sind in den meisten Anwendungen ausreichend.)

- Im Ergebnis sind die Funktionsbausteine im Abschnitt „Add-On Instructions“ des Controller Organizers aufgelistet:



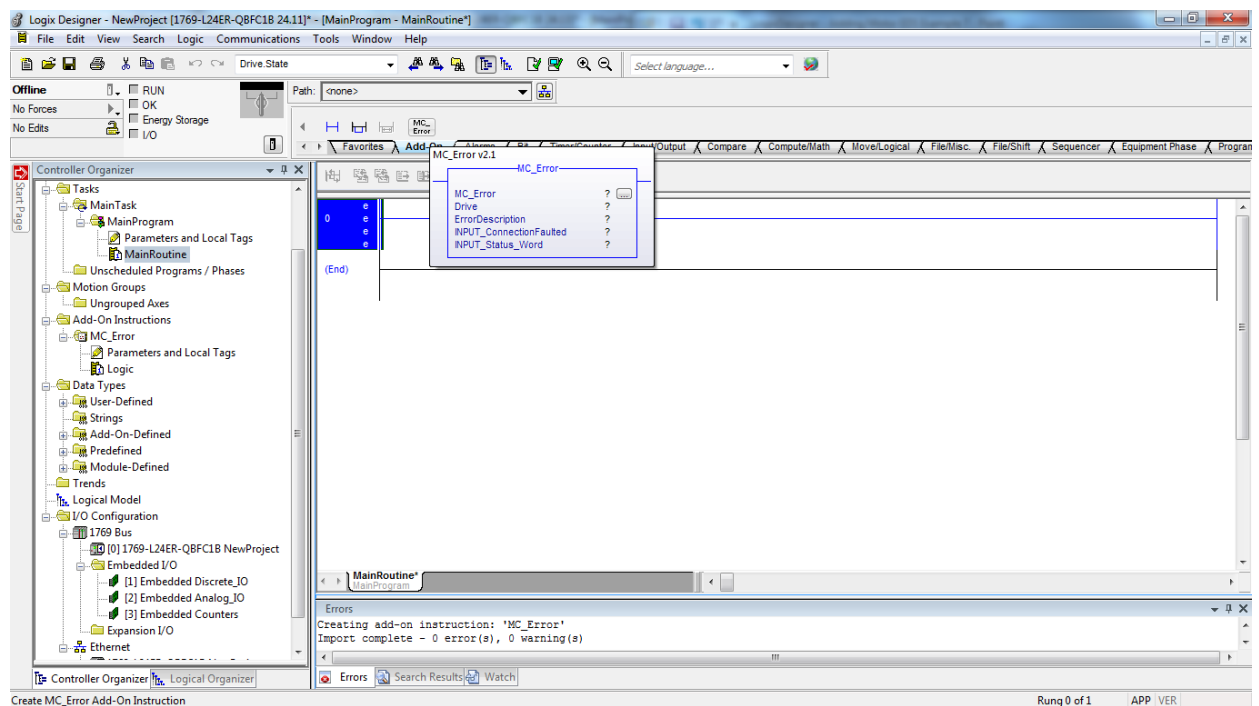
6.2 Darstellung der Antriebe und FBs im Controller Organizer

Zusammenfassend folgt eine Darstellung der gewünschten Antriebe und der gewünschten Funktionsblöcke im Controller Organizer im Hauptfenster von Logix Designer:

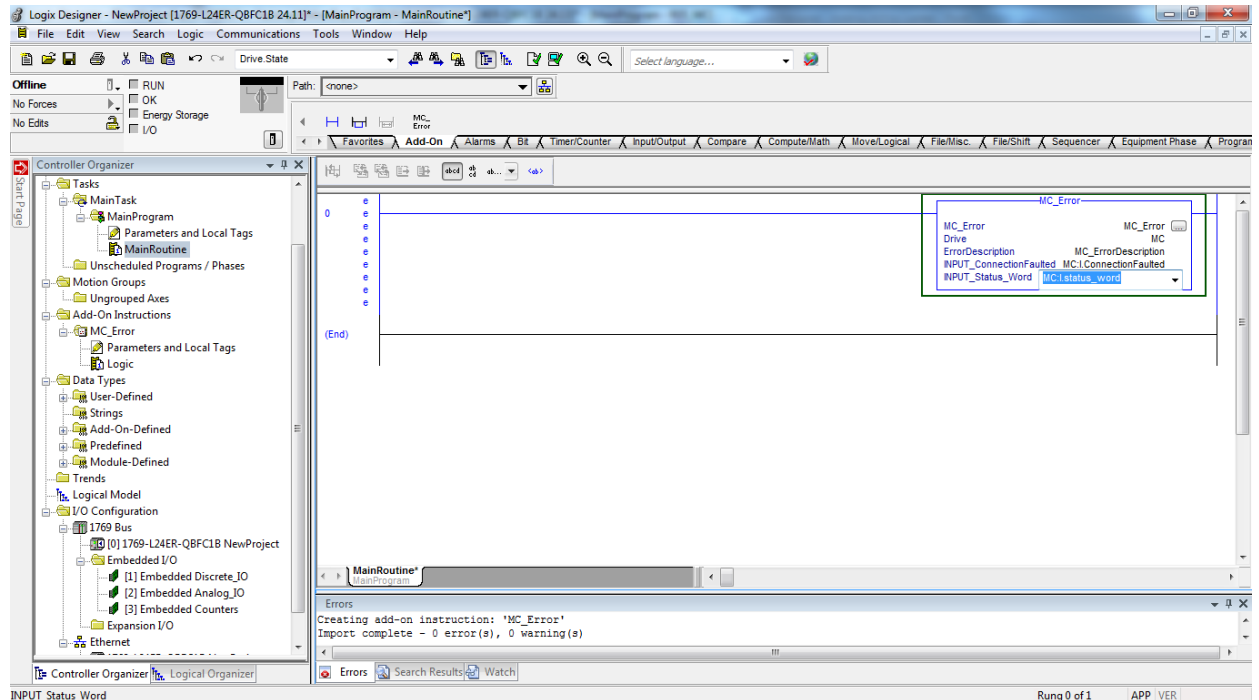


7 Hinzufügen von FBs und Controller Tags in ein Programm

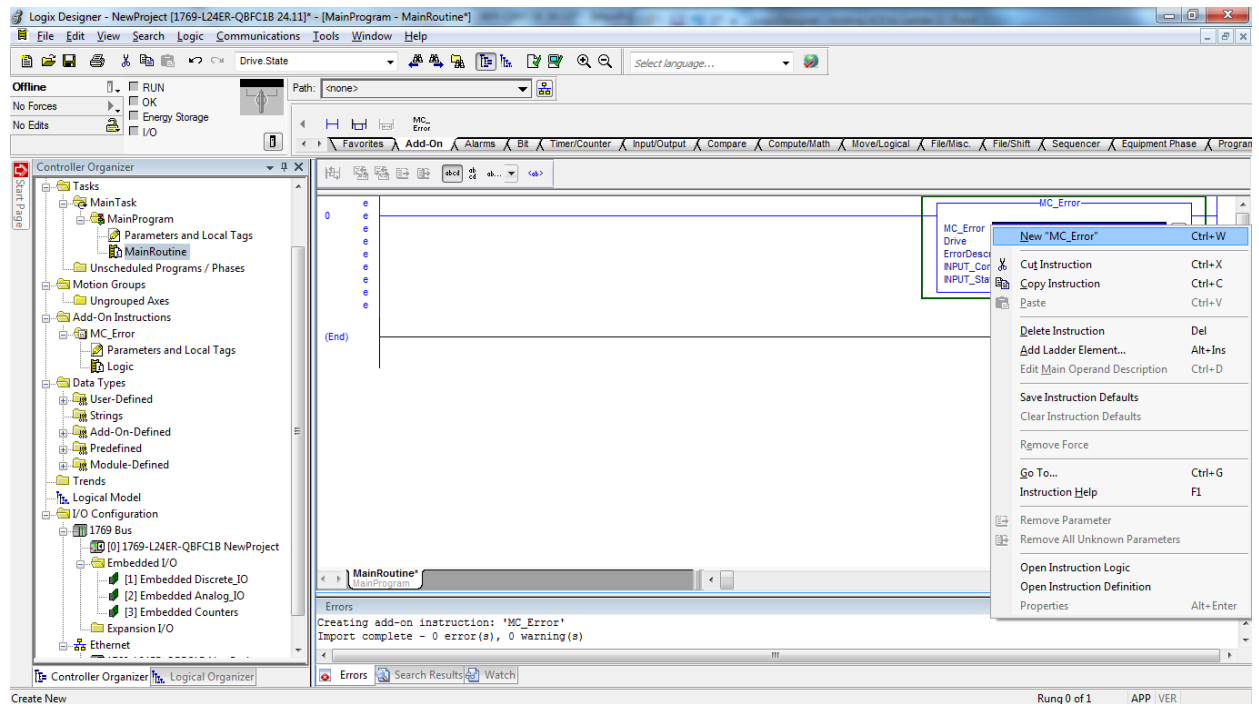
- In der gewünschten Routine innerhalb eines Kontaktplan-Programms auf „Add-On“ klicken und den gewünschten Funktionsblock auswählen (in diesem Beispiel den Funktionsblock „MC_Error“, der den Status des Antriebs und des FBs als Fehlerbit, Fehler-ID „ErrorID“ und als Text ausgibt):



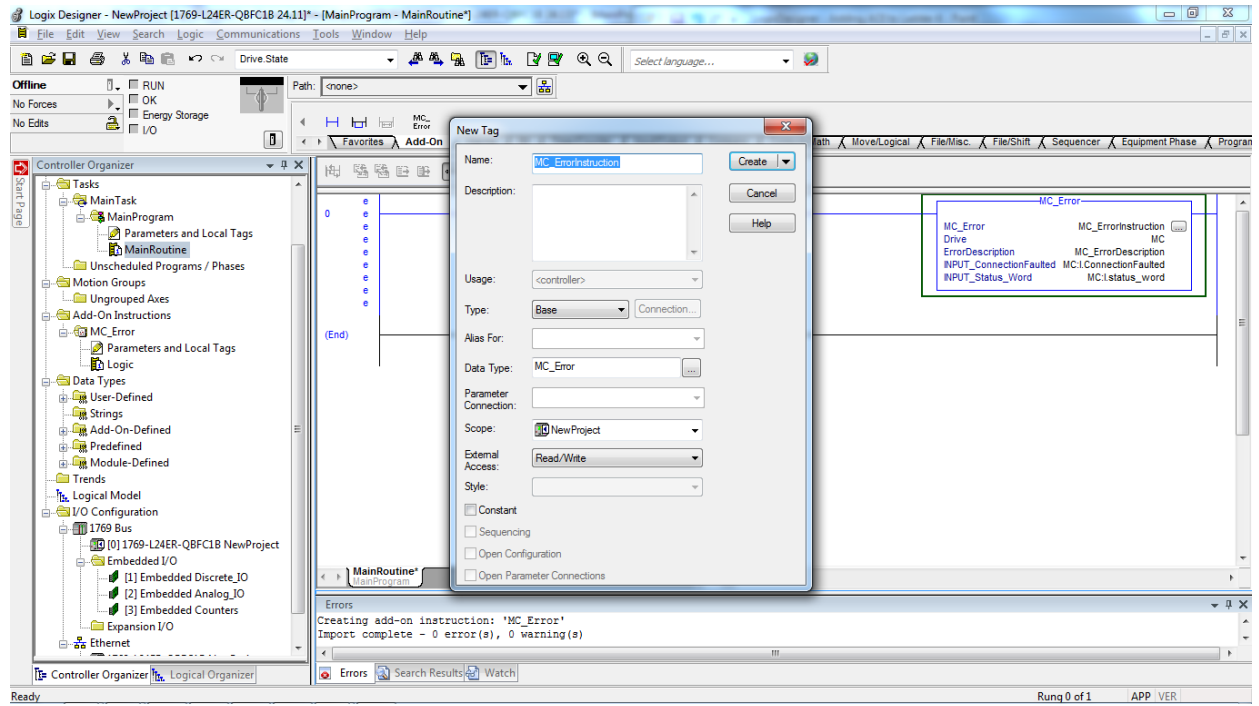
- Nach dem Einfügen des Funktionsblocks in das Kontaktplan-Programm präsentiert sich das Programm folgendermaßen:



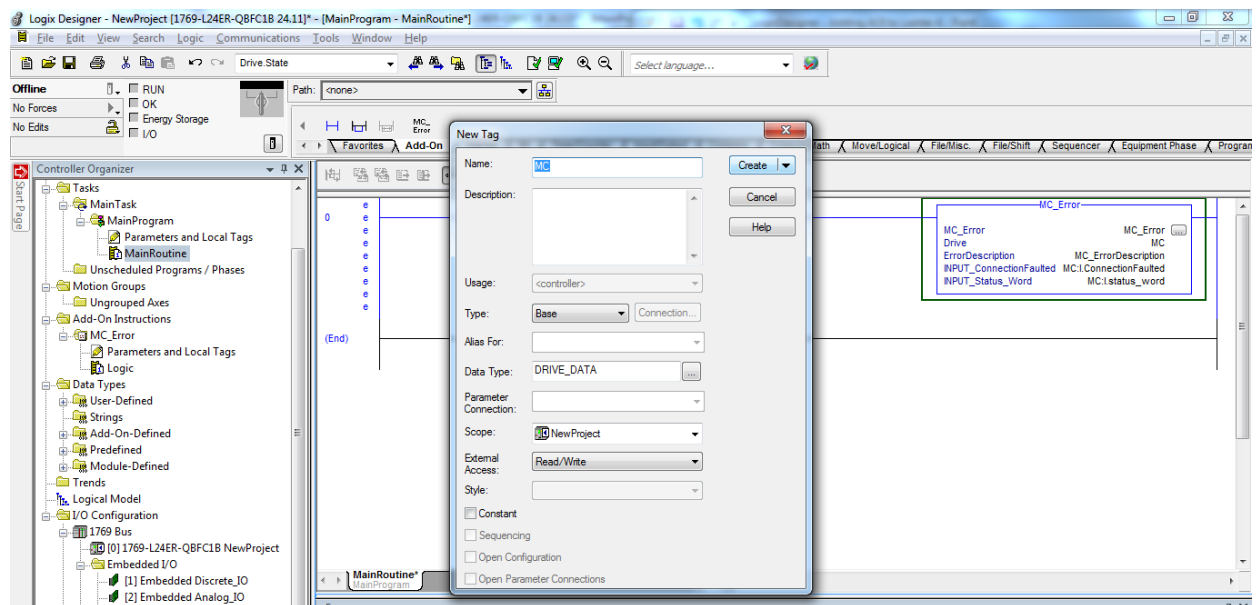
- Um einen Datenbereich für diese Funktionsblock-Instanz anzulegen, mit Rechtsklick auf die Variable „MC_Error“ klicken:



- Dann einen neuen Tag des Typs „MC_Error“ anlegen:



- Dann den gewünschten Antrieb auswählen, auf den die Funktionsblock-Instanz angewendet werden soll (Typ „DRIVE_DATA“).



-
- The screenshot displays the Logix Designer software interface. The main window shows a ladder logic diagram with a network containing several coils and a normally open contact labeled 'MC_Error'. A 'New Tag' dialog box is open in the center, allowing the user to define a new tag. The dialog fields are as follows:
- Name:** MC_ErrorDescription
 - Description:** (empty)
 - Usage:** <controller>
 - Type:** Base
 - Alias For:** (empty)
 - Data Type:** STRING
 - Parameter Connection:** (empty)
 - Scope:** New Project
 - External Access:** Read/Write
 - Style:** (empty)
 - Options:** Constant, Sequencing, Open Configuration, Open Parameter Connections (all unchecked)
- In the background, the 'Controller Organizer' tree on the left shows the project structure, including 'MainTask', 'MainProgram', and 'MainRoutine'. The 'Errors' window at the bottom displays the following message:
- ```

Errors
Creating add-on instruction: 'MC_Error'
Import complete - 0 error(s), 0 warning(s)

```

- 
- The screenshot displays the Logix Designer software interface for a project named "Logix Designer - NewProject [1769-L24ER-QBFC1B 24.11]\* - [MainProgram - MainRoutine\*]".
- Controller Organizer (Left Panel):** Shows the project structure. The "MainRoutine" is selected under "MainProgram". It includes sections for "Parameters and Local Tags", "Unscheduled Programs / Phases", "Motion Groups", "Add-On Instructions", "MC\_Error", "Logic", "Data Types", "User-Defined", "Strings", "Add-On-Defined", "Predefined", "Module-Defined", "Trends", "Logical Model", "I/O Configuration", and "1769 Bus". Under "1769 Bus", there is a sub-entry for "1769-L24ER-QBFC1B NewProject" containing "Embedded I/O" (with [1] Embedded Discrete I/O, [2] Embedded Analog I/O, and [3] Embedded Counters) and "Expansion I/O".
- MainRoutine Ladder Logic (Center):** Shows a single rungs ladder logic diagram. The rung is labeled "MC\_Error" and contains a network of logic. The network starts with a "Drive" input, followed by "ErrorDescription" and "MC\_ErrorDescription". The output is "INPUT\_ConnectionFaulted" and "MC\_11.ConnectionFaulted". The rung is labeled "(End)".
- MC\_Error Add-On Instruction Configuration (Right Panel):** Shows the configuration for the "MC\_Error" instruction. The "Name" field is set to "MC\_11". The "Data Type" is set to "MC\_11.ConnectionFaulted". The "Show controller tags" and "Show MainProgram tags" checkboxes are checked. The "Show parameters from other program:" dropdown is set to "none".
- Errors Panel (Bottom):** Shows a message: "Creating add-on instruction: 'MC\_Error'". Below this, it states "Import complete - 0 error(s), 0 warning(s)".
- Bottom Status Bar:** Displays "Run 0 of 1" and "APP VER".

- Dann die Integer-Variable für das Statuswort auswählen, die mit dem betreffenden Antrieb verknüpft ist:

