YASKAWA

VIPA Library

OPL-LIB | SW90MS0MA | Manual

HB00 | OPL-LIB | SW90MS0MA | en | 17-21

Block library - Simple Motion Control

www.vipa.com/en/service-support/manual

VIPA CONTROLS

VIPA GmbH
Ohmstr. 4
91074 Herzogenaurach
Telephone: +49 9132 744-0
Fax: +49 9132 744-1864
Email: info@vipa.com
Internet: www.vipa.com

SW90MS0MA_000_OPL-LIB,2,EN - © 2017

Table of contents

1 General

1.1 Copyright © VIPA GmbH

All Rights Reserved

Tel.: +49 9132 744 -0

Fax.: +49 9132 744-1864

EMail: info@vipa.de

http://www.vipa.com

Every effort has been made to ensure that the information contained in this document was complete and accurate at the time of publishing. Nevertheless, the authors retain the right to modify the information.

This customer document describes all the hardware units and functions known at the present time. Descriptions may be included for units which are not present at the customer site. The exact scope of delivery is described in the respective purchase contract.

CE Conformity Declaration

Hereby, VIPA GmbH declares that the products and systems are in compliance with the essential requirements and other relevant provisions. Conformity is indicated by the CE marking affixed to the product.

Conformity Information

For more information regarding CE marking and Declaration of Conformity (DoC), please contact your local VIPA customer service organization.

Trademarks

VIPA, SLIO, System 100V, System 200V, System 300V, System 300S, System 400V, System 500S and Commander Compact are registered trademarks of VIPA Gesellschaft für Visualisierung und Prozessautomatisierung mbH.

SPEED7 is a registered trademark of profichip GmbH.

SIMATIC, STEP, SINEC, TIA Portal, S7-300 and S7-400 are registered trademarks of Siemens AG.

Microsoft and Windows are registered trademarks of Microsoft Inc., USA.

Portable Document Format (PDF) and Postscript are registered trademarks of Adobe Systems, Inc.

All other trademarks, logos and service or product marks specified herein are owned by their respective companies.

Information product support

Contact your local VIPA Customer Service Organization representative if you wish to report errors or questions regarding the contents of this document. If you are unable to locate a customer service centre, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Telefax: +49 9132 744-1204

EMail: documentation@vipa.de

Technical support

Contact your local VIPA Customer Service Organization representative if you encounter problems with the product or have questions regarding the product. If you are unable to locate a customer service centre, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Tel.: +49 9132 744-1150 (Hotline)

EMail: support@vipa.de

1.2 About this manual

Objective and contents

This manual describes the *Simple Motion Control Library* of VIPA:

- It contains a description of the structure, project implementation and usage in several programming systems.
- The manual is targeted at users who have a background in automation technology.
- The manual is available in electronic form as PDF file. This requires Adobe Acrobat Reader.
- The manual consists of chapters. Every chapter provides a self-contained description of a specific topic.
- The following guides are available in the manual:
  - An overall table of contents at the beginning of the manual
  - References with pages numbers

Icons Headings

Important passages in the text are highlighted by following icons and headings:

DANGER!
Immediate or likely danger. Personal injury is possible.

CAUTION!
Damages to property is likely if these warnings are not heeded.

Supplementary information and useful tips.

2 Overview

2.1 Simple Motion Control

Properties

With the *Simple Motion Control Library* blocks, you can easily integrate drives into your applications without detailed knowledge. Here various drives and bus systems are supported. The PLCopen blocks enable you to implement simple drive tasks in your control system. This system offers the following features:

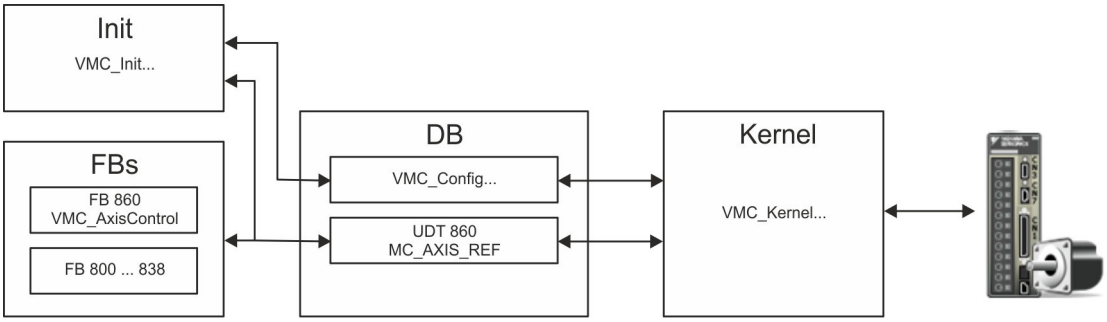
- Can be used in VIPA *SPEED7 Studio* and Siemens SIMATIC Manager
- Implementation of simple drive functions
  - Switch on or off
  - Speed setting
  - Relative or absolute positioning
  - Homing
  - Read and write parameters
  - Query of axis position and status
- Easy commissioning and diagnostics without detailed knowledge of the drives
- Support of various drives and field buses
- Visualization of individual axes
- Scalable by using PLCopen blocks

Structure

The *Simple Motion Control Library* is divided into the following groups:

- Axis Control
  - General blocks for controlling the drives.
- *Sigma-5* EtherCAT
  - Specific building blocks for the use of *Sigma-5* drives, which are connected via EtherCAT.
- *Sigma-7S* EtherCAT
  - Specific building blocks for the use of *Sigma-7* drives, which are connected via EtherCAT.
- *Sigma-7W* EtherCAT
  - Specific building blocks for the use of *Sigma-7W* drives, which are connected via EtherCAT.

2.2 Functional principle



- DB
  - A data block (axis DB) for configuration and status data must be created for each axis of a drive.
  - For the type declaration within the DB, there is a separate data structure (VMC_Config ...) for each drive in the *Simple Motion Control Library*.
- Init
  - The *Init*t block is used to configure an axis.
  - The configuration data for the initialization must be stored in the *axis DB*.
  - For each drive, there is a separate *Init* block (VMC_Init...). in the *Simple Motion Control Library*.
- Kernel
  - The *Kernel* block communicates with the drive via the appropriate bus system, processes the user requests and returns status messages.
  - The exchange of the data takes place by means of the *axis DB*.
  - For each drive, there is a separate *Kernel* block (VMC_Kernel...). in the *Simple Motion Control Library*.
- FBs
  - General block for all drives and bus systems.
  - The exchange of the data takes place by means of the *axis DB*.
  - The FB 860 - VMC_AxisControl is a universal block for simple motion commands and status queries.
  - The blocks FB 800 ... FB 838 are PLCopen blocks for programming complex motion sequences and status queries.

3 Usage *Sigma-5* EtherCAT

3.1 Overview

Precondition

- SPEED7 Studio from V1.6.1
- CPU with EtherCAT master, eg CPU 015-CEFNR00
- *Sigma-5* drive with EtherCAT option card

Steps of configuration

1. Set the parameters on the drive
   - The setting of the parameters happens by means of the software tool *Sigma Win+*.
2. Hardware configuration in VIPA *SPEED7 Studio* or Siemens SIMATIC Manager
   - Configuring a CPU with EtherCAT master functionality.
   - Configuration of a *Sigma-5* EtherCAT drive.
   - Configuring the EtherCAT connection via *SPEED7 EtherCAT Manager*.
3. Programming in VIPA *SPEED7 Studio* or Siemens SIMATIC Manager

   Programming in VIPA *SPEED7 Studio*
   - Connecting the *Init* block to configure the axis.
   - Connecting the *Kernel* block to communicate with the axis.
   - Connecting the blocks for the motion sequences.

3.2 Set the parameters on the drive

CAUTION!
Before the commissioning, you have to adapt your drive to your application with the *Sigma Win+* software tool! More may be found in the manual of your drive.

The following parameters must be set via *Sigma Win+* to match the *Simple Motion Control Library*:

Sigma-5 (20bit encoder)

Servopack Parameter	Address	Name	Value
Pn205	(2205h)	Multiturn Limit Setting	65535
Pn20E	(220Eh)	ElectronicGear Ratio (Numerator)	1
Pn210	(2210h)	Electronic Gear Ratio (Denominator)	1
PnB02	(2701h:01)	Position User Unit (Numerator)	1
PnB04	(2701h:02)	Position User Unit (Denominator)	1
PnB06	(2702h:01)	Velocity User Unit (Numerator)	1
PnB08	(2702h:02)	Velocity User Unit (Denominator)	1
PnB0A	(2703h:01)	Acceleration User Unit (Numerator)	1
PnB0C	(2703h:02)	Acceleration User Unit (Denominator)	1

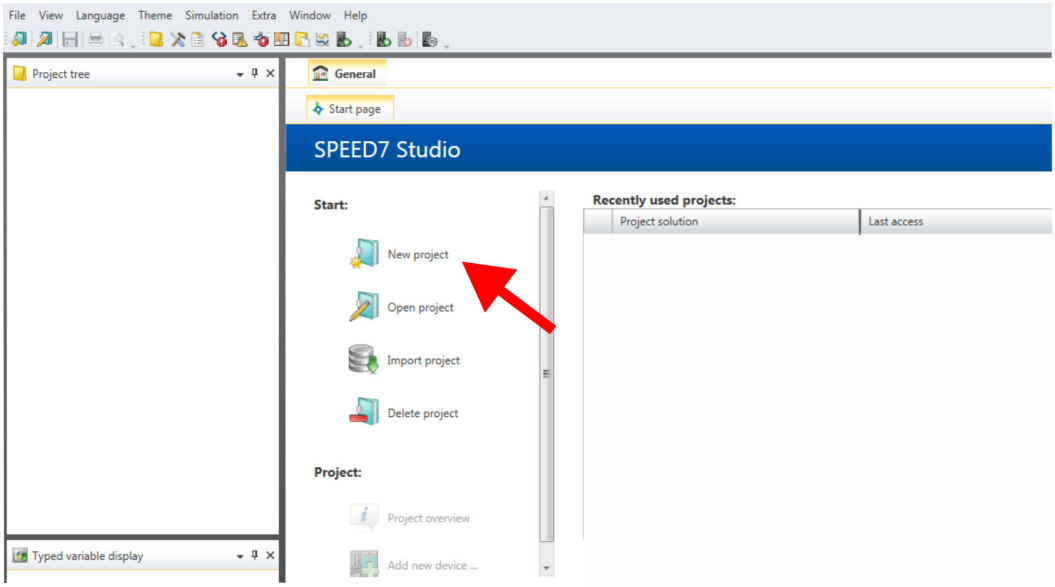
3.3 Usage in VIPA *SPEED7 Studio*

3.3.1 Hardware configuration

Add CPU in the project

Please use for configuration the*SPEED7 Studio* V1.6.1 and up.

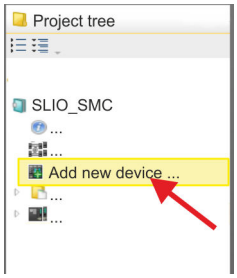
1. Start the *SPEED7 Studio*.

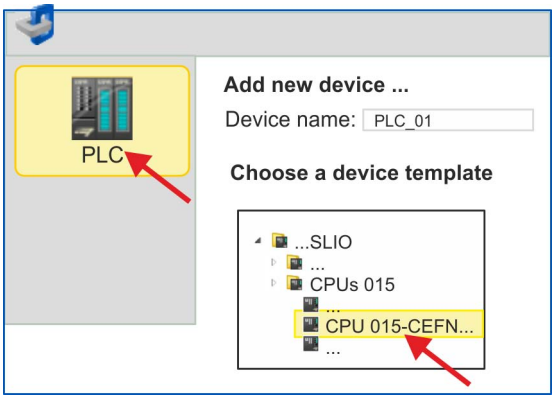


2. Create a new project at the start page with *'New project'*.

⇨ A new project is created and the view *'Devices and networking'* is shown.

3. Click in the *Project tree* at *'Add new device ...'*.



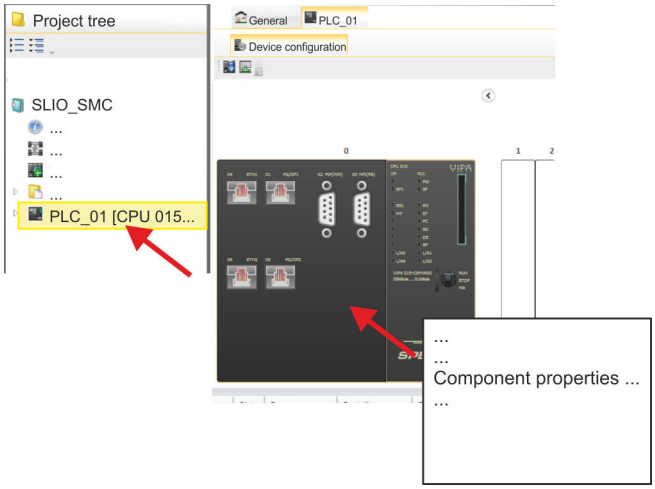


⇨ A dialog for device selection opens.

4. Select from the *'Device templates'* a CPU with EtherCAT master functions such as CPU 015-CEFNR00 and click at [OK].

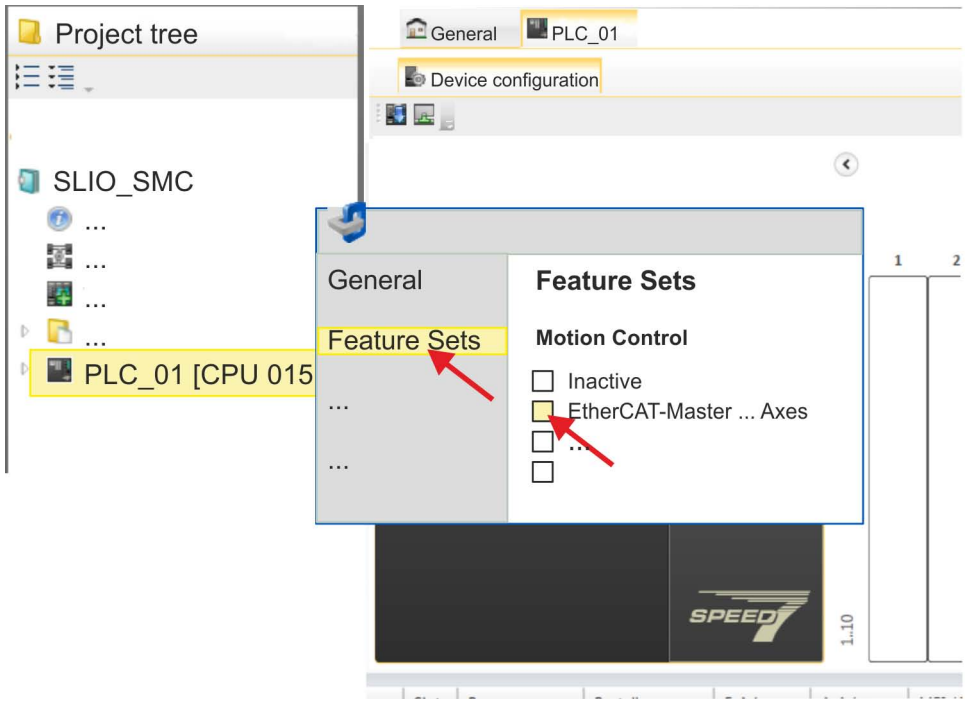
⇨ The CPU is inserted in *'Devices and networking'* and the *'Device configuration'* is opened.

Activate motion control functions



1. Click at the CPU in the *'Device configuration'* and select *'Context menu ➔ Components properties'*.

   ⇨ The properties dialog of the CPU is opened.



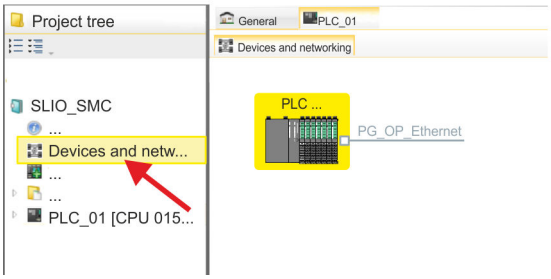
2. Click at *'Feature Sets'* and activate at *'Motion Control'* the parameter *'EtherCAT-Master... Axes'*. The number of axes is not relevant in this example.
3. Confirm your input with [OK].

   ⇨ The motion control functions are now available in your project.

CAUTION!
Please note due to the system, with every change to the feature set settings, the EtherCAT field bus system and its motion control configuration will be deleted from your project!

Configuration of Ethernet PG/OP channel

1. Click in the *Project tree* at *'Devices and networking'*.
   ⇨ You will get a graphical object view of your CPU.



2. Click at the network *'PG_OP_Ethernet'*.
3. Select *'Context menu ➔ Interface properties'*.
   ⇨ A dialog window opens. Here you can enter the IP address data for your Ethernet PG/OP channel. You get valid IP address parameters from your system administrator.
4. Confirm with [OK].
   ⇨ The IP address data are stored in your project listed in *'Devices and networking'* at *'Local components'*.

   After transferring your project your CPU can be accessed via Ethernet PG/OP channel with the set IP address data.

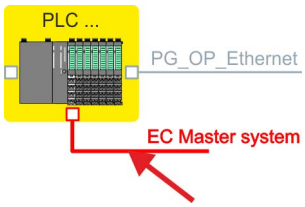
Installing the ESI file

For the *Sigma-5* EtherCAT drive can be configured in the *SPEED7 EtherCAT Manager*, the corresponding ESI file must be installed. Usually, the *SPEED7 Studio* is delivered with current ESI files and you can skip this part. If your ESI file is not up-to date, you will find the latest ESI file for the *Sigma-5* EtherCAT drive under www.yaskawa.eu.com at *'Service ➔ Drives & Motion Software'*.

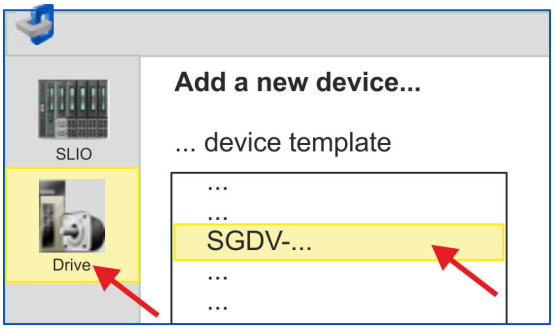
1. Download the according ESI file for your drive. Unzip this if necessary.
2. Navigate to your *SPEED7 Studio*.
3. Open the corresponding dialog window by clicking on *'Extra ➔ Install device description (EtherCAT - ESI)'*.
4. Under *'Source path'*, specify the ESI file and install it with [Install].
   ⇨ The devices of the ESI file are now available.

Add a *Sigma-5* drive

1. Click in the Project tree at *'Devices and networking'*.
2. Click here at *'EC-Mastersystem'* and select *'Context menu ➔ Add new device'*.



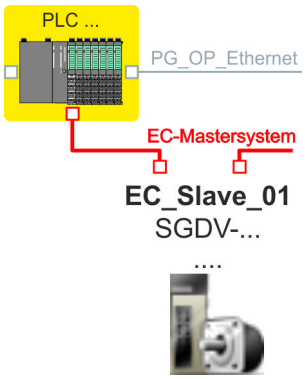
   ⇨ The device template for selecting an EtherCAT device opens.



3. Select your *Sigma-5* drive:

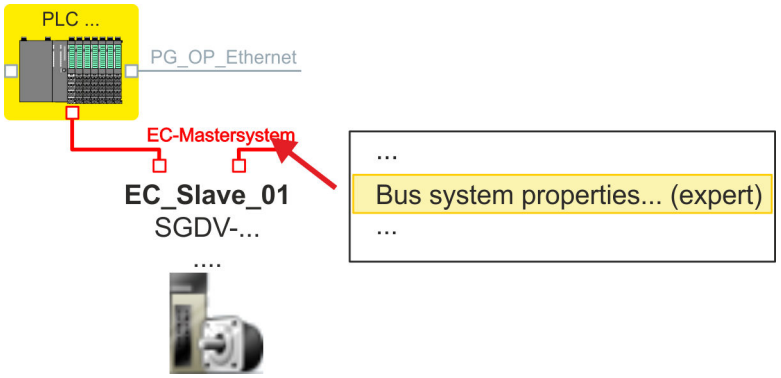
- SGDV-xxxxE5...
- SGDV-xxxxE1...

Confirm with [OK]. If your drive does not exist, you must install the corresponding ESI file as described above.



⇨ The *Sigma-5* drive is connected to your EC-Mastersystem.

Configure *Sigma-5* drive

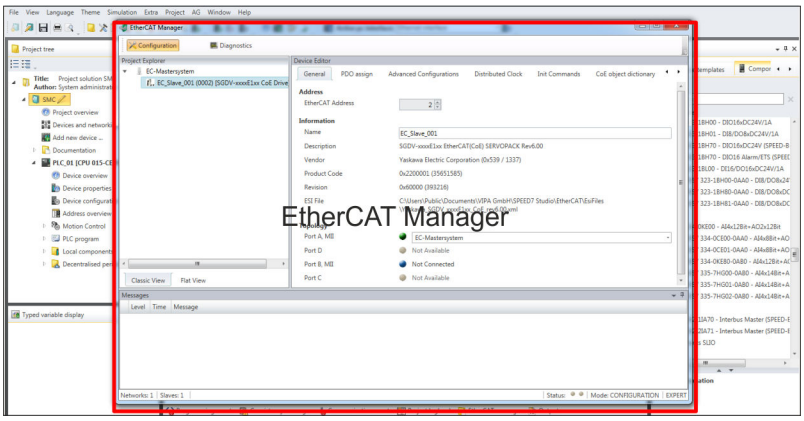


1. Click here at *'EC-Mastersystem'* and select *'Context menu ➔ Bus system properties (expert)'*.

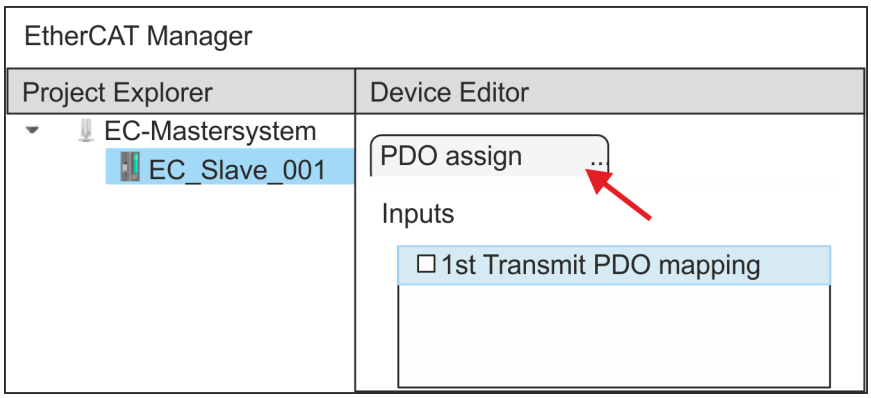
> *You can only edit PDOs in 'Expert mode'! Otherwise, the buttons are hidden.*

⇨ The *SPEED7 EtherCAT Manager* opens. Here you can configure the EtherCAT communication to your *Sigma-5* drive.

More information about the usage of the *SPEED7 EtherCAT Manager* may be found in the online help of the *SPEED7 Studio*.



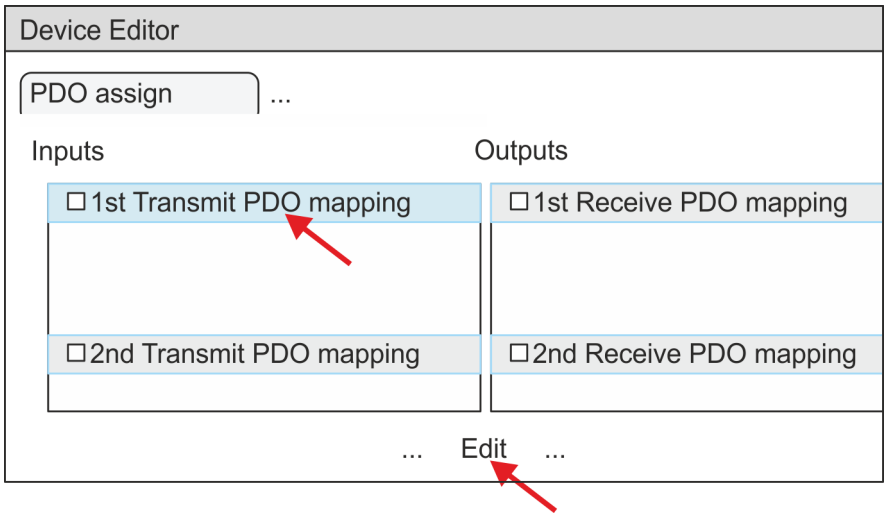
2. Click on the slave in the *SPEED7 EtherCAT Manager* and select the *'PDO assign'* tab in the *'Device editor'*.



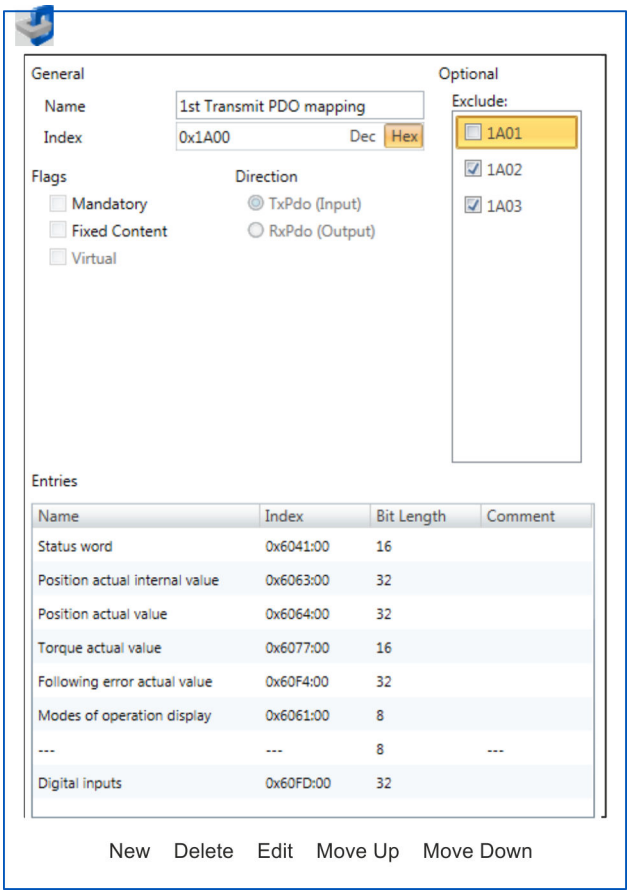
⇨ This dialog shows a list of the PDOs.

3. By selecting the appropriate mapping, you can edit the PDOs with [Edit]. Select the mapping *'1st Transmit PDO mapping'* and click at [Edit].

Please note that some PDOs can not be edited because of the default settings. By de-activating already activated PDOs, you can release the processing of locked PDOs.



⇨ The dialog *'Edit PDO'* is opened. Please check the PDO settings listed here and adjust them if necessary. Please also take into account the order of the *'Entries'* and add them accordingly.



The following functions are available for editing the *'Entries'*:

- New
  - Here you can create a new entry in a dialog by selecting the corresponding entry from the *'CoE object dictionary'* and making your settings. The entry is accepted with [OK] and is listed in the list of entries.
- Delete
  - This allows you to delete a selected entry.
- Edit
  - This allows you to edit the general data of an entry.
- Move Up/Down
  - This allows you to move the selected entry up or down in the list.

4. Perform the following settings:

Inputs: 1st Transmit PDO 0x1A00

- General
  - Name: 1st Transmit PDO mapping
  - Index: 0x1A00
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A01: de-activated
- Entries

Name	Index	Bit length
Status word	0x6041:00	16bit
Position actual internal value	0x6063:00	32bit
Position actual value	0x6064:00	32bit
Torque actual value	0x6077:00	16bit
Following error actual value	0x60F4:00	32bit
Modes of operation display	0x6061:00	8bit
---	---	8bit
Digital inputs	0x60FD:00	32bit

Close the dialog *'Edit PDO'* with [OK].

5. Select the mapping *'2nd Transmit PDO mapping'* and click at [Edit]. Perform the following settings:

Inputs: 2nd Transmit PDO 0x1A01

- General
  - Name: 2nd Transmit PDO mapping
  - Index: 0x1A01
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A00: de-activated
  - 1A02: de-activated
  - 1A03: de-activated
- Entries

Name	Index	Bit length
Touch probe status	0x60B9:00	16bit
Touch probe 1 position value	0x60BA:00	32bit
Touch probe 2 position value	0x60BC:00	32bit
Velocity actual value	0x606C:00	32bit

Close the dialog *'Edit PDO'* with [OK].

6. Select the mapping *'1st Receive PDO mapping'* and click at [Edit]. Perform the following settings:

Outputs: 1st Receive PDO 0x1600

- General
  - Name: 1st Receive PDO mapping
  - Index: 0x1600
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1601: de-activated
  - 1602: de-activated
  - 1603: de-activated
- Entries

Name	Index	Bit length
Control word	0x6040:00	16bit
Target position	0x607A:00	32bit
Target velocity	0x60FF:00	32bit
Modes of operation	0x6060:00	8bit
---	---	8bit
Touch probe function	0x60B8:00	16bit

Close the dialog *'Edit PDO'* with [OK].

7. Select the mapping *'2nd ReceivePDO mapping'* and click at [Edit]. Perform the following settings:

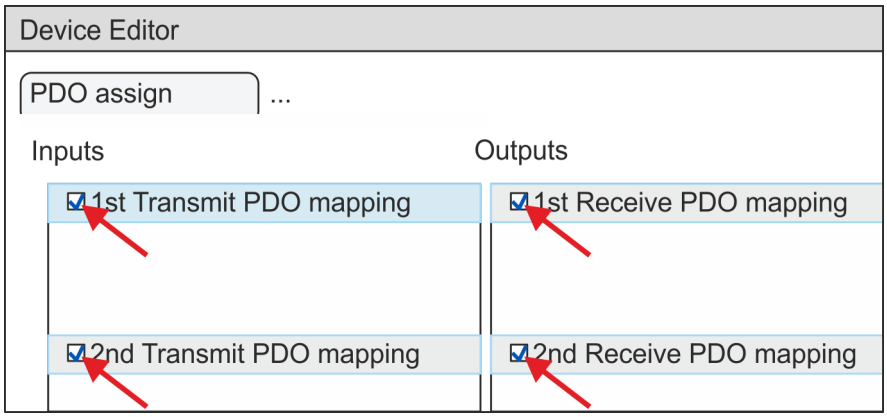
Outputs: 2nd Receive PDO 0x1601

- General
  - Name: 2nd Receive PDO mapping
  - Index: 0x1601
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

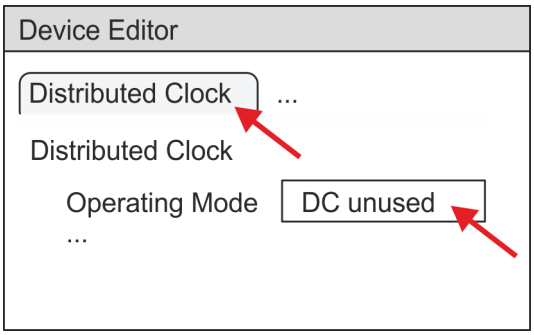
  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1600: de-activated
  - 1602: activated
  - 1603: activated
- Entries
  - Profile velocity: 0x6081:00 → 32 Bit
  - Profile acceleration: 0x6083:00 → 32 Bit
  - Profile deceleration: 0x6084:00 → 32 Bit

Close the dialog *'Edit PDO'* with [OK].

8. In PDO assignment, activate the PDOs 1 and 2 for the inputs and outputs. All subsequent PDOs must remain de-activated. If this is not possible, please check the respective PDO parameter *'Exclude'*.

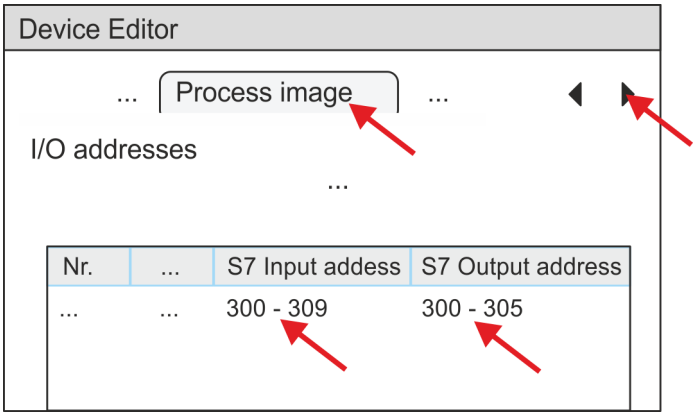


9. In the *'Device Editor'* of the *SPEED7 EtherCAT Manager*, select the *'Distributed clocks'* tab and set *'DC unused'* as *'Operating mode'*.



10. Select the *'Process image'* tab via the arrow key in the *'Device editor'* and note for the parameter of the block FB 871 - VMC_InitSigma5_EC the following PDO.

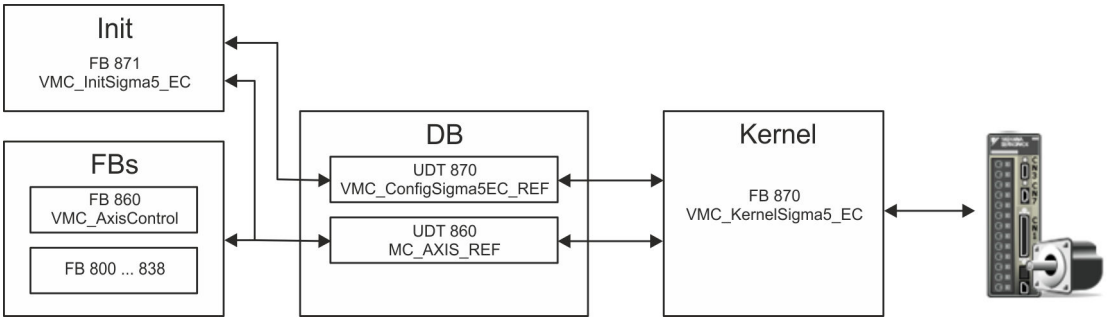
- *'S7 Input address'* → *'InputsStartAddressPDO'*
- *'S7 Output address'* → *'OutputsStartAddressPDO'*



11. By closing the dialog of the *SPEED7 EtherCAT Manager* with [X] the configuration is taken to the *SPEED7 Studio*.

3.3.2 User program

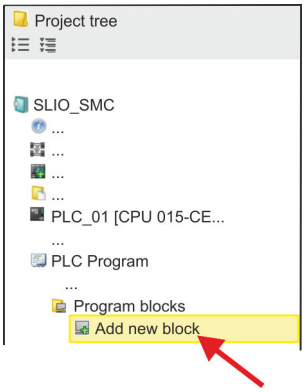
3.3.2.1 Program structure



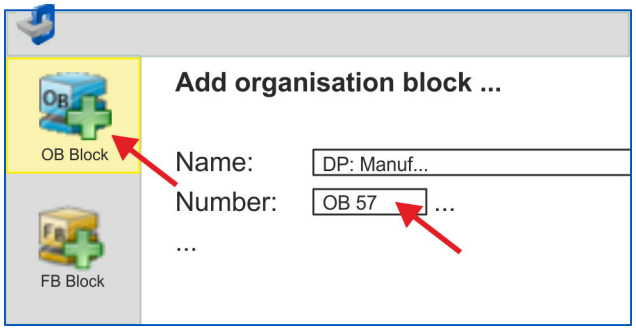
- DB
  A data block (axis DB) for configuration and status data must be created for each axis of a drive. The data block consists of the following data structures:
  – UDT 870 - *VMC_Config*Sigma5EC_REF
    The data structure describes the structure of the configuration of the drive.
    Specific data structure for *Sigma-5* EtherCAT.
  – UDT 860 - *MC_AXIS_REF*
    The data structure describes the structure of the parameters and status information of drives.
    General data structure for all drives and bus systems.
- FB 871 - *VMC_Init*Sigma5_EC
  – The *Init*t block is used to configure an axis.
  – Specific block for *Sigma-5* EtherCAT.
  – The configuration data for the initialization must be stored in the *axis DB*.
- FB 870 - *VMC_Kernel*Sigma5_EC
  – The *Kernel* block communicates with the drive via the appropriate bus system, processes the user requests and returns status messages.
  – Specific block for *Sigma-5* EtherCAT.
  – The exchange of the data takes place by means of the *axis DB*.
- FB 860 - *VMC_AxisControl*
  – General block for all drives and bus systems.
  – Supports simple motion commands and returns all relevant status messages.
  – The exchange of the data takes place by means of the *axis DB*.
  – For motion control and status query, via the instance data of the block you can link a visualization.
  – In addition to the FB 860 - *VMC_AxisControl*, *PLCopen* blocks can be used.
- FB 800 ... FB 838 - *PLCopen*
  – The PLCopen blocks are used to program motion sequences and status queries.
  – General blocks for all drives and bus systems.

3.3.2.2 Programming

Copy blocks into project

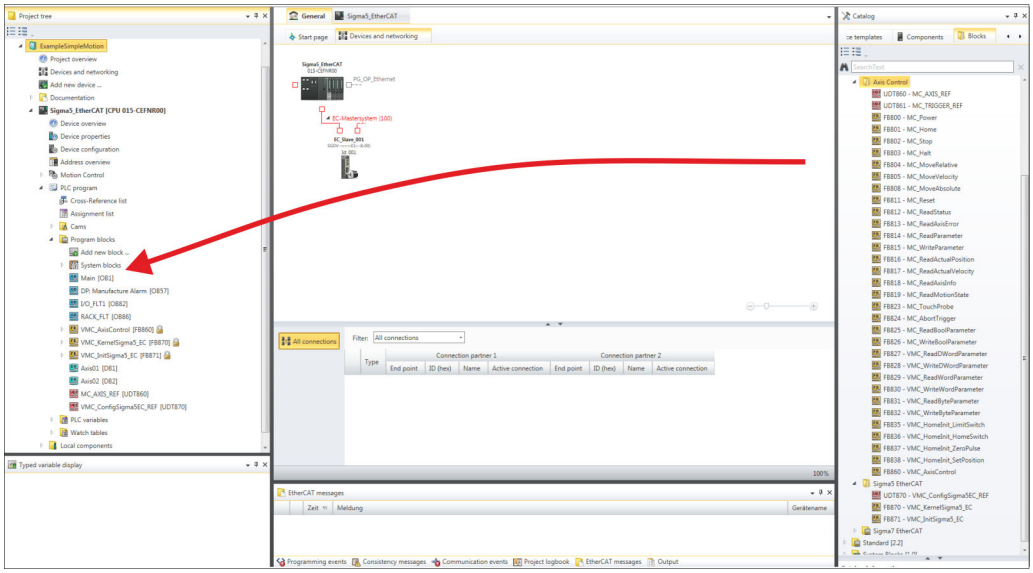


1. Click in the *Project tree* within the CPU at *'PLC program'*, *'Program blocks'* at *'Add New block'*.



⇨ The dialog *'Add block'* is opened.

2. Select the block type *'OB block'* and add OB 57, OB 82 and OB 86 to your project.



3. In the *'Catalog'*, open the *'Simple Motion Control'* library at *'Blocks'* and drag and drop the following blocks into *'Program blocks'* of the *Project tree*:
   - *Sigma-5* EtherCAT:
     - UDT 870 - VMC_ConfigSigma5EC_REF
     - FB 870 - VMC_KernelSigma5_EC
     - FB 871 - VMC_InitSigma5_EC
   - Axis Control
     - UDT 860 - MC_AXIS_REF
     - Blocks for your movement sequences

Create axis DB

1. Add a new DB as your *axis DB* to your project. Click in the *Project tree* within the CPU at *'PLC program'*, *'Program blocks'* at *'Add New block'*, select the block type *'DB block'* and assign the name "Axis01" to it. The DB number can freely be selected such as DB 10.

⇨ The block is created and opened.

2. ■ In "Axis01", create the variable "Config" of type UDT 870. These are specific axis configuration data.

■ In "Axis01", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

Axis01 [DB10]

Data block structure

		Adr...	Name	Data type	...
		...	Config	UDT	[870]
		...	Axis	UDT	[860]

OB 1

Configuration of the axis

Open OB 1 and program the following FB calls with associated DBs:

FB 871 - VMC_InitSigma5_EC, DB 871 *Chapter 3.5.3 'FB 871 - VMC_Init-Sigma5_EC - Sigma-5 EtherCAT initialization' on page 43*

At *InputsStartAddressPDO* respectively *OutputsStartAddressPDO*, enter the address from the *SPEED7 EtherCAT Manager*. *21*

```
⇨ CALL  "VMC_InitSigma5_EC" , "DI_InitSgm5ETC01"
  Enable                :="InitS5EC1_Enable"
  LogicalAddress        :=300
  InputsStartAddressPDO :=300(EtherCAT-Man:S7 Input address)
  OutputsStartAddressPDO:=300(EtherCAT-Man:S7 Output address)
  EncoderType           :=1
  EncoderResolutionBits :=20
  FactorPosition        :=1.048576e+006
  FactorVelocity        :=1.048576e+006
  FactorAcceleration    :=1.048576e+002
  OffsetPosition        :=0.000000e+000
  MaxVelocityApp        :=5.000000e+001
  MaxAccelerationApp    :=1.000000e+002
  MaxDecelerationApp    :=1.000000e+002
  MaxVelocityDrive      :=6.000000e+001
  MaxAccelerationDrive  :=1.500000e+002
  MaxDecelerationDrive  :=1.500000e+002
  MaxPosition           :=1.048500e+003
  MinPosition           :=-1.048514e+003
  EnableMaxPosition     :=TRUE
  EnableMinPosition     :=TRUE
  MinUserPosition       :="InitS5EC1_MinUserPos"
  MaxUserPosition       :="InitS5EC1_MaxUserPos"
  Valid                 :="InitS5EC1_Valid"
  Error                 :="InitS5EC1_Error"
  ErrorID               :="InitS5EC1_ErrorID"
  Config                :="Axis01".Config
  Axis                  :="Axis01".Axis
```

Connecting the Kernel for the axis

The *Kernel* processes the user commands and passes them appropriately processed on to the drive via the respective bus system.

FB 870 - VMC_KernelSigma5_EC, DB 870 *Chapter 3.5.2 'FB 870 - VMC_Kernel-Sigma5_EC - Sigma-5 EtherCAT Kernel' on page 43*

```
⇨       CALL  "VMC_KernelSigma5_EC" , "DI_KernelSgm5ETC01"
         Init  :="KernelS5EC1_Init"
         Config:="Axis01".Config
         Axis  :="Axis01".Axis
```

Connecting the block for motion sequences

For simplicity, the connection of the FB 860 - VMC_AxisControl is to be shown here. This universal block supports simple motion commands and returns status messages. The inputs and outputs can be individually connected. Please specify the reference to the corresponding axis data at *'Axis'* in the *axis DB*.

FB 860 - VMC_AxisControl, DB 860 *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*

```
⇨       CALL  "VMC_AxisControl" , "DI_AxisControl01"
         SourceInputs     :="AxCtrl1_SourceInputs"
         AxisEnable       :="AxCtrl1_AxisEnable"
         AxisReset        :="AxCtrl1_AxisReset"
         HomeExecute      :="AxCtrl1_HomeExecute"
         HomePosition     :="AxCtrl1_HomePosition"
         StopExecute      :="AxCtrl1_StopExecute"
         MvVelocityExecute:="AxCtrl1_MvVelExecute"
         MvRelativeExecute:="AxCtrl1_MvRelExecute"
         MvAbsoluteExecute:="AxCtrl1_MvAbsExecute"
         PositionDistance :="AxCtrl1_PositionDistance"
         Velocity         :="AxCtrl1_Velocity"
         Acceleration     :="AxCtrl1_Acceleration"
         Deceleration     :="AxCtrl1_Deceleration"
         JogPositive      :="AxCtrl1_JogPositive"
         JogNegative      :="AxCtrl1_JogNegative"
         JogVelocity      :="AxCtrl1_JogVelocity"
         JogAcceleration  :="AxCtrl1_JogAcceleration"
         JogDeceleration  :="AxCtrl1_JogDeceleration"
         AxisReady        :="AxCtrl1_AxisReady"
         AxisEnabled      :="AxCtrl1_AxisEnabled"
         AxisError        :="AxCtrl1_AxisError"
         AxisErrorID      :="AxCtrl1_AxisErrorID"
         DriveWarning     :="AxCtrl1_DriveWarning"
         DriveError       :="AxCtrl1_DriveError"
         DriveErrorID     :="AxCtrl1_DriveErrorID"
         IsHomed          :="AxCtrl1_IsHomed"
         ModeOfOperation  :="AxCtrl1_ModeOfOperation"
         PLCopenState     :="AxCtrl1_PLCopenState"
         ActualPosition   :="AxCtrl1_ActualPosition"
         ActualVelocity   :="AxCtrl1_ActualVelocity"
         CmdDone          :="AxCtrl1_CmdDone"
         CmdBusy          :="AxCtrl1_CmdBusy"
         CmdAborted       :="AxCtrl1_CmdAborted"
         CmdError         :="AxCtrl1_CmdError"
         CmdErrorID       :="AxCtrl1_CmdErrorID"
         DirectionPositive:="AxCtrl1_DirectionPos"
         DirectionNegative:="AxCtrl1_DirectionNeg"
         SWLimitMinActive :="AxCtrl1_SWLimitMinActive"
         SWLimitMaxActive :="AxCtrl1_SWLimitMaxActive"
         HWLimitMinActive :="AxCtrl1_HWLimitMinActive"
         HWLimitMaxActive :="AxCtrl1_HWLimitMaxActive"
         Axis             :="Axis01".Axis
```

For complex motion tasks, you can use the PLCopen blocks. Please specify the reference to the corresponding axis data at Axis in the axis DB.

Your project now includes the following blocks:

- OB 1 - Main
- OB 57 - DP Manufacturer Alarm
- OB 82 - I/O_FLT1
- OB 86 - Rack_FLT

- FB 860 - VMC_AxisControl with instance DB
- FB 870 - VMC_KernelSigma5_EC with instance DB
- FB 871 - VMC_InitSigma5_EC with instance DB
- UDT 860 - MC_Axis_REF
- UDT 870 - VMC_ConfigSigma5EC_REF

Sequence of operations

1. Select *'Project ➔ Compile all'* and transfer the project into your CPU. You can find more information on the transfer of your project in the online help of the *SPEED7 Studio*.

   ⇨ You can take your application into operation now.

CAUTION!
Please always observe the safety instructions for your drive, especially during commissioning!

2. Before an axis can be controlled, it must be initialized. To do this, call the *Init* block FB 871 - VMC_InitSigma5_EC with *Enable* = TRUE.

   ⇨ The output *Valid* returns TRUE. In the event of a fault, you can determine the error by evaluating the *ErrorID*.

   You have to call the *Init* block again if you load a new axis DB or you have changed parameters on the *Init* block.

Do not continue until the Init block report an error!

3. Ensure that the *Kernel* block FB 870 - VMC_KernelSigma5_EC is cyclically called. In this way, control signals are transmitted to the drive and status messages are reported.
4. Program your application with the FB 860 - VMC_AxisControl or with the PLCopen blocks.

3.4 Usage in Siemens SIMATIC Manager

3.4.1 Precondition

Overview

- Please use for configuration the Siemens SIMATIC Manager V 5.5 SP2 and up.
- The configuration of the System SLIO CPU happens in the Siemens SIMATIC Manager by means of a virtual PROFINET IO device *'VIPA SLIO CPU'*. The *'VIPA SLIO CPU'* is to be installed in the hardware catalog by means of the GSDML.
- The configuration of the EtherCAT masters happens in the Siemens SIMATIC Manager by means of a virtual PROFINET IO device *'EtherCAT network'*. The *'EtherCAT network'* is to be installed in the hardware catalog by means of the GSDML.
- The *'EtherCAT network'* can be configured with the VIPA Tool *SPEED7 EtherCAT Manager*.
- For the configuration of the drive in the *SPEED7 EtherCAT Manager* the installation of the according ESI file is necessary.

Installing the IO device *'VIPA SLIO System'*

The installation of the PROFINET IO device *'VIPA SLIO CPU'* happens in the hardware catalog with the following approach:

1. Go to the service area of www.vipa.com.
2. Download the configuration file for your CPU from the download area via *'Config files ➔ PROFINET'*.
3. Extract the file into your working directory.
4. Start the Siemens hardware configurator.
5. Close all the projects.
6. Select *'Options ➔ Install new GSD file'*.
7. Navigate to your working directory and install the according GSDML file.
   ⇨ After the installation the according PROFINET IO device can be found at *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA SLIO System'*.

Installing the IO device EtherCAT network

The installation of the PROFINET IO devices *'EtherCAT Network'* happens in the hardware catalog with the following approach:

1. Go to the service area of www.vipa.com
2. Load from the download area at *'Config files ➔ EtherCAT'* the GSDML file for your EtherCAT master.
3. Extract the files into your working directory.
4. Start the Siemens hardware configurator.
5. Close all the projects.
6. Select *'Options ➔ Install new GSD file'*.
7. Navigate to your working directory and install the according GSDML file.
   ⇨ After the installation the *'EtherCAT Network'* can be found at *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA VIPA EtherCAT System'*.

Installing the *SPEED7 EtherCAT Manager*

The configuration of the PROFINET IO device *'EtherCAT Network'* happens by means of the *SPEED7 EtherCAT Manager* from VIPA. This may be found in the service area of www.vipa.com at *'Service/Support ➔ Downloads ➔ SPEED7'*.

The installation happens with the following proceeding:

1. Close the Siemens SIMATIC Manager.
2. Go to the service area of www.vipa.com
3. Load the *SPEED7 EtherCAT Manager* and unzip it on your PC.
4. For installation start the file EtherCATManager_v... .exe.
5. Select the language for the installation.
6. Accept the licensing agreement.
7. Select the installation directory and start the installation.
8. After installation you have to reboot your PC.
   ⇨ The *SPEED7 EtherCAT Manager* is installed and can now be called via the context menu of the Siemens SIMATIC Manager.

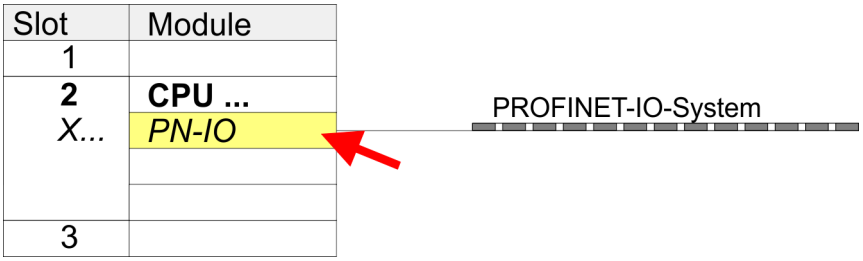
3.4.2 Hardware configuration

Configuring the CPU in the project

Slot	Module
1	
2	**CPU 315-2 PN/DP**
X1	*MPI/DP*
X2	*PN-IO*
X2...	*Port 1*
X2...	*Port 2*
3	

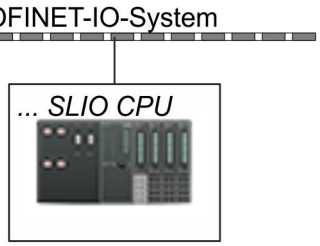
To be compatible with the Siemens SIMATIC Manager the following steps should be executed:

1. Start the Siemens hardware configurator with a new project.
2. Insert a profile rail from the hardware catalog.
3. Place at *'Slot'* number 2 the CPU 315-2 PN/DP (315-2EH14 V3.2).
4. The integrated PROFIBUS DP master (jack X3) is to be configured and connected via the sub module *'X1 MPI/DP'*.
5. The integrated EtherCAT master is to be configured via the sub module *'X2 PN-IO'* as a virtual PROFINET network.
6. Click at the sub module *'PN-IO'* of the CPU.
7. Select *'Context menu ➔ Insert PROFINET IO System'*.



8. Create with [New] a new sub net and assign valid address data
9. Click at the sub module *'PN-IO'* of the CPU and open with *'Context menu ➔ Properties'* the properties dialog.
10. Enter at *'General'* a *'Device name'*. The device name must be unique at the Ethernet subnet.

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	



Slot	Module	Order number	
0	***... SLIO CPU ...***	***015-...***	
X2	*015-...*		
1			
2			
3			
...			

11. Navigate in the hardware catalog to the directory *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA SLIO System'* and connect the IO device *'015-CFFNR00 CPU'* to your PROFINET system.

⇨ In the Device overview of the PROFINET IO device *'VIPA SLIO CPU'* the CPU is already placed at slot 0. From slot 1 you can place your System SLIO modules.

Configuration of Ethernet PG/OP channel

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	
4	343-1EX30
5	
...	

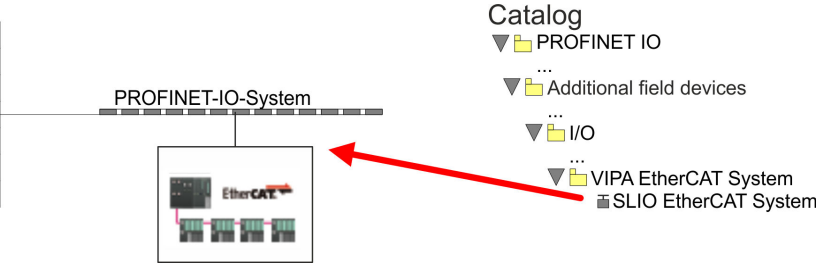
1. Place for the Ethernet PG/OP channel at slot 4 the Siemens CP 343-1 (SIMATIC 300 \ CP 300 \ Industrial Ethernet \CP 343-1 \ 6GK7 343-1EX30 0XE0 V3.0).

2. Open the properties dialog by clicking on the CP 343-1EX30 and enter for the CP at *'Properties'* the IP address data. You get valid IP address parameters from your system administrator.

3. Assign the CP to a *'Subnet'*. The IP address data are not accepted without assignment!

Insert *'EtherCAT network'*

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	

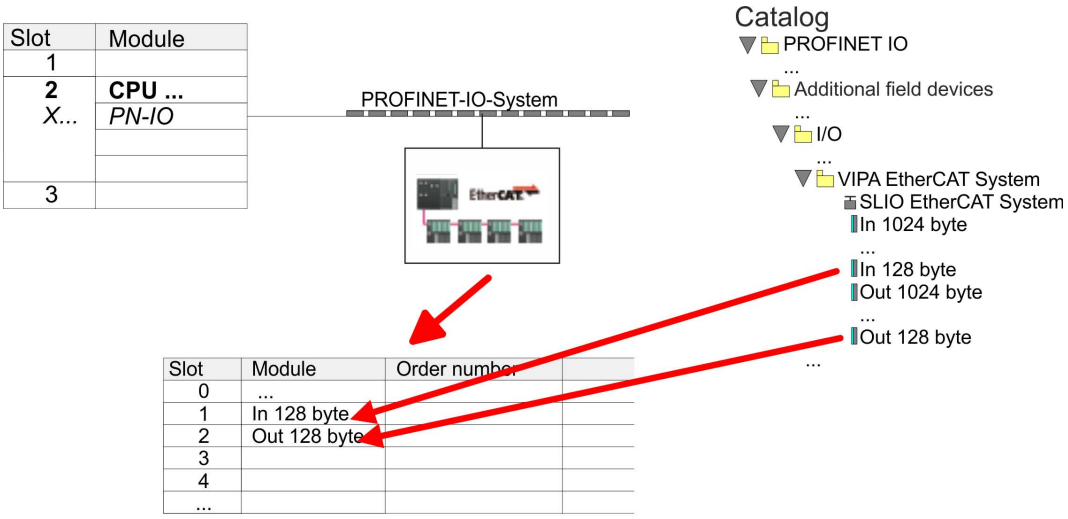


1. Navigate in the hardware catalog to the directory *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA EtherCAT System'* and connect the IO device *'SLIO EtherCAT System'* to your PROFINET system.

2. Click at the inserted IO device *'EtherCAT Network'* and define the areas for in and output by drag and dropping the according *'Out'* or *'In'* area to a slot.

Create the following areas:

- In 128byte
- Out 128byte

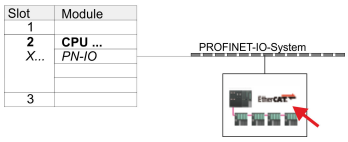


3. Select *'Station ➔ Save and compile'*

Sigma-5 Configure EtherCAT drive

The drive is configured in the *SPEED7 EtherCAT Manager*.

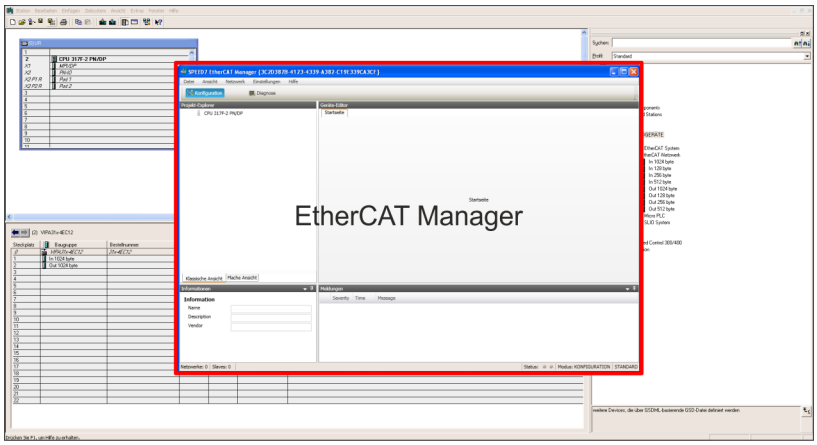
> *Before calling the SPEED7 EtherCAT Manager you have always to save your project with 'Station ➔ Save and compile'.*



1. Click at an inserted IO device *'EtherCAT Network'* and select *'Context menu ➔ Start Device-Tool ➔ SPEED7 EtherCAT Manager'*.

⇨ The *SPEED7 EtherCAT Manager* opens. Here you can configure the EtherCAT communication to your *Sigma-5* drive.

More information about the usage of the *SPEED7 EtherCAT Manager* may be found in the according manual or online help.



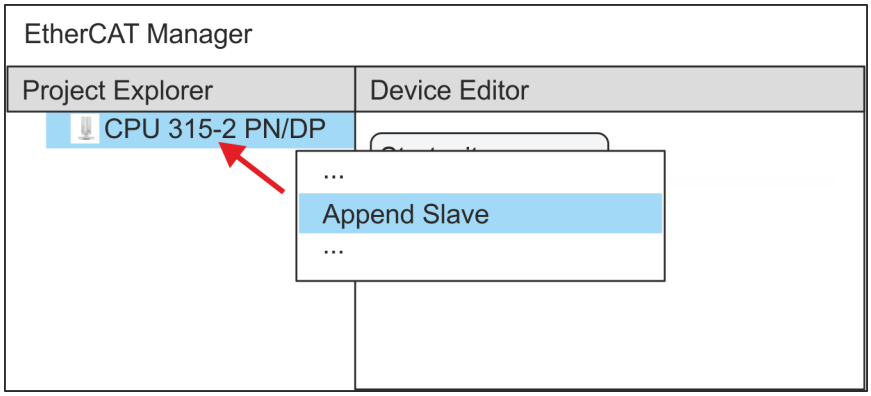
3. For the *Sigma-5* EtherCAT drive to be configured in the *SPEED7 EtherCAT Manager*, the corresponding ESI file must be installed. The ESI file for the *Sigma-5* EtherCAT drive can be found under www.yaskawa.eu.com at *'Service ➔ Drives & Motion Software'*. Download the according ESI file for your drive. Unzip this if necessary.

4. Open in the *SPEED7 EtherCAT Manager* via *'File ➔ ESI Manager'* the dialogue window *'ESI Manager'*.

5. In the *'ESI Manager'* click at [Add File] and select your ESI file. With [Open], the ESI file is installed in the *SPEED7 EtherCAT Manager*.

6. Close the *'ESI Manager'*.

⇨ Your *Sigma-5* EtherCAT drive is now available for configuration.



7. In the EtherCAT Manager, click on your CPU and open via *'Context menu ➔ Append Slave'* the dialog box for adding an EtherCAT slave.

⇨ The dialog window for selecting an EtherCAT slave is opened.

8. Select your *Sigma-5* EtherCAT drive and confirm your selection with [OK].

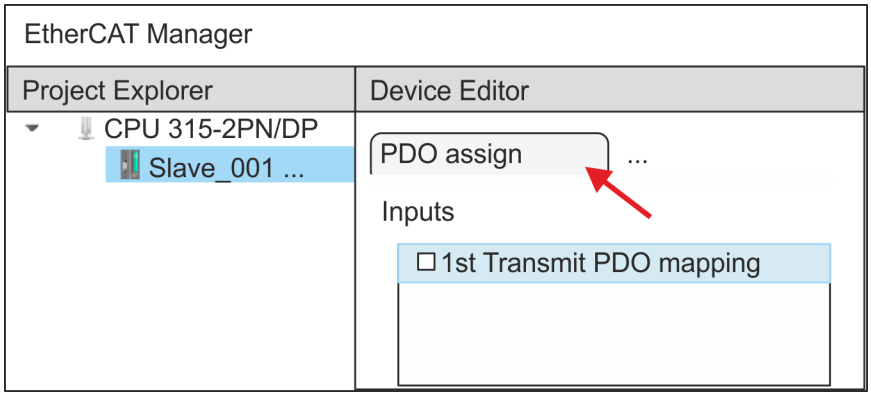
⇨ The *Sigma-5* EtherCAT drive is connected to the master and can now be configured.

9.

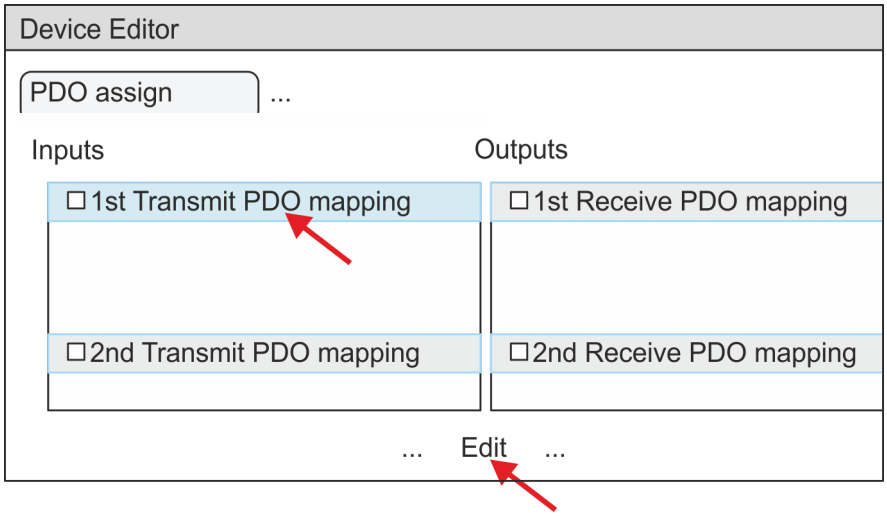
You can only edit PDOs in 'Expert mode'! Otherwise, the buttons are hidden. By activating the 'Expert mode' you can switch to advanced setting.

By activating *'View ➔ Expert'* you can switch to the *Expert mode*.

10. Click on the *Sigma-5* EtherCAT Slave in the *SPEED7 EtherCAT Manager* and select the *'PDO assign'* tab in the *'Device editor'*.

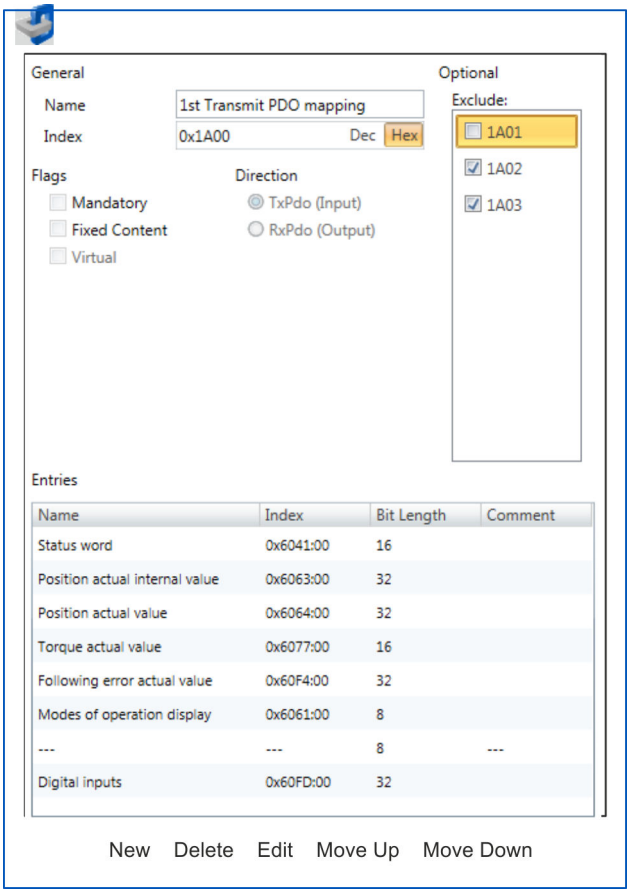


⇨ This dialog shows a list of the PDOs.



11. By selecting the appropriate PDO mapping, you can edit the PDOs with [Edit]. Select the mapping *'1st Transmit PDO mapping'* and click at [Edit].

Please note that some PDOs can not be edited because of the default settings. By de-activating already activated PDOs, you can release the processing of locked PDOs.



⇨ The dialog *'Edit PDO'* is opened. Please check the PDO settings listed here and adjust them if necessary. Please also take into account the order of the *'Entries'* and add them accordingly.

The following functions are available for editing the *'Entries'*:

- New
  - Here you can create a new entry in a dialog by selecting the corresponding entry from the *'CoE object dictionary'* and making your settings. The entry is accepted with [OK] and is listed in the list of entries.
- Delete
  - This allows you to delete a selected entry.
- Edit
  - This allows you to edit the general data of an entry.
- Move Up/Down
  - This allows you to move the selected entry up or down in the list.

12. Perform the following settings:

Inputs: 1st Transmit PDO 0x1A00

- General
  - Name: 1st Transmit PDO mapping
  - Index: 0x1A00
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A01: de-activated
- Entries

Name	Index	Bit length
Status word	0x6041:00	16bit
Position actual internal value	0x6063:00	32bit
Position actual value	0x6064:00	32bit
Torque actual value	0x6077:00	16bit
Following error actual value	0x60F4:00	32bit
Modes of operation display	0x6061:00	8bit
---	---	8bit
Digital inputs	0x60FD:00	32bit

Close the dialog *'Edit PDO'* with [OK].

13. Select the mapping *'2nd Transmit PDO mapping'* and click at [Edit]. Perform the following settings:

Inputs: 2nd Transmit PDO 0x1A01

- General
  - Name: 2nd Transmit PDO mapping
  - Index: 0x1A01
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A00: de-activated
  - 1A02: de-activated
  - 1A03: de-activated
- Entries

Name	Index	Bit length
Touch probe status	0x60B9:00	16bit
Touch probe 1 position value	0x60BA:00	32bit
Touch probe 2 position value	0x60BC:00	32bit
Velocity actual value	0x606C:00	32bit

Close the dialog *'Edit PDO'* with [OK].

14. Select the mapping *'1st Receive PDO mapping'* and click at [Edit]. Perform the following settings:

Outputs: 1st Receive PDO 0x1600

- General
  - Name: 1st Receive PDO mapping
  - Index: 0x1600
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1601: de-activated
  - 1602: de-activated
  - 1603: de-activated
- Entries

Name	Index	Bit length
Control word	0x6040:00	16bit
Target position	0x607A:00	32bit
Target velocity	0x60FF:00	32bit
Modes of operation	0x6060:00	8bit
---	---	8bit
Touch probe function	0x60B8:00	16bit

Close the dialog *'Edit PDO'* with [OK].

15. Select the mapping *'2nd ReceivePDO mapping'* and click at [Edit]. Perform the following settings:

Outputs: 2nd Receive PDO 0x1601

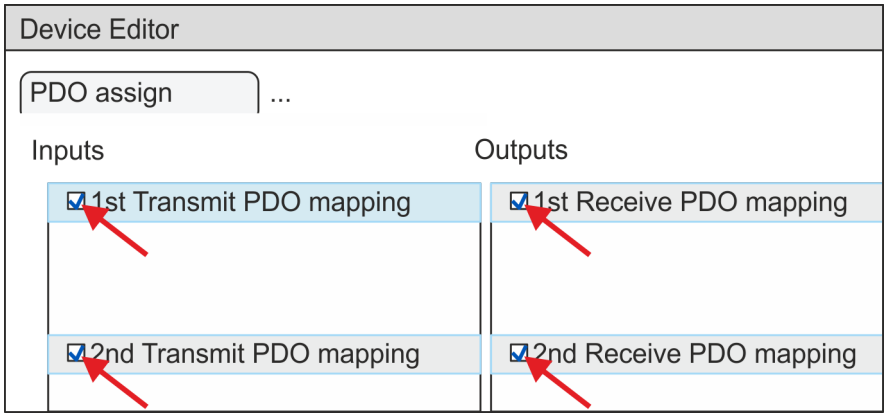
- General
  - Name: 2nd Receive PDO mapping
  - Index: 0x1601
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1600: de-activated
  - 1602: activated
  - 1603: activated
- Entries

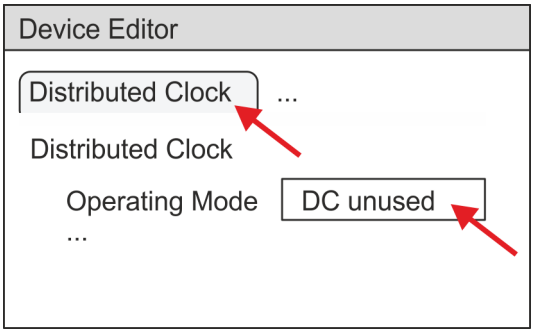
Name	Index	Bit length
Profile velocity	0x6081:00	32bit
Profile acceleration	0x6083:00	32bit
Profile deceleration	0x6084:00	32bit

Close the dialog *'Edit PDO'* with [OK].

16. In PDO assignment, activate the PDOs 1 and 2 for the inputs and outputs. All subsequent PDOs must remain de-activated. If this is not possible, please check the respective PDO parameter *'Exclude'*.

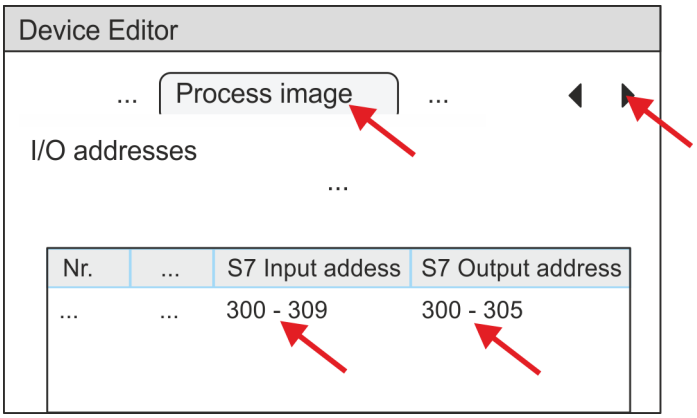


17. In the *'Device Editor'* of the *SPEED7 EtherCAT Manager*, select the *'Distributed clocks'* tab and set *'DC unused'* as *'Operating mode'*.



18. Select the *'Process image'* tab via the arrow key in the *'Device editor'* and note for the parameter of the block FB 871 - VMC_InitSigma5_EC the following PDO.

- *'S7 Input address'* → *'InputsStartAddressPDO'*
- *'S7 Output address'* → *'OutputsStartAddressPDO'*

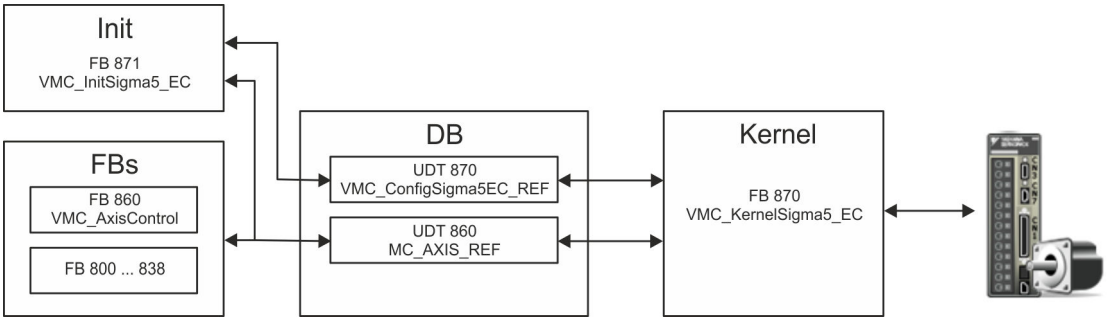


19. By closing the *SPEED7 EtherCAT Manager* with [X] the configuration is taken to the project. You can always edit your EtherCAT configuration in the *SPEED7 EtherCAT Manager*, since the configuration is stored in your project.

20. Save and compile your configuration

3.4.3 User program

3.4.3.1 Program structure



- DB
  A data block (axis DB) for configuration and status data must be created for each axis of a drive. The data block consists of the following data structures:
  – UDT 870 - *VMC_Config*Sigma5EC_REF
    The data structure describes the structure of the configuration of the drive.
    Specific data structure for *Sigma-5* EtherCAT.
  – UDT 860 - *MC_AXIS_REF*
    The data structure describes the structure of the parameters and status information of drives.
    General data structure for all drives and bus systems.
- FB 871 - *VMC_Init*Sigma5_EC
  – The *Init*t block is used to configure an axis.
  – Specific block for *Sigma-5* EtherCAT.
  – The configuration data for the initialization must be stored in the *axis DB*.

- FB 870 - *VMC_Kernel*Sigma5_EC
  - The *Kernel* block communicates with the drive via the appropriate bus system, processes the user requests and returns status messages.
  - Specific block for *Sigma-5* EtherCAT.
  - The exchange of the data takes place by means of the *axis DB*.
- FB 860 - *VMC_AxisControl*
  - General block for all drives and bus systems.
  - Supports simple motion commands and returns all relevant status messages.
  - The exchange of the data takes place by means of the *axis DB*.
  - For motion control and status query, via the instance data of the block you can link a visualization.
  - In addition to the FB 860 - *VMC_AxisControl*, *PLCopen* blocks can be used.
- FB 800 ... FB 838 - *PLCopen*
  - The PLCopen blocks are used to program motion sequences and status queries.
  - General blocks for all drives and bus systems.

3.4.3.2 Programming

Include library

1. Go to the service area of www.vipa.com.
2. Download the *Simple Motion Control* library from the download area at *'VIPA Lib'*.
3. Open the dialog window for ZIP file selection via *'File ➔ Retrieve'*.
4. Select the according ZIP file and click at [Open].
5. Specify a target directory in which the blocks are to be stored and start the unzip process with [OK].

Copy blocks into project

→ Open the library after unzipping and drag and drop the following blocks into *'Blocks'* of your project:

- *Sigma-5* EtherCAT:
  - UDT 870 - VMC_ConfigSigma5EC_REF
  - FB 870 - VMC_KernelSigma5_EC
  - FB 871 - VMC_InitSigma5_EC
- Axis Control
  - UDT 860 - MC_AXIS_REF
  - Blocks for your movement sequences

Create interrupt OBs

1. In your project, click at *'Blocks'* and choose *'Context menu ➔ Insert new object ➔ Organization block'*.

   ⇨ The dialog *'Properties Organization block'* opens.
2. Add OB 57, OB 82, and OB 86 successively to your project.

Create axis DB

1. In your project, click at *'Blocks'* and choose *'Context menu ➔ Insert new object ➔ Data block'*.

   Specify the following parameters:

   - Name and type
     - The DB no. as *'Name'* can freely be chosen, such as DB 10.
     - Set *'Shared DB'* as the *'Type'*.
   - Symbolic name
     - Specify "Axis01".

   Confirm your input with [OK].

   ⇨ The block is created.

2. Open DB 10 "Axis01" by double-click.

   - In "Axis01", create the variable "Config" of type UDT 870. These are specific axis configuration data.
   - In "Axis01", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

   DB10

Address	Name	Typ	...
		Struct	
...	Config	"VMC_ConfigSigma5EC_REF"	
...	Axis	"MC_AXIS_REF	
...		END_STRUCT	

OB 1

Configuration of the axis

Open OB 1 and program the following FB calls with associated DBs:

FB 871 - VMC_InitSigma5_EC, DB 871 *Chapter 3.5.3 'FB 871 - VMC_InitSigma5_EC - Sigma-5 EtherCAT initialization' on page 43*

At *InputsStartAddressPDO* respectively *OutputsStartAddressPDO*, enter the address from the *SPEED7 EtherCAT Manager*. *36*

```
⇨ CALL  "VMC_InitSigma5_EC" , "DI_InitSgm5ETC01"
  Enable                :="InitS5EC1_Enable"
  LogicalAddress        :=300
  InputsStartAddressPDO :=300 (EtherCAT-Man.: S7 Input
  address)
  OutputsStartAddressPDO:=300 (EtherCAT-Man.: S7 Output
  address)
  EncoderType           :=1
  EncoderResolutionBits :=20
  FactorPosition        :=1.048576e+006
  FactorVelocity        :=1.048576e+006
  FactorAcceleration    :=1.048576e+002
  OffsetPosition        :=0.000000e+000
  MaxVelocityApp        :=5.000000e+001
  MaxAccelerationApp    :=1.000000e+002
  MaxDecelerationApp    :=1.000000e+002
  MaxVelocityDrive      :=6.000000e+001
  MaxAccelerationDrive  :=1.500000e+002
  MaxDecelerationDrive  :=1.500000e+002
  MaxPosition           :=1.048500e+003
  MinPosition           :=-1.048514e+003
  EnableMaxPosition     :=TRUE
  EnableMinPosition     :=TRUE
  MinUserPosition       :="InitS5EC1_MinUserPos"
  MaxUserPosition       :="InitS5EC1_MaxUserPos"
  Valid                 :="InitS5EC1_Valid"
  Error                 :="InitS5EC1_Error"
  ErrorID               :="InitS5EC1_ErrorID"
  Config                :="Axis01".Config
  Axis                  :="Axis01".Axis
```

Connecting the Kernel for the axis

The *Kernel* processes the user commands and passes them appropriately processed on to the drive via the respective bus system.

FB 870 - VMC_KernelSigma5_EC, DB 870 *Chapter 3.5.2 'FB 870 - VMC_KernelSigma5_EC - Sigma-5 EtherCAT Kernel' on page 43*

```
⇨       CALL  "VMC_KernelSigma5_EC" , "DI_KernelSgm5ETC01"
         Init  :="KernelS5EC1_Init"
         Config:="Axis01".Config
         Axis  :="Axis01".Axis
```

Connecting the block for motion sequences

For simplicity, the connection of the FB 860 - VMC_AxisControl is to be shown here. This universal block supports simple motion commands and returns status messages. The inputs and outputs can be individually connected. Please specify the reference to the corresponding axis data at *'Axis'* in the *axis DB*.

FB 860 - VMC_AxisControl, DB 860 *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*

```
⇨       CALL  "VMC_AxisControl" , "DI_AxisControl01"
         SourceInputs     :="AxCtrl1_SourceInputs"
         AxisEnable       :="AxCtrl1_AxisEnable"
         AxisReset        :="AxCtrl1_AxisReset"
         HomeExecute      :="AxCtrl1_HomeExecute"
         HomePosition     :="AxCtrl1_HomePosition"
         StopExecute      :="AxCtrl1_StopExecute"
         MvVelocityExecute:="AxCtrl1_MvVelExecute"
         MvRelativeExecute:="AxCtrl1_MvRelExecute"
         MvAbsoluteExecute:="AxCtrl1_MvAbsExecute"
         PositionDistance :="AxCtrl1_PositionDistance"
         Velocity         :="AxCtrl1_Velocity"
         Acceleration     :="AxCtrl1_Acceleration"
         Deceleration     :="AxCtrl1_Deceleration"
         JogPositive      :="AxCtrl1_JogPositive"
         JogNegative      :="AxCtrl1_JogNegative"
         JogVelocity      :="AxCtrl1_JogVelocity"
         JogAcceleration  :="AxCtrl1_JogAcceleration"
         JogDeceleration  :="AxCtrl1_JogDeceleration"
         AxisReady        :="AxCtrl1_AxisReady"
         AxisEnabled      :="AxCtrl1_AxisEnabled"
         AxisError        :="AxCtrl1_AxisError"
         AxisErrorID      :="AxCtrl1_AxisErrorID"
         DriveWarning     :="AxCtrl1_DriveWarning"
         DriveError       :="AxCtrl1_DriveError"
         DriveErrorID     :="AxCtrl1_DriveErrorID"
         IsHomed          :="AxCtrl1_IsHomed"
         ModeOfOperation  :="AxCtrl1_ModeOfOperation"
         PLCopenState     :="AxCtrl1_PLCopenState"
         ActualPosition   :="AxCtrl1_ActualPosition"
         ActualVelocity   :="AxCtrl1_ActualVelocity"
         CmdDone          :="AxCtrl1_CmdDone"
         CmdBusy          :="AxCtrl1_CmdBusy"
         CmdAborted       :="AxCtrl1_CmdAborted"
         CmdError         :="AxCtrl1_CmdError"
         CmdErrorID       :="AxCtrl1_CmdErrorID"
         DirectionPositive:="AxCtrl1_DirectionPos"
         DirectionNegative:="AxCtrl1_DirectionNeg"
         SWLimitMinActive :="AxCtrl1_SWLimitMinActive"
         SWLimitMaxActive :="AxCtrl1_SWLimitMaxActive"
         HWLimitMinActive :="AxCtrl1_HWLimitMinActive"
         HWLimitMaxActive :="AxCtrl1_HWLimitMaxActive"
         Axis             :="Axis01".Axis
```

For complex motion tasks, you can use the PLCopen blocks. Please specify the reference to the corresponding axis data at Axis in the axis DB.

Your project now includes the following blocks:

- OB 1 - Main
- OB 57 - DP Manufacturer Alarm
- OB 82 - I/O_FLT1
- OB 86 - Rack_FLT

- FB 860 - VMC_AxisControl with instance DB
- FB 870 - VMC_KernelSigma5_EC with instance DB
- FB 871 - VMC_InitSigma5_EC with instance DB
- UDT 860 - MC_Axis_REF
- UDT 870 - VMC_ConfigSigma5EC_REF

Sequence of operations

1. Choose the Siemens SIMATIC Manager and transfer your project into the CPU.

   The transfer can only be done by the Siemens SIMATIC Manager - not hardware configurator!

   Since slave and module parameters are transmitted by means of SDO respectively SDO Init command, the configuration remains active, until a power cycle is performed or new parameters for the same SDO objects are transferred.

   With an overall reset the slave and module parameters are not reset!

   ⇨ You can take your application into operation now.

   CAUTION!
   Please always observe the safety instructions for your drive, especially during commissioning!

2. Before an axis can be controlled, it must be initialized. To do this, call the *Init* block FB 871 - VMC_InitSigma5_EC with *Enable* = TRUE.

   ⇨ The output *Valid* returns TRUE. In the event of a fault, you can determine the error by evaluating the *ErrorID*.

   You have to call the *Init* block again if you load a new axis DB or you have changed parameters on the *Init* block.

   Do not continue until the Init block report an error!

3. Ensure that the *Kernel* block FB 870 - VMC_KernelSigma5_EC is cyclically called. In this way, control signals are transmitted to the drive and status messages are reported.
4. Program your application with the FB 860 - VMC_AxisControl or with the PLCopen blocks.

3.4.4 Copy project

Proceeding

In the example, the station *'Source'* is copied and saved as *'Target '*.

1. Open the hardware configuration of the *'Source'* CPU and start the *SPEED7 EtherCAT Manager*.
2. In the *SPEED7 EtherCAT Manager*, via *'File ➔ Save as'* save the configuration in your working directory.
3. Close the *SPEED7 EtherCAT Manager* and the hardware configurator.
4. Copy the station *'Source'* with Ctrl + C and paste it as *'Target'* into your project with Ctrl + V.

5. Select the *'Blocks'* directory of the *'Target'* CPU and delete the *'System data'*.
6. Open the hardware configuration of the *'Target'* CPU. Adapt the IP address data or re-network the CPU or the CP again.

Before calling the SPEED7 EtherCAT Manager you have always to save your project with 'Station ➔ Save and compile'.

7. Safe your project with *'Station ➔ Safe and compile'*.
8. Open the *SPEED7 EtherCAT Manager*.
9. Use *'File ➔ Open'* to load the configuration from your working directory.
10. Close the *SPEED7 EtherCAT Manager*.
11. Save and compile your configuration.

3.5 Drive specific blocks

3.5.1 UDT 870 - VMC_ConfigSigma5EC_REF - *Sigma-5* EtherCAT Data structure axis configuration

This is a user-defined data structure that contains information about the configuration data. The UDT is specially adapted to the use of a *Sigma-5* drive, which is connected via EtherCAT.

3.5.2 FB 870 - VMC_KernelSigma5_EC - *Sigma-5* EtherCAT Kernel

Description

This block converts the drive commands for a *Sigma-5* axis via EtherCAT and communicates with the drive. For each *Sigma-5* axis, an instance of this FB is to be cyclically called.

Please note that this module calls the SFB 238 internally.

In the SPEED7 Studio, this module is automatically inserted into your project.

In Siemens SIMATIC Manager, you have to copy the SFB 238 from the Motion Control Library into your project.

Parameter	Declaration	Data type	Description
Init	INPUT	BOOL	The block is internally reset with an edge 0-1. Existing motion commands are aborted and the block is initialized.
Config	IN_OUT	UDT870	Data structure for transferring axis-dependent configuration data to the *AxisKernel*.
Axis	IN_OUT	MC_AXIS_REF	Data structure for transferring axis-dependent information to the *AxisKernel* and PLCopen blocks.

3.5.3 FB 871 - VMC_InitSigma5_EC - *Sigma-5* EtherCAT initialization

Description

This block is used to configure the axis. The module is specially adapted to the use of a *Sigma-5* drive, which is connected via EtherCAT.

Parameter	Declaration	Data type	Description
Config	IN_OUT	UDT870	Data structure for transferring axis-dependent configuration data to the *AxisKernel*.
Axis	IN_OUT	MC_AXIS_REF	Data structure for transferring axis-dependent information to the *AxisKernel* and PLCopen blocks.
Enable	INPUT	BOOL	Release of initialization
Logical address	INPUT	INT	Start address of the PDO input data
InputsStartAddressPDO	INPUT	INT	Start address of the input PDOs
OutputsStartAddressPDO	INPUT	INT	Start address of the output PDOs

Parameter	Declaration	Data type	Description
EncoderType	INPUT	INT	Encoder type ■ 1: Absolute encoder ■ 2: Incremental encoder
EncoderResolutionBits	INPUT	INT	Number of bits corresponding to one encoder revolution. Default: 20
FactorPosition	INPUT	REAL	Factor for converting the position of user units [u] into drive units [increments] and back. It's valid: $p_{[increments]} = p_{[u]}$ x *FactorPosition* Please consider the factor which can be specified on the drive via the objects 0x2701: 1 and 0x2701: 2. This should be 1.
Velocity Factor	INPUT	REAL	Factor for converting the speed of user units [u/s] into drive units [increments/s] and back. It's valid: $v_{[increments/s]} = v_{[u/s]}$ x *FactorVelocity* Please also take into account the factor which you can specify on the drive via objects 0x2702: 1 and 0x2702: 2. This should be 1.
FactorAcceleration	INPUT	REAL	Factor to convert the acceleration of user units [u/s^2] in drive units [10^{-4} x increments/s^2] and back. It's valid: 10^{-4} x $a_{[increments/s^2]} = a_{[u/s^2]}$ x *FactorAcceleration* Please also take into account the factor which you can specify on the drive via objects 0x2703: 1 and 0x2703: 2. This should be 1.
OffsetPosition	INPUT	REAL	Offset for the zero position [u].
MaxVelocityApp	INPUT	REAL	Maximum application speed [u/s]. The command inputs are checked to the maximum value before execution.
MaxAccelerationApp	INPUT	REAL	Maximum acceleration of the application [u/s^2]. The command inputs are checked to the maximum value before execution.
MaxDecelerationApp	INPUT	REAL	Maximum application deceleration [u/s^2]. The command inputs are checked to the maximum value before execution.
MaxPosition	INPUT	REAL	Maximum position for monitoring the software limits [u].
MinPosition	INPUT	REAL	Minimum position for monitoring the software limits [u].
EnableMaxPosition	INPUT	BOOL	Monitoring maximum position ■ TRUE: Activates the monitoring of the maximum position.
EnableMinPosition	INPUT	BOOL	Monitoring minimum position ■ TRUE: Activation of the monitoring of the minimum position.
MinUserPosition	OUTPUT	REAL	Minimum user position based on the minimum encoder value of 0x80000000 and the *FactorPosition* [u].

Parameter	Declaration	Data type	Description
MaxUserPosition	OUTPUT	REAL	Maximum user position based on the maximum encoder value of 0x7FFFFFFF and the *FactorPosition* [u].
Valid	OUTPUT	BOOL	Initialization ■ TRUE: Initialization is valid.
Error	OUTPUT	BOOL	■ Error – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*. The axis is disabled.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

4 Usage *Sigma-7S* EtherCAT

4.1 Overview

Usage of the double-axis drive *Chapter 5 'Usage Sigma-7W EtherCAT' on page 83*

Precondition

- SPEED7 Studio from V1.6.1
  or
- Siemens SIMATIC Manager from V 5.5, SP2 & *SPEED7 EtherCAT Manager* & *Simple Motion Control Library*
- CPU with EtherCAT master, eg CPU 015-CEFNR00
- *Sigma-7S* drive with EtherCAT option card

Steps of configuration

1. Set the parameters on the drive
   - The setting of the parameters happens by means of the software tool *Sigma Win+*.
2. Hardware configuration in VIPA *SPEED7 Studio* or Siemens SIMATIC Manager
   - Configuring a CPU with EtherCAT master functionality.
   - Configuration of a *Sigma-7S* EtherCAT drive.
   - Configuring the EtherCAT connection via *SPEED7 EtherCAT Manager*.
3. Programming in VIPA *SPEED7 Studio* or Siemens SIMATIC Manager
   - Connecting the *Init* block to configure the axis.
   - Connecting the *Kernel* block to communicate with the axis.
   - Connecting the blocks for the motion sequences.

4.2 Set the parameters on the drive

CAUTION!
Before the commissioning, you have to adapt your drive to your application with the *Sigma Win+* software tool! More may be found in the manual of your drive.

The following parameters must be set via *Sigma Win+* to match the *Simple Motion Control Library*:

Sigma-7S (24bit encoder)

Servopack Parameter	Address	Name	Value
Pn205	(2205h)	Multiturn Limit Setting	65535
Pn20E	(220Eh)	ElectronicGear Ratio (Numerator)	16
Pn210	(2210h)	Electronic Gear Ratio (Denominator)	1
PnB02	(2701h:01)	Position User Unit (Numerator)	1
PnB04	(2701h:02)	Position User Unit (Denominator)	1
PnB06	(2702h:01)	Velocity User Unit (Numerator)	1
PnB08	(2702h:02)	Velocity User Unit (Denominator)	1

Servopack Parameter	Address	Name	Value
PnB0A	(2703h:01)	Acceleration User Unit (Numerator)	1
PnB0C	(2703h:02)	Acceleration User Unit (Denominator)	1

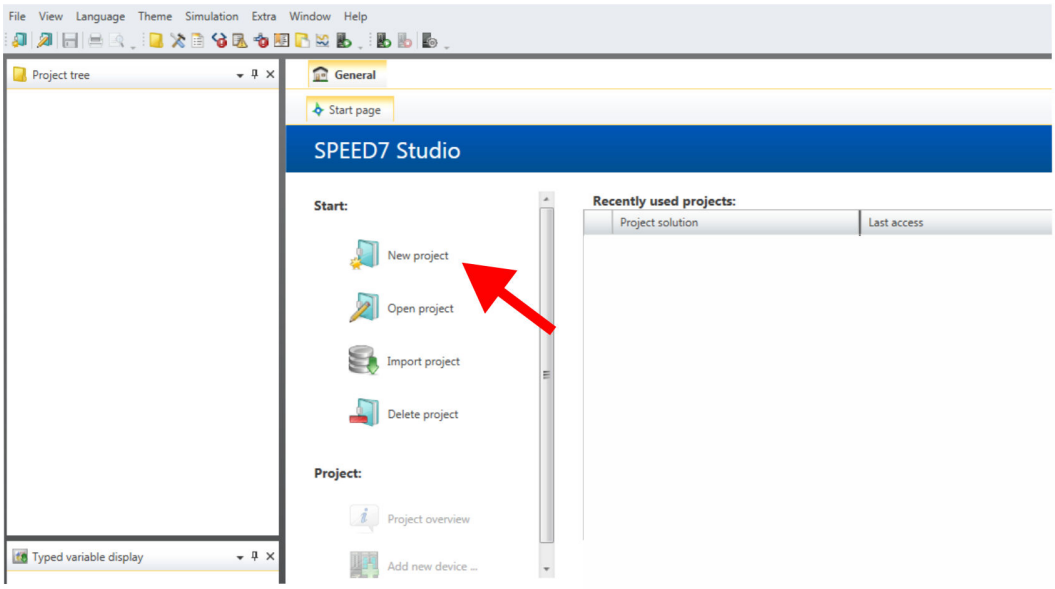
4.3 Usage in VIPA *SPEED7 Studio*

4.3.1 Hardware configuration

Add CPU in the project

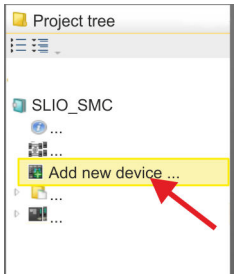
Please use for configuration the*SPEED7 Studio* V1.6.1 and up.

1. Start the *SPEED7 Studio*.

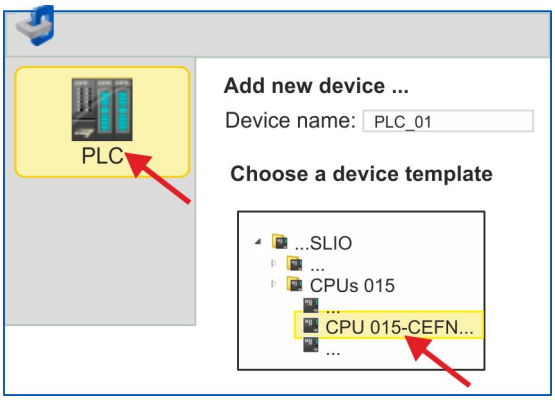


2. Create a new project at the start page with *'New project'*.

⇨ A new project is created and the view *'Devices and networking'* is shown.



3. Click in the *Project tree* at *'Add new device ...'*.

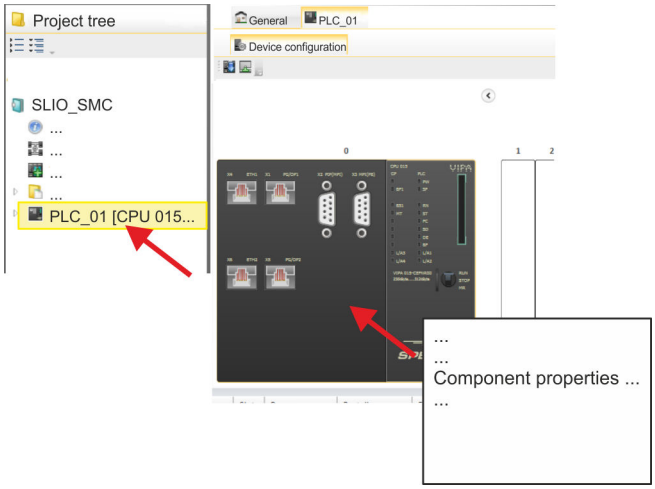


⇨ A dialog for device selection opens.

4. Select from the *'Device templates'* a CPU with EtherCAT master functions such as CPU 015-CEFNR00 and click at [OK].

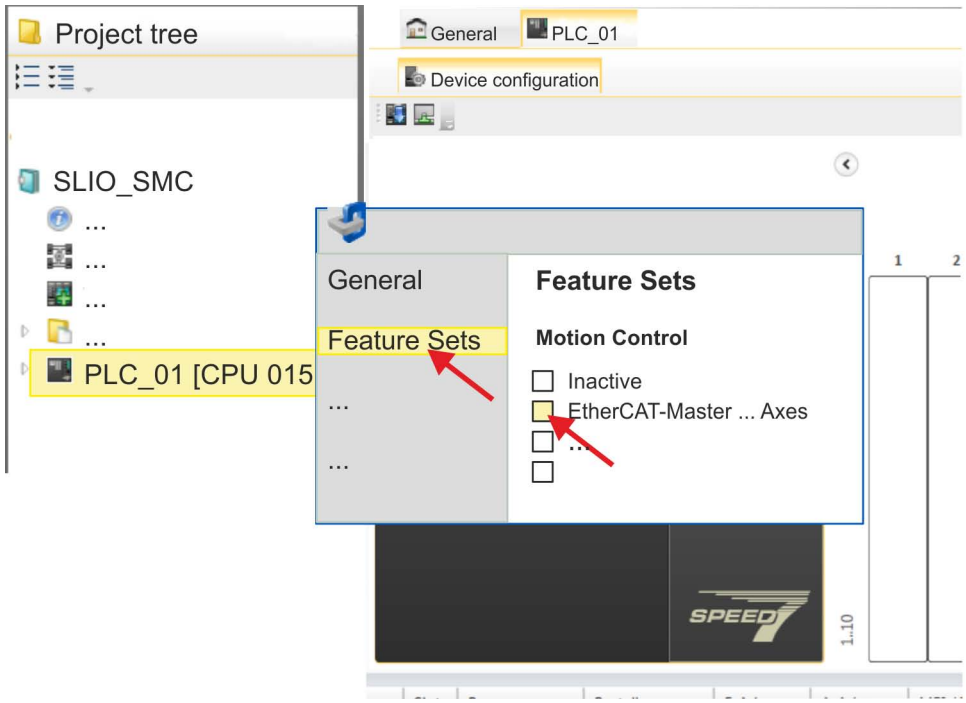
⇨ The CPU is inserted in *'Devices and networking'* and the *'Device configuration'* is opened.

Activate motion control functions



1. Click at the CPU in the *'Device configuration'* and select *'Context menu ➔ Components properties'*.

   ⇨ The properties dialog of the CPU is opened.



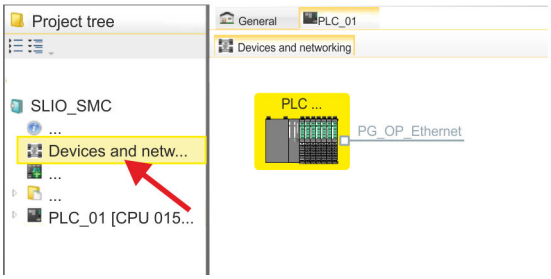
2. Click at *'Feature Sets'* and activate at *'Motion Control'* the parameter *'EtherCAT-Master... Axes'*. The number of axes is not relevant in this example.
3. Confirm your input with [OK].

   ⇨ The motion control functions are now available in your project.

CAUTION!
Please note due to the system, with every change to the feature set settings, the EtherCAT field bus system and its motion control configuration will be deleted from your project!

Configuration of Ethernet PG/OP channel

1. Click in the *Project tree* at *'Devices and networking'*.
   ⇨ You will get a graphical object view of your CPU.



2. Click at the network *'PG_OP_Ethernet'*.
3. Select *'Context menu ➔ Interface properties'*.
   ⇨ A dialog window opens. Here you can enter the IP address data for your Ethernet PG/OP channel. You get valid IP address parameters from your system administrator.
4. Confirm with [OK].
   ⇨ The IP address data are stored in your project listed in *'Devices and networking'* at *'Local components'*.

   After transferring your project your CPU can be accessed via Ethernet PG/OP channel with the set IP address data.

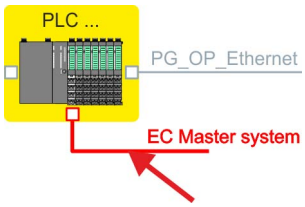
Installing the ESI file

For the *Sigma-7* EtherCAT drive can be configured in the *SPEED7 EtherCAT Manager*, the corresponding ESI file must be installed. Usually, the *SPEED7 Studio* is delivered with current ESI files and you can skip this part. If your ESI file is not up-to date, you will find the latest ESI file for the *Sigma-7* EtherCAT drive under www.yaskawa.eu.com at *'Service ➔ Drives & Motion Software'*.

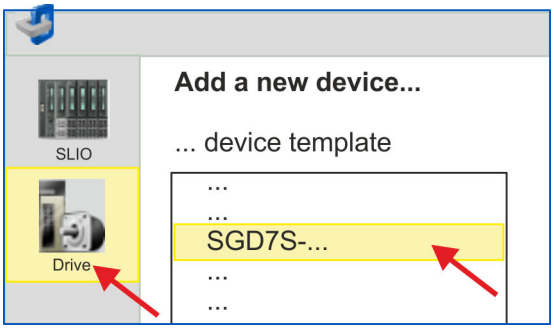
1. Download the according ESI file for your drive. Unzip this if necessary.
2. Navigate to your *SPEED7 Studio*.
3. Open the corresponding dialog window by clicking on *'Extra ➔ Install device description (EtherCAT - ESI)'*.
4. Under *'Source path'*, specify the ESI file and install it with [Install].
   ⇨ The devices of the ESI file are now available.

Add a *Sigma-7S* single axis drive

1. Click in the Project tree at *'Devices and networking'*.
2. Click here at *'EC-Mastersystem'* and select *'Context menu ➔ Add new device'*.



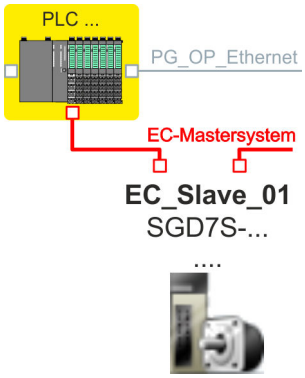
   ⇨ The device template for selecting an EtherCAT device opens.



3. Select your *Sigma-7* drive:

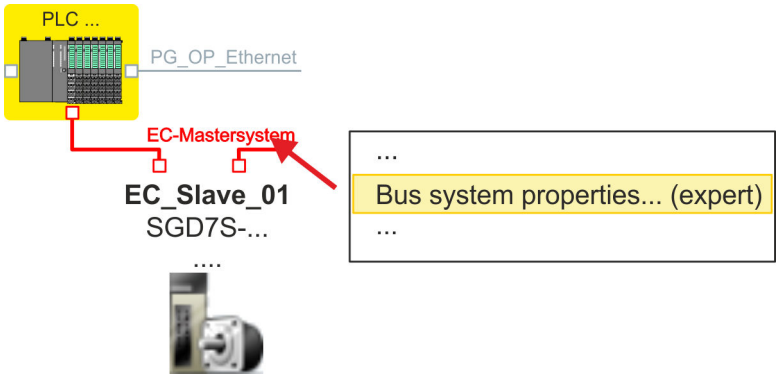
- SGD7S-xxxxAA0...
- SGD7S-xxxxDA0...
- SGD7S-xxxxA0...

Confirm with [OK]. If your drive does not exist, you must install the corresponding ESI file as described above.



⇨ The *Sigma-7* drive is connected to your EC-Mastersystem.

Configure *Sigma-7S* single axis drive

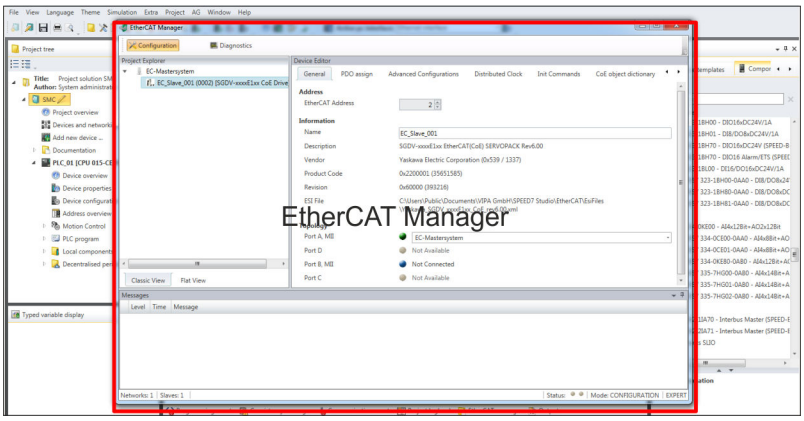


1. Click here at *'EC-Mastersystem'* and select *'Context menu ➔ Bus system properties (expert)'*.

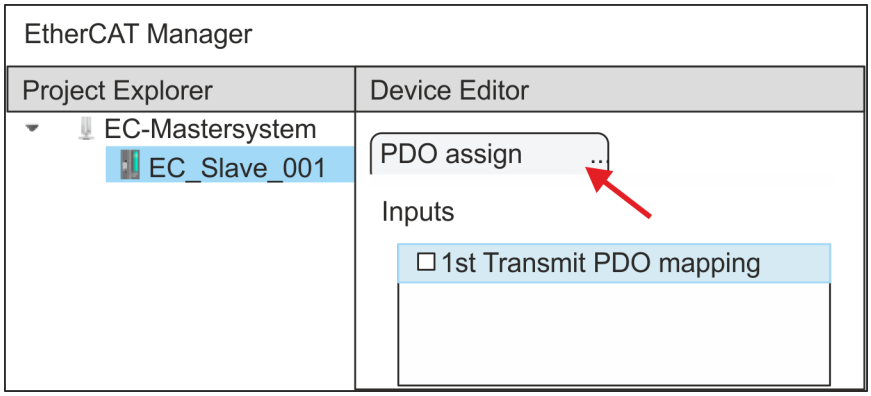
> *You can only edit PDOs in 'Expert mode'! Otherwise, the buttons are hidden.*

⇨ The *SPEED7 EtherCAT Manager* opens. Here you can configure the EtherCAT communication to your *Sigma-7* drive.

More information about the usage of the *SPEED7 EtherCAT Manager* may be found in the online help of the *SPEED7 Studio*.



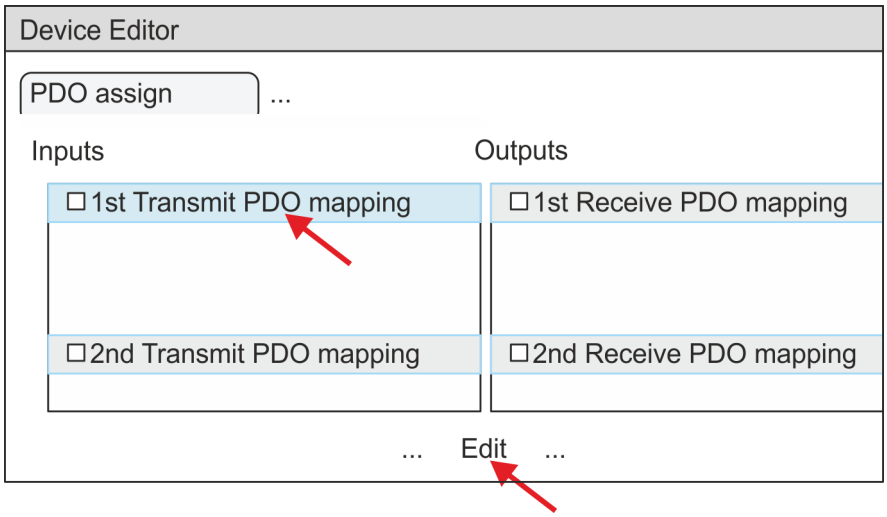
2. Click on the slave in the *SPEED7 EtherCAT Manager* and select the *'PDO assign'* tab in the *'Device editor'*.



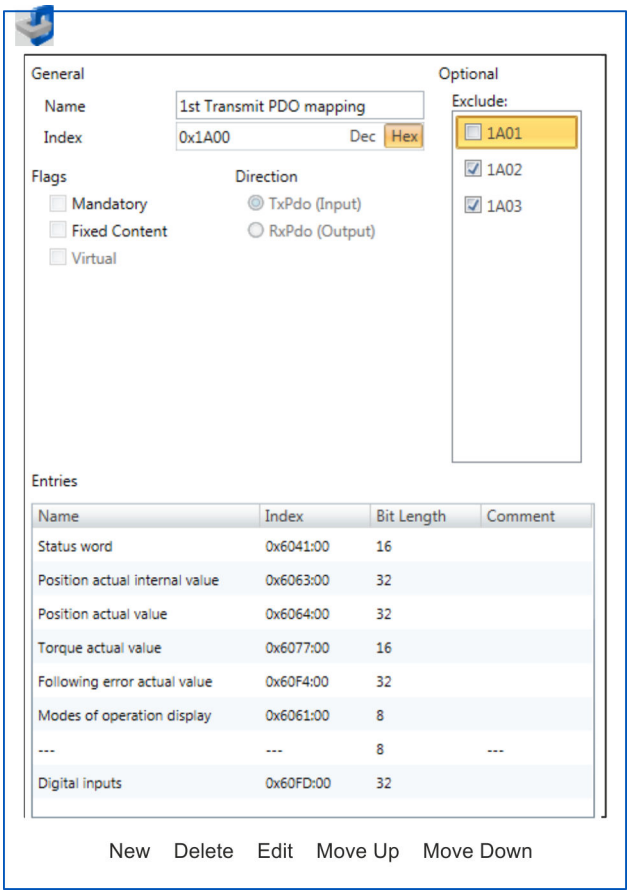
⇨ This dialog shows a list of the PDOs.

3. By selecting the appropriate mapping, you can edit the PDOs with [Edit]. Select the mapping *'1st Transmit PDO mapping'* and click at [Edit].

Please note that some PDOs can not be edited because of the default settings. By de-activating already activated PDOs, you can release the processing of locked PDOs.



⇨ The dialog *'Edit PDO'* is opened. Please check the PDO settings listed here and adjust them if necessary. Please also take into account the order of the *'Entries'* and add them accordingly.



The following functions are available for editing the *'Entries'*:

- New
  - Here you can create a new entry in a dialog by selecting the corresponding entry from the *'CoE object dictionary'* and making your settings. The entry is accepted with [OK] and is listed in the list of entries.
- Delete
  - This allows you to delete a selected entry.
- Edit
  - This allows you to edit the general data of an entry.
- Move Up/Down
  - This allows you to move the selected entry up or down in the list.

4. Perform the following settings:

Inputs: 1st Transmit PDO 0x1A00

- General
  - Name: 1st Transmit PDO mapping
  - Index: 0x1A00
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A01: de-activated
- Entries

Name	Index	Bit length
Status word	0x6041:00	16bit
Position actual internal value	0x6063:00	32bit
Position actual value	0x6064:00	32bit
Torque actual value	0x6077:00	16bit
Following error actual value	0x60F4:00	32bit
Modes of operation display	0x6061:00	8bit
---	---	8bit
Digital inputs	0x60FD:00	32bit

Close the dialog *'Edit PDO'* with [OK].

5. Select the mapping *'2nd Transmit PDO mapping'* and click at [Edit]. Perform the following settings:

Inputs: 2nd Transmit PDO 0x1A01

- General
  - Name: 2nd Transmit PDO mapping
  - Index: 0x1A01
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A00: de-activated
  - 1A02: de-activated
  - 1A03: de-activated
- Entries

Name	Index	Bit length
Touch probe status	0x60B9:00	16bit
Touch probe 1 position value	0x60BA:00	32bit
Touch probe 2 position value	0x60BC:00	32bit
Velocity actual value	0x606C:00	32bit

Close the dialog *'Edit PDO'* with [OK].

6. Select the mapping *'1st Receive PDO mapping'* and click at [Edit]. Perform the following settings:

Outputs: 1st Receive PDO 0x1600

- General
  - Name: 1st Receive PDO mapping
  - Index: 0x1600
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1601: de-activated
  - 1602: de-activated
  - 1603: de-activated
- Entries

Name	Index	Bit length
Control word	0x6040:00	16bit
Target position	0x607A:00	32bit
Target velocity	0x60FF:00	32bit
Modes of operation	0x6060:00	8bit
---	---	8bit
Touch probe function	0x60B8:00	16bit

Close the dialog *'Edit PDO'* with [OK].

7. Select the mapping *'2nd ReceivePDO mapping'* and click at [Edit]. Perform the following settings:

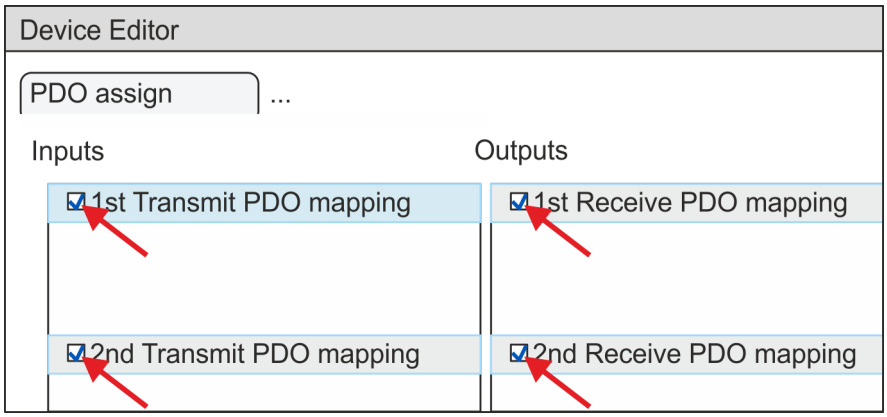
Outputs: 2nd Receive PDO 0x1601

- General
  - Name: 2nd Receive PDO mapping
  - Index: 0x1601
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

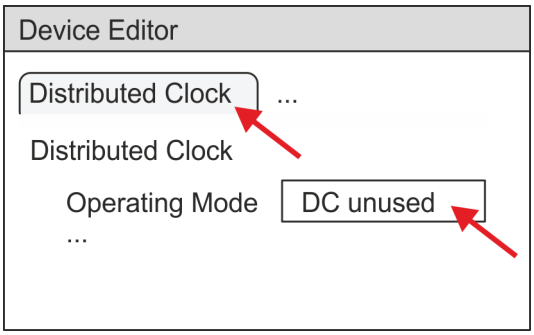
  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1600: de-activated
  - 1602: activated
  - 1603: activated
- Entries
  - Profile velocity: 0x6081:00 → 32 Bit
  - Profile acceleration: 0x6083:00 → 32 Bit
  - Profile deceleration: 0x6084:00 → 32 Bit

Close the dialog *'Edit PDO'* with [OK].

8. In PDO assignment, activate the PDOs 1 and 2 for the inputs and outputs. All subsequent PDOs must remain de-activated. If this is not possible, please check the respective PDO parameter *'Exclude'*.

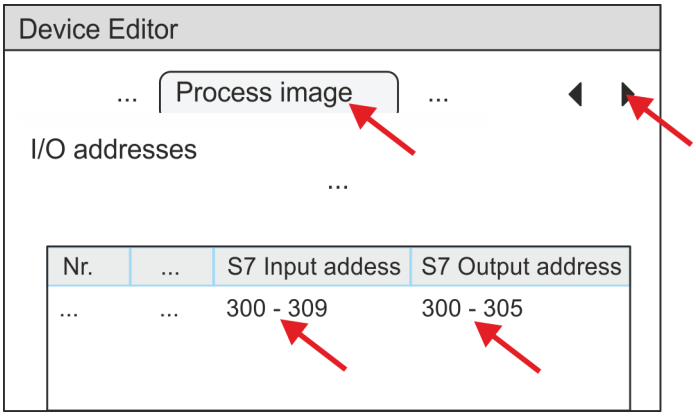


9. In the *'Device Editor'* of the *SPEED7 EtherCAT Manager*, select the *'Distributed clocks'* tab and set *'DC unused'* as *'Operating mode'*.



10. Select the *'Process image'* tab via the arrow key in the *'Device editor'* and note for the parameter of the block FB 873 - VMC_InitSigma7S_EC the following PDO.

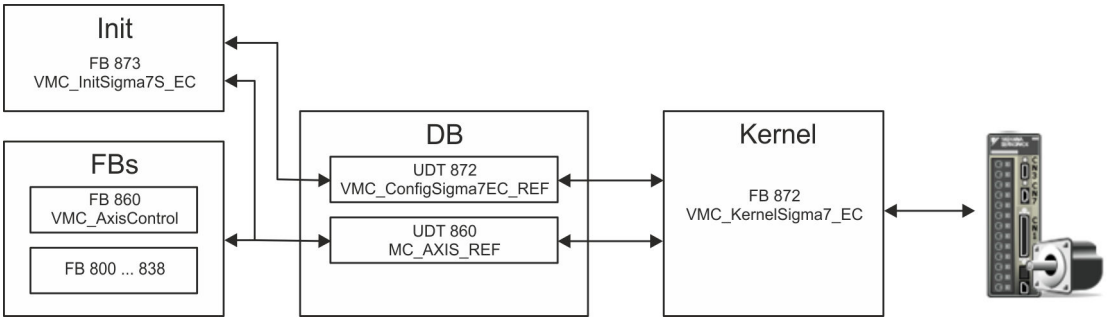
- *'S7 Input address'* → *'InputsStartAddressPDO'*
- *'S7 Output address'* → *'OutputsStartAddressPDO'*



11. By closing the dialog of the *SPEED7 EtherCAT Manager* with [X] the configuration is taken to the *SPEED7 Studio*.

4.3.2 User program

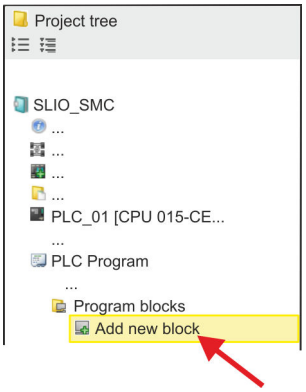
4.3.2.1 Program structure



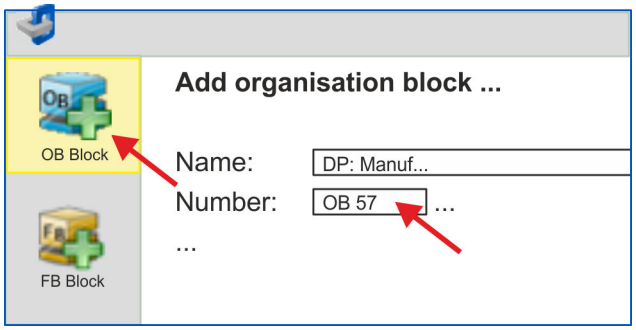
- DB
  A data block (axis DB) for configuration and status data must be created for each axis of a drive. The data block consists of the following data structures:
  – UDT 872 - *VMC_Config*Sigma7EC_REF
    The data structure describes the structure of the configuration of the drive.
    Specific data structure for *Sigma-7* EtherCAT.
  – UDT 860 - *MC_AXIS_REF*
    The data structure describes the structure of the parameters and status information of drives.
    General data structure for all drives and bus systems.
- FB 873 - *VMC_Init*Sigma7S_EC
  – The *Init*t block is used to configure an axis.
  – Specific block for *Sigma-7S* EtherCAT.
  – The configuration data for the initialization must be stored in the *axis DB*.
- FB 872 - *VMC_Kernel*Sigma7_EC
  – The *Kernel* block communicates with the drive via the appropriate bus system, processes the user requests and returns status messages.
  – Specific block for *Sigma-7* EtherCAT.
  – The exchange of the data takes place by means of the *axis DB*.
- FB 860 - *VMC_AxisControl*
  – General block for all drives and bus systems.
  – Supports simple motion commands and returns all relevant status messages.
  – The exchange of the data takes place by means of the *axis DB*.
  – For motion control and status query, via the instance data of the block you can link a visualization.
  – In addition to the FB 860 - *VMC_AxisControl*, *PLCopen* blocks can be used.
- FB 800 ... FB 838 - *PLCopen*
  – The PLCopen blocks are used to program motion sequences and status queries.
  – General blocks for all drives and bus systems.

4.3.2.2 Programming

Copy blocks into project

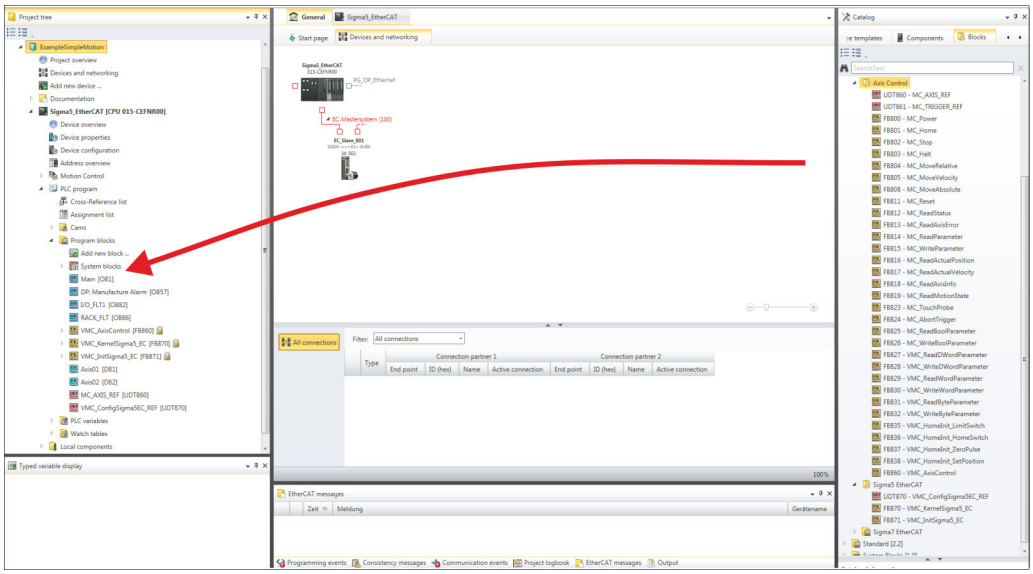


1. Click in the *Project tree* within the CPU at *'PLC program'*, *'Program blocks'* at *'Add New block'*.



⇨ The dialog *'Add block'* is opened.

2. Select the block type *'OB block'* and add one after the other OB 57, OB 82 and OB 86 to your project.



3. In the *'Catalog'*, open the *'Simple Motion Control'* library at *'Blocks'* and drag and drop the following blocks into *'Program blocks'* of the *Project tree*:
   - *Sigma-7* EtherCAT:
     - UDT 872 - VMC_ConfigSigma7EC_REF
     - FB 872 - VMC_KernelSigma7_EC
     - FB 873 - VMC_InitSigma7S_EC
   - Axis Control
     - UDT 860 - MC_AXIS_REF
     - Blocks for your movement sequences

Create axis DB

1. Add a new DB as your *axis DB* to your project. Click in the *Project tree* within the CPU at *'PLC program'*, *'Program blocks'* at *'Add New block'*, select the block type *'DB block'* and assign the name "Axis01" to it. The DB number can freely be selected such as DB10.

⇨ The block is created and opened.

2. ■ In "Axis01", create the variable "Config" of type UDT 872. These are specific axis configuration data.
   ■ In "Axis01", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

Axis01 [DB10]
Data block structure

		Adr...	Name	Data type	...
		...	Config	UDT	[872]
		...	Axis	UDT	[860]

OB 1

Configuration of the axis

Open OB 1 and program the following FB calls with associated DBs:

FB 873 - VMC_InitSigma7S_EC, DB 873 *Chapter 4.5.3 'FB 873 - VMC_InitSigma7S_EC - Sigma-7S EtherCAT Initialization' on page 80*

At *InputsStartAddressPDO* respectively *OutputsStartAddressPDO*, enter the address from the *SPEED7 EtherCAT Manager*. *57*

```
⇨ CALL  "VMC_InitSigma7S_EC" , "DI_InitSgm7SETC01"
  Enable                :="InitS7SEC1_Enable"
  LogicalAddress        :=300
  InputsStartAddressPDO :=300 (EtherCAT-Man.: S7 Input
  address)
  OutputsStartAddressPDO:=300 (EtherCAT-Man.: S7 Output
  address)
  EncoderType           :=1
  EncoderResolutionBits :=20
  FactorPosition        :=1.048576e+006
  FactorVelocity        :=1.048576e+006
  FactorAcceleration    :=1.048576e+002
  OffsetPosition        :=0.000000e+000
  MaxVelocityApp        :=5.000000e+001
  MaxAccelerationApp    :=1.000000e+002
  MaxDecelerationApp    :=1.000000e+002
  MaxVelocityDrive      :=6.000000e+001
  MaxAccelerationDrive  :=1.500000e+002
  MaxDecelerationDrive  :=1.500000e+002
  MaxPosition           :=1.048500e+003
  MinPosition           :=-1.048514e+003
  EnableMaxPosition     :=TRUE
  EnableMinPosition     :=TRUE
  MinUserPosition       :="InitS7SEC1_MinUserPos"
  MaxUserPosition       :="InitS7SEC1_MaxUserPos"
  Valid                 :="InitS7SEC1_Valid"
  Error                 :="InitS7SEC1_Error"
  ErrorID               :="InitS7SEC1_ErrorID"
  Config                :="Axis01".Config
  Axis                  :="Axis01".Axis
```

Connecting the Kernel for the axis

The *Kernel* processes the user commands and passes them appropriately processed on to the drive via the respective bus system.

FB 872 - VMC_KernelSigma7_EC, DB 872 *Chapter 4.5.2 'FB 872 - VMC_Kernel-Sigma7_EC - Sigma-7 EtherCAT Kernel' on page 80*

```
⇨       CALL  "VMC_KernelSigma7_EC" , "DI_KernelSgm5ETC01"
         Init  :="KernelS7SEC1_Init"
         Config:="Axis01".Config
         Axis  :="Axis01".Axis
```

Connecting the block for motion sequences

For simplicity, the connection of the FB 860 - VMC_AxisControl is to be shown here. This universal block supports simple motion commands and returns status messages. The inputs and outputs can be individually connected. Please specify the reference to the corresponding axis data at *'Axis'* in the *axis DB*.

FB 860 - VMC_AxisControl, DB 860 ↬ *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*

```
⇨       CALL  "VMC_AxisControl" , "DI_AxisControl01"
         SourceInputs     :="AxCtrl1_SourceInputs"
         AxisEnable       :="AxCtrl1_AxisEnable"
         AxisReset        :="AxCtrl1_AxisReset"
         HomeExecute      :="AxCtrl1_HomeExecute"
         HomePosition     :="AxCtrl1_HomePosition"
         StopExecute      :="AxCtrl1_StopExecute"
         MvVelocityExecute:="AxCtrl1_MvVelExecute"
         MvRelativeExecute:="AxCtrl1_MvRelExecute"
         MvAbsoluteExecute:="AxCtrl1_MvAbsExecute"
         PositionDistance :="AxCtrl1_PositionDistance"
         Velocity         :="AxCtrl1_Velocity"
         Acceleration     :="AxCtrl1_Acceleration"
         Deceleration     :="AxCtrl1_Deceleration"
         JogPositive      :="AxCtrl1_JogPositive"
         JogNegative      :="AxCtrl1_JogNegative"
         JogVelocity      :="AxCtrl1_JogVelocity"
         JogAcceleration  :="AxCtrl1_JogAcceleration"
         JogDeceleration  :="AxCtrl1_JogDeceleration"
         AxisReady        :="AxCtrl1_AxisReady"
         AxisEnabled      :="AxCtrl1_AxisEnabled"
         AxisError        :="AxCtrl1_AxisError"
         AxisErrorID      :="AxCtrl1_AxisErrorID"
         DriveWarning     :="AxCtrl1_DriveWarning"
         DriveError       :="AxCtrl1_DriveError"
         DriveErrorID     :="AxCtrl1_DriveErrorID"
         IsHomed          :="AxCtrl1_IsHomed"
         ModeOfOperation  :="AxCtrl1_ModeOfOperation"
         PLCopenState     :="AxCtrl1_PLCopenState"
         ActualPosition   :="AxCtrl1_ActualPosition"
         ActualVelocity   :="AxCtrl1_ActualVelocity"
         CmdDone          :="AxCtrl1_CmdDone"
         CmdBusy          :="AxCtrl1_CmdBusy"
         CmdAborted       :="AxCtrl1_CmdAborted"
         CmdError         :="AxCtrl1_CmdError"
         CmdErrorID       :="AxCtrl1_CmdErrorID"
         DirectionPositive:="AxCtrl1_DirectionPos"
         DirectionNegative:="AxCtrl1_DirectionNeg"
         SWLimitMinActive :="AxCtrl1_SWLimitMinActive"
         SWLimitMaxActive :="AxCtrl1_SWLimitMaxActive"
         HWLimitMinActive :="AxCtrl1_HWLimitMinActive"
         HWLimitMaxActive :="AxCtrl1_HWLimitMaxActive"
         Axis             :="Axis01".Axis
```

For complex motion tasks, you can use the PLCopen blocks. Please specify the reference to the corresponding axis data at Axis in the axis DB.

Your project now includes the following blocks:

- OB 1 - Main
- OB 57 - DP Manufacturer Alarm
- OB 82 - I/O_FLT1
- OB 86 - Rack_FLT

- FB 860 - VMC_AxisControl with instance DB
- FB 872 - VMC_KernelSigma7_EC with instance DB
- FB 873 - VMC_InitSigma7S_EC with instance DB
- UDT 860 - MC_Axis_REF
- UDT 872 - VMC_ConfigSigma7EC_REF

Sequence of operations

1. Select *'Project ➔ Compile all'* and transfer the project into your CPU. You can find more information on the transfer of your project in the online help of the *SPEED7 Studio*.

   ⇨ You can take your application into operation now.

CAUTION!
Please always observe the safety instructions for your drive, especially during commissioning!

2. Before an axis can be controlled, it must be initialized. To do this, call the *Init* block FB 873 - VMC_InitSigma7S_EC with *Enable* = TRUE.

   ⇨ The output *Valid* returns TRUE. In the event of a fault, you can determine the error by evaluating the *ErrorID*.

   You have to call the *Init* block again if you load a new axis DB or you have changed parameters on the *Init* block.

Do not continue until the Init block report an error!

3. Ensure that the *Kernel* block FB 872 - VMC_KernelSigma7_EC is called cyclically. In this way, control signals are transmitted to the drive and status messages are reported.
4. Program your application with the FB 860 - VMC_AxisControl or with the PLCopen blocks.

4.4 Usage in Siemens SIMATIC Manager

4.4.1 Precondition

Overview

- Please use for configuration the Siemens SIMATIC Manager V 5.5 SP2 and up.
- The configuration of the System SLIO CPU happens in the Siemens SIMATIC Manager by means of a virtual PROFINET IO device *'VIPA SLIO CPU'*. The *'VIPA SLIO CPU'* is to be installed in the hardware catalog by means of the GSDML.
- The configuration of the EtherCAT masters happens in the Siemens SIMATIC Manager by means of a virtual PROFINET IO device *'EtherCAT network'*. The *'EtherCAT network'* is to be installed in the hardware catalog by means of the GSDML.
- The *'EtherCAT network'* can be configured with the VIPA Tool *SPEED7 EtherCAT Manager*.
- For the configuration of the drive in the *SPEED7 EtherCAT Manager* the installation of the according ESI file is necessary.

Installing the IO device *'VIPA SLIO System'*

The installation of the PROFINET IO device *'VIPA SLIO CPU'* happens in the hardware catalog with the following approach:

1. Go to the service area of www.vipa.com.
2. Download the configuration file for your CPU from the download area via *'Config files ➔ PROFINET'*.
3. Extract the file into your working directory.
4. Start the Siemens hardware configurator.
5. Close all the projects.
6. Select *'Options ➔ Install new GSD file'*.
7. Navigate to your working directory and install the according GSDML file.
   ⇨ After the installation the according PROFINET IO device can be found at *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA SLIO System'*.

Installing the IO device EtherCAT network

The installation of the PROFINET IO devices *'EtherCAT Network'* happens in the hardware catalog with the following approach:

1. Go to the service area of www.vipa.com
2. Load from the download area at *'Config files ➔ EtherCAT'* the GSDML file for your EtherCAT master.
3. Extract the files into your working directory.
4. Start the Siemens hardware configurator.
5. Close all the projects.
6. Select *'Options ➔ Install new GSD file'*.
7. Navigate to your working directory and install the according GSDML file.
   ⇨ After the installation the *'EtherCAT Network'* can be found at *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA VIPA EtherCAT System'*.

Installing the *SPEED7 EtherCAT Manager*

The configuration of the PROFINET IO device *'EtherCAT Network'* happens by means of the *SPEED7 EtherCAT Manager* from VIPA. This may be found in the service area of www.vipa.com at *'Service/Support ➔ Downloads ➔ SPEED7'*.

The installation happens with the following proceeding:

1. Close the Siemens SIMATIC Manager.
2. Go to the service area of www.vipa.com
3. Load the *SPEED7 EtherCAT Manager* and unzip it on your PC.
4. For installation start the file EtherCATManager_v... .exe.
5. Select the language for the installation.
6. Accept the licensing agreement.
7. Select the installation directory and start the installation.
8. After installation you have to reboot your PC.
   ⇨ The *SPEED7 EtherCAT Manager* is installed and can now be called via the context menu of the Siemens SIMATIC Manager.

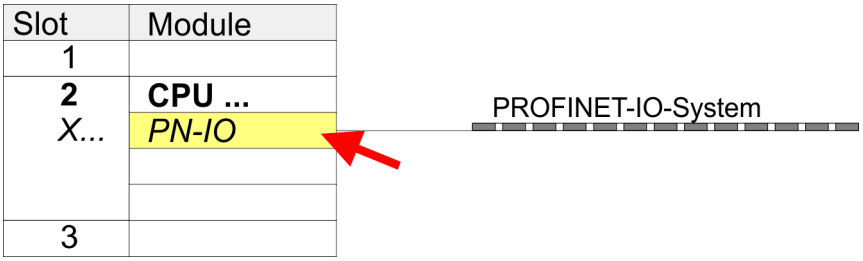
4.4.2 Hardware configuration

Configuring the CPU in the project

Slot	Module
1	
2	**CPU 315-2 PN/DP**
X1	*MPI/DP*
X2	*PN-IO*
X2...	*Port 1*
X2...	*Port 2*
3	

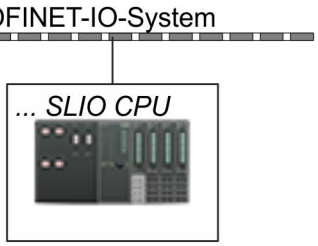
To be compatible with the Siemens SIMATIC Manager the following steps should be executed:

1. Start the Siemens hardware configurator with a new project.
2. Insert a profile rail from the hardware catalog.
3. Place at *'Slot'* number 2 the CPU 315-2 PN/DP (315-2EH14 V3.2).
4. The integrated PROFIBUS DP master (jack X3) is to be configured and connected via the sub module *'X1 MPI/DP'*.
5. The integrated EtherCAT master is to be configured via the sub module *'X2 PN-IO'* as a virtual PROFINET network.
6. Click at the sub module *'PN-IO'* of the CPU.
7. Select *'Context menu ➔ Insert PROFINET IO System'*.



8. Create with [New] a new sub net and assign valid address data
9. Click at the sub module *'PN-IO'* of the CPU and open with *'Context menu ➔ Properties'* the properties dialog.
10. Enter at *'General'* a *'Device name'*. The device name must be unique at the Ethernet subnet.

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	



Slot	Module	Order number	
0	***... SLIO CPU ...***	***015-...***	
X2	*015-...*		
1			
2			
3			
...			

11. Navigate in the hardware catalog to the directory *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA SLIO System'* and connect the IO device *'015-CFFNR00 CPU'* to your PROFINET system.

⇨ In the Device overview of the PROFINET IO device *'VIPA SLIO CPU'* the CPU is already placed at slot 0. From slot 1 you can place your System SLIO modules.

Configuration of Ethernet PG/OP channel

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	
4	343-1EX30
5	
...	

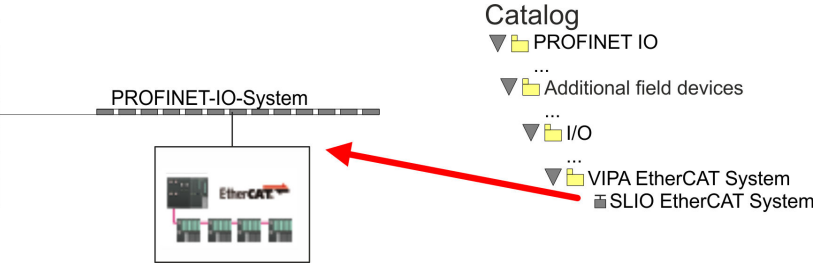
1. Place for the Ethernet PG/OP channel at slot 4 the Siemens CP 343-1 (SIMATIC 300 \ CP 300 \ Industrial Ethernet \CP 343-1 \ 6GK7 343-1EX30 0XE0 V3.0).

2. Open the properties dialog by clicking on the CP 343-1EX30 and enter for the CP at *'Properties'* the IP address data. You get valid IP address parameters from your system administrator.

3. Assign the CP to a *'Subnet'*. The IP address data are not accepted without assignment!

Insert *'EtherCAT network'*

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	

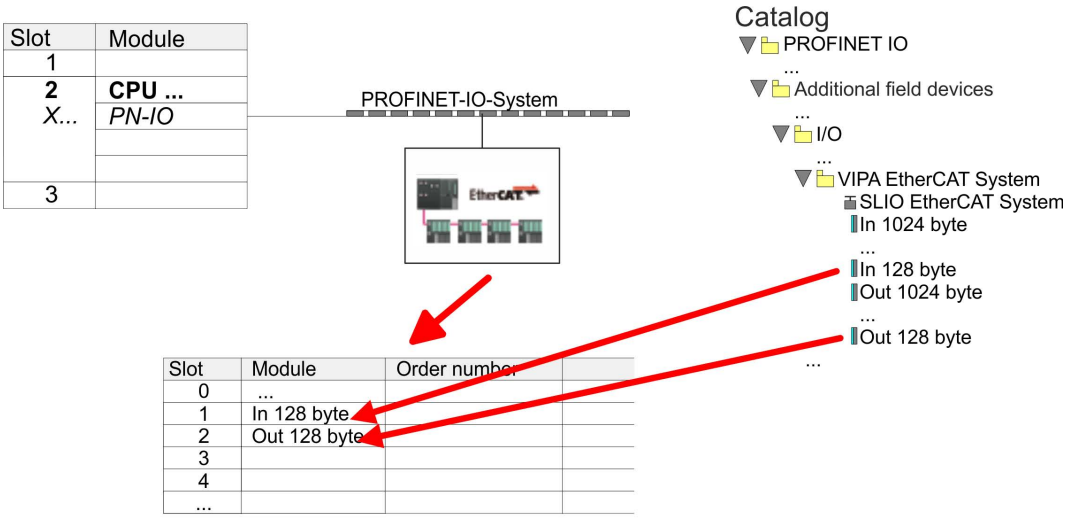


1. Navigate in the hardware catalog to the directory *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA EtherCAT System'* and connect the IO device *'SLIO EtherCAT System'* to your PROFINET system.

2. Click at the inserted IO device *'EtherCAT Network'* and define the areas for in and output by drag and dropping the according *'Out'* or *'In'* area to a slot.

Create the following areas:

- In 128byte
- Out 128byte

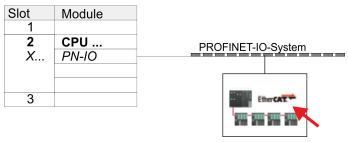


3. Select *'Station ➔ Save and compile'*

Sigma-7S Configure EtherCAT drive

The drive is configured in the *SPEED7 EtherCAT Manager*.

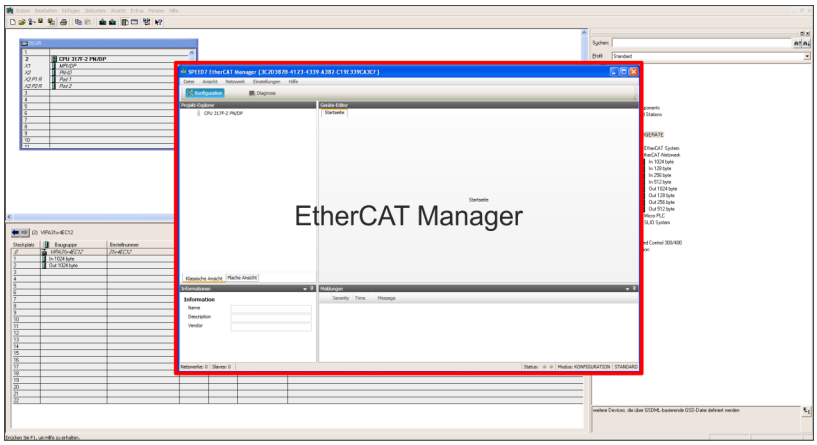
> *Before calling the SPEED7 EtherCAT Manager you have always to save your project with 'Station ➔ Save and compile'.*



1. Click at an inserted IO device *'EtherCAT Network'* and select *'Context menu ➔ Start Device-Tool ➔ SPEED7 EtherCAT Manager'*.

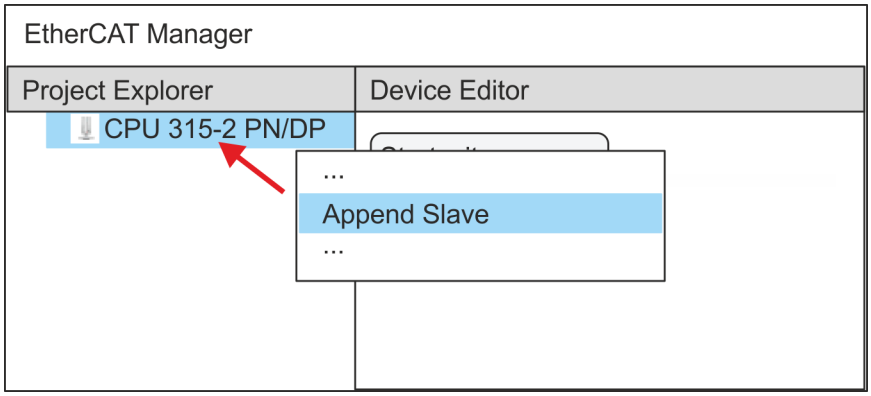
⇨ The *SPEED7 EtherCAT Manager* opens. Here you can configure the EtherCAT communication to your *Sigma-7S* drive.

More information about the usage of the *SPEED7 EtherCAT Manager* may be found in the according manual or online help.



3. For the *Sigma-7S* EtherCAT drive to be configured in the *SPEED7 EtherCAT Manager*, the corresponding ESI file must be installed. The ESI file for the *Sigma-7S* EtherCAT drive can be found under www.yaskawa.eu.com at *'Service ➔ Drives & Motion Software'*. Download the according ESI file for your drive. Unzip this if necessary.
4. Open in the *SPEED7 EtherCAT Manager* via *'File ➔ ESI Manager'* the dialogue window *'ESI Manager'*.
5. In the *'ESI Manager'* click at [Add File] and select your ESI file. With [Open], the ESI file is installed in the *SPEED7 EtherCAT Manager*.
6. Close the *'ESI Manager'*.

   ⇨ Your *Sigma-7S* EtherCAT drive is now available for configuration.

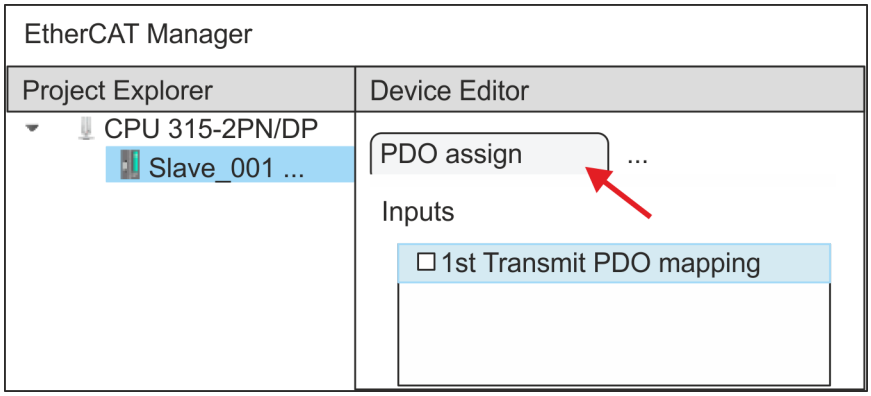


7. In the EtherCAT Manager, click on your CPU and open via *'Context menu ➔ Append Slave'* the dialog box for adding an EtherCAT slave.

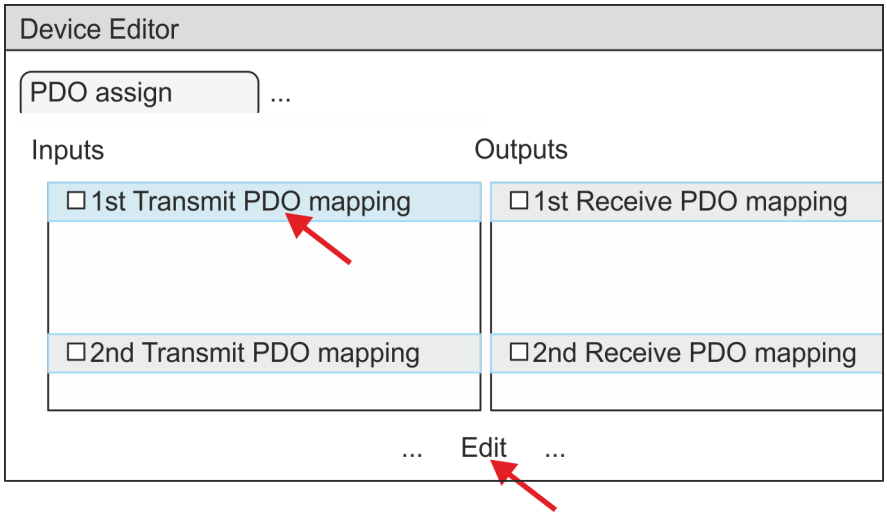
   ⇨ The dialog window for selecting an EtherCAT slave is opened.
8. Select your *Sigma-7S* EtherCAT drive and confirm your selection with [OK].

   ⇨ The *Sigma-7S* EtherCAT drive is connected to the master and can now be configured.
9. *You can only edit PDOs in 'Expert mode'! Otherwise, the buttons are hidden. By activating the 'Expert mode' you can switch to advanced setting.*

   By activating *'View ➔ Expert'* you can switch to the *Expert mode*.
10. Click on the *Sigma-7S* EtherCAT Slave in the *SPEED7 EtherCAT Manager* and select the *'PDO assign'* tab in the *'Device editor'*.

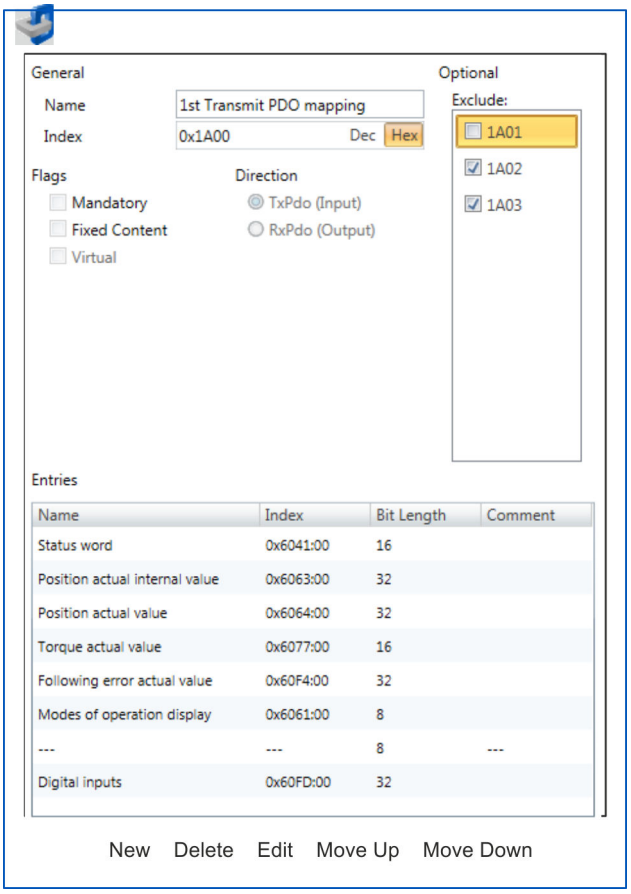


   ⇨ This dialog shows a list of the PDOs.



11. By selecting the appropriate PDO mapping, you can edit the PDOs with [Edit]. Select the mapping *'1st Transmit PDO mapping'* and click at [Edit].

Please note that some PDOs can not be edited because of the default settings. By de-activating already activated PDOs, you can release the processing of locked PDOs.



⇨ The dialog *'Edit PDO'* is opened. Please check the PDO settings listed here and adjust them if necessary. Please also take into account the order of the *'Entries'* and add them accordingly.

The following functions are available for editing the *'Entries'*:

- New
  - Here you can create a new entry in a dialog by selecting the corresponding entry from the *'CoE object dictionary'* and making your settings. The entry is accepted with [OK] and is listed in the list of entries.
- Delete
  - This allows you to delete a selected entry.
- Edit
  - This allows you to edit the general data of an entry.
- Move Up/Down
  - This allows you to move the selected entry up or down in the list.

12. Perform the following settings:

Inputs: 1st Transmit PDO 0x1A00

- General
  - Name: 1st Transmit PDO mapping
  - Index: 0x1A00
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A01: de-activated
- Entries

Name	Index	Bit length
Status word	0x6041:00	16bit
Position actual internal value	0x6063:00	32bit
Position actual value	0x6064:00	32bit
Torque actual value	0x6077:00	16bit
Following error actual value	0x60F4:00	32bit
Modes of operation display	0x6061:00	8bit
---	---	8bit
Digital inputs	0x60FD:00	32bit

Close the dialog *'Edit PDO'* with [OK].

13. Select the mapping *'2nd Transmit PDO mapping'* and click at [Edit]. Perform the following settings:

Inputs: 2nd Transmit PDO 0x1A01

- General
  - Name: 2nd Transmit PDO mapping
  - Index: 0x1A01
- Flags
  - Everything de-activated
- Direction
  - TxPdo (Input): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1A00: de-activated
  - 1A02: de-activated
  - 1A03: de-activated
- Entries

Name	Index	Bit length
Touch probe status	0x60B9:00	16bit
Touch probe 1 position value	0x60BA:00	32bit
Touch probe 2 position value	0x60BC:00	32bit
Velocity actual value	0x606C:00	32bit

Close the dialog *'Edit PDO'* with [OK].

14. Select the mapping *'1st Receive PDO mapping'* and click at [Edit]. Perform the following settings:

Outputs: 1st Receive PDO 0x1600

- General
  - Name: 1st Receive PDO mapping
  - Index: 0x1600
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1601: de-activated
  - 1602: de-activated
  - 1603: de-activated
- Entries

Name	Index	Bit length
Control word	0x6040:00	16bit
Target position	0x607A:00	32bit
Target velocity	0x60FF:00	32bit
Modes of operation	0x6060:00	8bit
---	---	8bit
Touch probe function	0x60B8:00	16bit

Close the dialog *'Edit PDO'* with [OK].

15. Select the mapping *'2nd ReceivePDO mapping'* and click at [Edit]. Perform the following settings:

Outputs: 2nd Receive PDO 0x1601

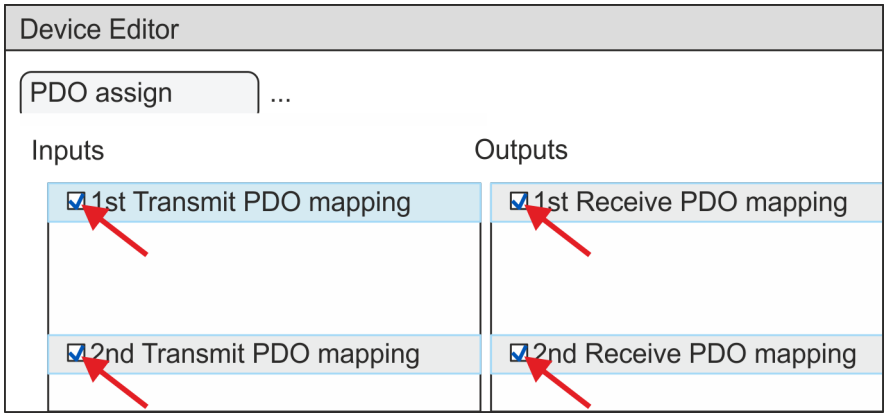
- General
  - Name: 2nd Receive PDO mapping
  - Index: 0x1601
- Flags
  - Everything de-activated
- Direction
  - RxPdo (Output): activated
- Exclude

  Please note these settings, otherwise the PDO mappings can not be activated at the same time!
  - 1600: de-activated
  - 1602: activated
  - 1603: activated
- Entries

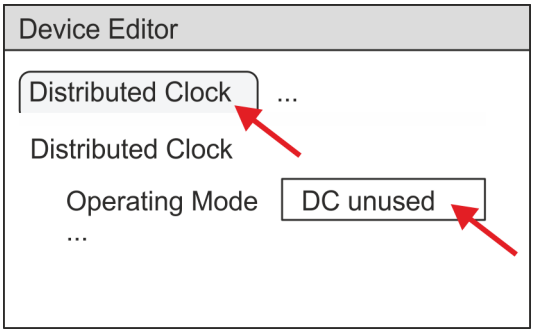
Name	Index	Bit length
Profile velocity	0x6081:00	32bit
Profile acceleration	0x6083:00	32bit
Profile deceleration	0x6084:00	32bit

Close the dialog *'Edit PDO'* with [OK].

16. In PDO assignment, activate the PDOs 1 and 2 for the inputs and outputs. All subsequent PDOs must remain de-activated. If this is not possible, please check the respective PDO parameter *'Exclude'*.

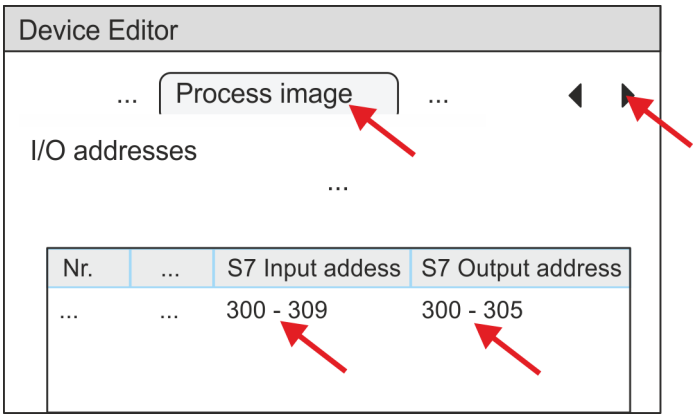


17. In the *'Device Editor'* of the *SPEED7 EtherCAT Manager*, select the *'Distributed clocks'* tab and set *'DC unused'* as *'Operating mode'*.



18. Select the *'Process image'* tab via the arrow key in the *'Device editor'* and note for the parameter of the block FB 873 - VMC_InitSigma7S_EC the following PDO.

- *'S7 Input address'* → *'InputsStartAddressPDO'*
- *'S7 Output address'* → *'OutputsStartAddressPDO'*

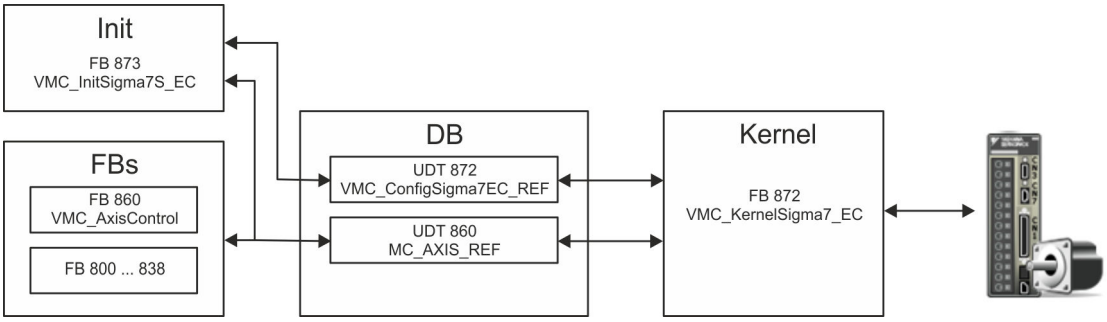


19. By closing the *SPEED7 EtherCAT Manager* with [X] the configuration is taken to the project. You can always edit your EtherCAT configuration in the *SPEED7 EtherCAT Manager*, since the configuration is stored in your project.

20. Save and compile your configuration.

4.4.3 User program

4.4.3.1 Program structure



- DB
  A data block (axis DB) for configuration and status data must be created for each axis of a drive. The data block consists of the following data structures:
  – UDT 872 - *VMC_Config*Sigma7EC_REF
    The data structure describes the structure of the configuration of the drive.
    Specific data structure for *Sigma-7* EtherCAT.
  – UDT 860 - *MC_AXIS_REF*
    The data structure describes the structure of the parameters and status information of drives.
    General data structure for all drives and bus systems.
- FB 873 - *VMC_Init*Sigma7S_EC
  – The *Init*t block is used to configure an axis.
  – Specific block for *Sigma-7S* EtherCAT.
  – The configuration data for the initialization must be stored in the *axis DB*.

- FB 872 - *VMC_Kernel*Sigma7_EC
  - The *Kernel* block communicates with the drive via the appropriate bus system, processes the user requests and returns status messages.
  - Specific block for *Sigma-7* EtherCAT.
  - The exchange of the data takes place by means of the *axis DB*.
- FB 860 - *VMC_AxisControl*
  - General block for all drives and bus systems.
  - Supports simple motion commands and returns all relevant status messages.
  - The exchange of the data takes place by means of the *axis DB*.
  - For motion control and status query, via the instance data of the block you can link a visualization.
  - In addition to the FB 860 - *VMC_AxisControl*, *PLCopen* blocks can be used.
- FB 800 ... FB 838 - *PLCopen*
  - The PLCopen blocks are used to program motion sequences and status queries.
  - General blocks for all drives and bus systems.

4.4.3.2 Programming

Include library

1. Go to the service area of www.vipa.com.
2. Download the *Simple Motion Control* library from the download area at *'VIPA Lib'*.
3. Open the dialog window for ZIP file selection via *'File ➔ Retrieve'*.
4. Select the according ZIP file and click at [Open].
5. Specify a target directory in which the blocks are to be stored and start the unzip process with [OK].

Copy blocks into project

→ Open the library after unzipping and drag and drop the following blocks into *'Blocks'* of your project:

- *Sigma-7S* EtherCAT:
  - UDT 872 - VMC_ConfigSigma7EC_REF
  - FB 872 - VMC_KernelSigma7_EC
  - FB 873 - VMC_InitSigma7S_EC
- Axis Control
  - UDT 860 - MC_AXIS_REF
  - Blocks for your movement sequences

Create interrupt OBs

1. In your project, click at *'Blocks'* and choose *'Context menu ➔ Insert new object ➔ Organization block'*.
   ⇨ The dialog *'Properties Organization block'* opens.
2. Add OB 57, OB 82, and OB 86 successively to your project.

Create axis DB

1. In your project, click at *'Blocks'* and choose *'Context menu ➔ Insert new object ➔ Data block'*.

Specify the following parameters:

- Name and type
  - The DB no. as *'Name'* can freely be chosen, such as DB10.
  - Set *'Shared DB'* as the *'Type'*.
- Symbolic name
  - Specify "Axis01".

Confirm your input with [OK].

⇨ The block is created.

2. Open DB10 "Axis01" by double-click.

- In "Axis01", create the variable "Config" of type UDT 872. These are specific axis configuration data.
- In "Axis01", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

DB10

Address	Name	Type	...
		Struct	
...	Config	"VMC_ConfigSigma7EC_REF"	
...	Axis	"MC_AXIS_REF	
...		END_STRUCT	

OB 1

Configuration of the axis

Open OB 1 and program the following FB calls with associated DBs:

FB 873 - VMC_InitSigma7S_EC, DB 873 *Chapter 4.5.3 'FB 873 - VMC_InitSigma7S_EC - Sigma-7S EtherCAT Initialization' on page 80*

At *InputsStartAddressPDO* respectively *OutputsStartAddressPDO*, enter the address from the *SPEED7 EtherCAT Manager*. *73*

```
⇨ CALL  "VMC_InitSigma7S_EC" , "DI_InitSgm7SETC01"
  Enable                :="InitS7SEC1_Enable"
  LogicalAddress        :=300
  InputsStartAddressPDO :=300(EtherCAT-Man:S7 Input address)
  OutputsStartAddressPDO:=300(EtherCAT-Man:S7 Output address)
  EncoderType           :=1
  EncoderResolutionBits :=20
  FactorPosition        :=1.048576e+006
  FactorVelocity        :=1.048576e+006
  FactorAcceleration    :=1.048576e+002
  OffsetPosition        :=0.000000e+000
  MaxVelocityApp        :=5.000000e+001
  MaxAccelerationApp    :=1.000000e+002
  MaxDecelerationApp    :=1.000000e+002
  MaxVelocityDrive      :=6.000000e+001
  MaxAccelerationDrive  :=1.500000e+002
  MaxDecelerationDrive  :=1.500000e+002
  MaxPosition           :=1.048500e+003
  MinPosition           :=-1.048514e+003
  EnableMaxPosition     :=TRUE
  EnableMinPosition     :=TRUE
  MinUserPosition       :="InitS5EC1_MinUserPos"
  MaxUserPosition       :="InitS5EC1_MaxUserPos"
  Valid                 :="InitS5EC1_Valid"
  Error                 :="InitS5EC1_Error"
  ErrorID               :="InitS5EC1_ErrorID"
  Config                :="Axis01".Config
  Axis                  :="Axis01".Axis
```

Connecting the Kernel for the axis

The *Kernel* processes the user commands and passes them appropriately processed on to the drive via the respective bus system.

FB 872 - VMC_KernelSigma7_EC, DB 872 *Chapter 4.5.2 'FB 872 - VMC_KernelSigma7_EC - Sigma-7 EtherCAT Kernel' on page 80*

```
⇨       CALL  "VMC_KernelSigma7_EC" , "DI_KernelSgm7ETC01"
         Init  :="KernelS7EC1_Init"
         Config:="Axis01".Config
         Axis  :="Axis01".Axis
```

Connecting the block for motion sequences

For simplicity, the connection of the FB 860 - VMC_AxisControl is to be shown here. This universal block supports simple motion commands and returns status messages. The inputs and outputs can be individually connected. Please specify the reference to the corresponding axis data at *'Axis'* in the *axis DB*.

FB 860 - VMC_AxisControl, DB 860 *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*

```
⇨       CALL  "VMC_AxisControl" , "DI_AxisControl01"
         SourceInputs     :="AxCtrl1_SourceInputs"
         AxisEnable       :="AxCtrl1_AxisEnable"
         AxisReset        :="AxCtrl1_AxisReset"
         HomeExecute      :="AxCtrl1_HomeExecute"
         HomePosition     :="AxCtrl1_HomePosition"
         StopExecute      :="AxCtrl1_StopExecute"
         MvVelocityExecute:="AxCtrl1_MvVelExecute"
         MvRelativeExecute:="AxCtrl1_MvRelExecute"
         MvAbsoluteExecute:="AxCtrl1_MvAbsExecute"
         PositionDistance :="AxCtrl1_PositionDistance"
         Velocity         :="AxCtrl1_Velocity"
         Acceleration     :="AxCtrl1_Acceleration"
         Deceleration     :="AxCtrl1_Deceleration"
         JogPositive      :="AxCtrl1_JogPositive"
         JogNegative      :="AxCtrl1_JogNegative"
         JogVelocity      :="AxCtrl1_JogVelocity"
         JogAcceleration  :="AxCtrl1_JogAcceleration"
         JogDeceleration  :="AxCtrl1_JogDeceleration"
         AxisReady        :="AxCtrl1_AxisReady"
         AxisEnabled      :="AxCtrl1_AxisEnabled"
         AxisError        :="AxCtrl1_AxisError"
         AxisErrorID      :="AxCtrl1_AxisErrorID"
         DriveWarning     :="AxCtrl1_DriveWarning"
         DriveError       :="AxCtrl1_DriveError"
         DriveErrorID     :="AxCtrl1_DriveErrorID"
         IsHomed          :="AxCtrl1_IsHomed"
         ModeOfOperation  :="AxCtrl1_ModeOfOperation"
         PLCopenState     :="AxCtrl1_PLCopenState"
         ActualPosition   :="AxCtrl1_ActualPosition"
         ActualVelocity   :="AxCtrl1_ActualVelocity"
         CmdDone          :="AxCtrl1_CmdDone"
         CmdBusy          :="AxCtrl1_CmdBusy"
         CmdAborted       :="AxCtrl1_CmdAborted"
         CmdError         :="AxCtrl1_CmdError"
         CmdErrorID       :="AxCtrl1_CmdErrorID"
         DirectionPositive:="AxCtrl1_DirectionPos"
         DirectionNegative:="AxCtrl1_DirectionNeg"
         SWLimitMinActive :="AxCtrl1_SWLimitMinActive"
         SWLimitMaxActive :="AxCtrl1_SWLimitMaxActive"
         HWLimitMinActive :="AxCtrl1_HWLimitMinActive"
         HWLimitMaxActive :="AxCtrl1_HWLimitMaxActive"
         Axis             :="Axis01".Axis
```

For complex motion tasks, you can use the PLCopen blocks. Please specify the reference to the corresponding axis data at Axis in the axis DB.

Your project now includes the following blocks:

- OB 1 - Main
- OB 57 - DP Manufacturer Alarm
- OB 82 - I/O_FLT1
- OB 86 - Rack_FLT

- FB 860 - VMC_AxisControl with instance DB
- FB 872 - VMC_KernelSigma7_EC with instance DB
- FB 873 - VMC_InitSigma7S_EC with instance DB
- UDT 860 - MC_Axis_REF
- UDT 872 - VMC_ConfigSigma7EC_REF

Sequence of operations

1. Choose the Siemens SIMATIC Manager and transfer your project into the CPU.

   The transfer can only be done by the Siemens SIMATIC Manager - not hardware configurator!

   Since slave and module parameters are transmitted by means of SDO respectively SDO Init command, the configuration remains active, until a power cycle is performed or new parameters for the same SDO objects are transferred.

   With an overall reset the slave and module parameters are not reset!

   ⇨ You can take your application into operation now.

   CAUTION!
   Please always observe the safety instructions for your drive, especially during commissioning!

2. Before an axis can be controlled, it must be initialized. To do this, call the *Init* block FB 873 - VMC_InitSigma7S_EC with *Enable* = TRUE.

   ⇨ The output *Valid* returns TRUE. In the event of a fault, you can determine the error by evaluating the *ErrorID*.

   You have to call the *Init* block again if you load a new axis DB or you have changed parameters on the *Init* block.

   Do not continue until the Init block report an error!

3. Ensure that the *Kernel* block FB 872 - VMC_KernelSigma7_EC is called cyclically. In this way, control signals are transmitted to the drive and status messages are reported.
4. Program your application with the FB 860 - VMC_AxisControl or with the PLCopen blocks.

4.4.4 Copy project

Proceeding

In the example, the station *'Source'* is copied and saved as *'Target '*.

1. Open the hardware configuration of the *'Source'* CPU and start the *SPEED7 EtherCAT Manager*.
2. In the *SPEED7 EtherCAT Manager*, via *'File ➔ Save as'* save the configuration in your working directory.
3. Close the *SPEED7 EtherCAT Manager* and the hardware configurator.
4. Copy the station *'Source'* with Ctrl + C and paste it as *'Target'* into your project with Ctrl + V.

5. Select the *'Blocks'* directory of the *'Target'* CPU and delete the *'System data'*.
6. Open the hardware configuration of the *'Target'* CPU. Adapt the IP address data or re-network the CPU or the CP again.

Before calling the SPEED7 EtherCAT Manager you have always to save your project with 'Station ➔ Save and compile'.

7. Safe your project with *'Station ➔ Safe and compile'*.
8. Open the *SPEED7 EtherCAT Manager*.
9. Use *'File ➔ Open'* to load the configuration from your working directory.
10. Close the *SPEED7 EtherCAT Manager*.
11. Save and compile your configuration.

4.5 Drive specific blocks

4.5.1 UDT 872 - VMC_ConfigSigma7EC_REF - *Sigma-7* EtherCAT Data structure axis configuration

This is a user-defined data structure that contains information about the configuration data. The UDT is specially adapted to the use of a *Sigma-7* drive, which is connected via EtherCAT.

4.5.2 FB 872 - VMC_KernelSigma7_EC - *Sigma-7* EtherCAT Kernel

Description

This block converts the drive commands for a *Sigma-7* axis via EtherCAT and communicates with the drive. For each *Sigma-7* axis, an instance of this FB is to be cyclically called.

Please note that this module calls the SFB 238 internally.

In the SPEED7 Studio, this module is automatically inserted into your project.

In Siemens SIMATIC Manager, you have to copy the SFB 238 from the Motion Control Library into your project.

Parameter	Declaration	Data type	Description
Init	INPUT	BOOL	The block is internally reset with an edge 0-1. Existing motion commands are aborted and the block is initialized.
Config	IN_OUT	UDT872	Data structure for transferring axis-dependent configuration data to the *AxisKernel*.
Axis	IN_OUT	MC_AXIS_REF	Data structure for transferring axis-dependent information to the *AxisKernel* and PLCopen blocks.

4.5.3 FB 873 - VMC_InitSigma7S_EC - *Sigma-7S* EtherCAT Initialization

Description

This block is used to configure the axis. The module is specially adapted to the use of a *Sigma-7* drive, which is connected via EtherCAT.

Parameter	Declaration	Data type	Description
Config	IN_OUT	UDT872	Data structure for transferring axis-dependent configuration data to the *AxisKernel*.
Axis	IN_OUT	MC_AXIS_REF	Data structure for transferring axis-dependent information to the *AxisKernel* and PLCopen blocks.
Enable	INPUT	BOOL	Release of initialization
Logical address	INPUT	INT	Start address of the PDO input data
InputsStartAddressPDO	INPUT	INT	Start address of the input PDOs
OutputsStartAddressPDO	INPUT	INT	Start address of the output PDOs

Parameter	Declaration	Data type	Description
EncoderType	INPUT	INT	Encoder type ■ 1: Absolute encoder ■ 2: Incremental encoder
EncoderResolutionBits	INPUT	INT	Number of bits corresponding to one encoder revolution. Default: 20
FactorPosition	INPUT	REAL	Factor for converting the position of user units [u] into drive units [increments] and back. It's valid: $p_{[increments]} = p_{[u]}$ x *FactorPosition* Please consider the factor which can be specified on the drive via the objects 0x2701: 1 and 0x2701: 2. This should be 1.
Velocity Factor	INPUT	REAL	Factor for converting the speed of user units [u/s] into drive units [increments/s] and back. It's valid: $v_{[increments/s]} = v_{[u/s]}$ x *FactorVelocity* Please also take into account the factor which you can specify on the drive via objects 0x2702: 1 and 0x2702: 2. This should be 1.
FactorAcceleration	INPUT	REAL	Factor to convert the acceleration of user units [u/s^2] in drive units [10^{-4} x increments/s^2] and back. It's valid: 10^{-4} x $a_{[increments/s^2]} = a_{[u/s^2]}$ x *FactorAcceleration* Please also take into account the factor which you can specify on the drive via objects 0x2703: 1 and 0x2703: 2. This should be 1.
OffsetPosition	INPUT	REAL	Offset for the zero position [u].
MaxVelocityApp	INPUT	REAL	Maximum application speed [u/s]. The command inputs are checked to the maximum value before execution.
MaxAccelerationApp	INPUT	REAL	Maximum acceleration of application [u/s^2]. The command inputs are checked to the maximum value before execution.
MaxDecelerationApp	INPUT	REAL	Maximum application delay [u/s^2]. The command inputs are checked to the maximum value before execution.
MaxPosition	INPUT	REAL	Maximum position for monitoring the software limits [u].
MinPosition	INPUT	REAL	Minimum position for monitoring the software limits [u].
EnableMaxPosition	INPUT	BOOL	Monitoring maximum position ■ TRUE: Activates the monitoring of the maximum position.
EnableMinPosition	INPUT	BOOL	Monitoring minimum position ■ TRUE: Activation of the monitoring of the minimum position.
MinUserPosition	OUTPUT	REAL	Minimum user position based on the minimum encoder value of 0x80000000 and the *FactorPosition* [u].

Parameter	Declaration	Data type	Description
MaxUserPosition	OUTPUT	REAL	Maximum user position based on the maximum encoder value of 0x7FFFFFFF and the *FactorPosition* [u].
Valid	OUTPUT	BOOL	Initialization ■ TRUE: Initialization is valid.
Error	OUTPUT	BOOL	■ Error – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*. The axis is disabled.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

5 Usage *Sigma-7W* EtherCAT

5.1 Overview

Usage of the single-axis drive *Chapter 4 'Usage Sigma-7S EtherCAT' on page 46*

Precondition

- SPEED7 Studio from V1.6.1
  or
- Siemens SIMATIC Manager from V 5.5, SP2 & *SPEED7 EtherCAT Manager* & *Simple Motion Control Library*
- CPU with EtherCAT master, such as CPU 015-CEFNR00
- *Sigma-7W* Double-axis drive with EtherCAT option card

Steps of configuration

1. Set the parameters on the drive
   - The setting of the parameters happens by means of the software tool *Sigma Win+*.
2. Hardware configuration in VIPA *SPEED7 Studio* or Siemens SIMATIC Manager
   - Configuring a CPU with EtherCAT master functionality
   - Configuration of the *Sigma-7W* EtherCAT double axes.
   - Configuring the EtherCAT connection via *SPEED7 EtherCAT Manager*
3. Programming in VIPA *SPEED7 Studio* or Siemens SIMATIC Manager
   - *Init* block for the configuration of the double axes.
   - *Kernel* block for communication with one axis each.
   - Connecting the blocks for motion sequences.

5.2 Set the parameters on the drive

CAUTION!
Before the commissioning, you have to adapt your drive to your application with the *Sigma Win+* software tool! More may be found in the manual of your drive.

The following parameters must be set via *Sigma Win+* to match the *Simple Motion Control Library*:

Axis 1 - Module 1 (24bit encoder)

Servopack Parameter	Address	Name	Value
Pn205	(2205h)	Multiturn Limit Setting	65535
Pn20E	(220Eh)	ElectronicGear Ratio (Numerator)	16
Pn210	(2210h)	Electronic Gear Ratio (Denominator)	1
PnB02	(2701h:01)	Position User Unit (Numerator)	1
PnB04	(2701h:02)	Position User Unit (Denominator)	1
PnB06	(2702h:01)	Velocity User Unit (Numerator)	1
PnB08	(2702h:02)	Velocity User Unit (Denominator)	1

Servopack Parameter	Address	Name	Value
PnB0A	(2703h:01)	Acceleration User Unit (Numerator)	1
PnB0C	(2703h:02)	Acceleration User Unit (Denominator)	1

Achse 2 - Module 2 (24Bit Encoder)

Servopack Parameter	Address	Name	Value
Pn205	(2A05h)	Multiturn Limit Setting	65535
Pn20E	(2A0Eh)	ElectronicGear Ratio (Numerator)	16
Pn210	(2A10h)	Electronic Gear Ratio (Denominator)	1
PnB02	(2F01h:01)	Position User Unit (Numerator)	1
PnB04	(2F01h:02)	Position User Unit (Denominator)	1
PnB06	(2F02h:01)	Velocity User Unit (Numerator)	1
PnB08	(2F02h:02)	Velocity User Unit (Denominator)	1
PnB0A	(2F03h:01)	Acceleration User Unit (Numerator)	1
PnB0C	(2F03h:02)	Acceleration User Unit (Denominator)	1

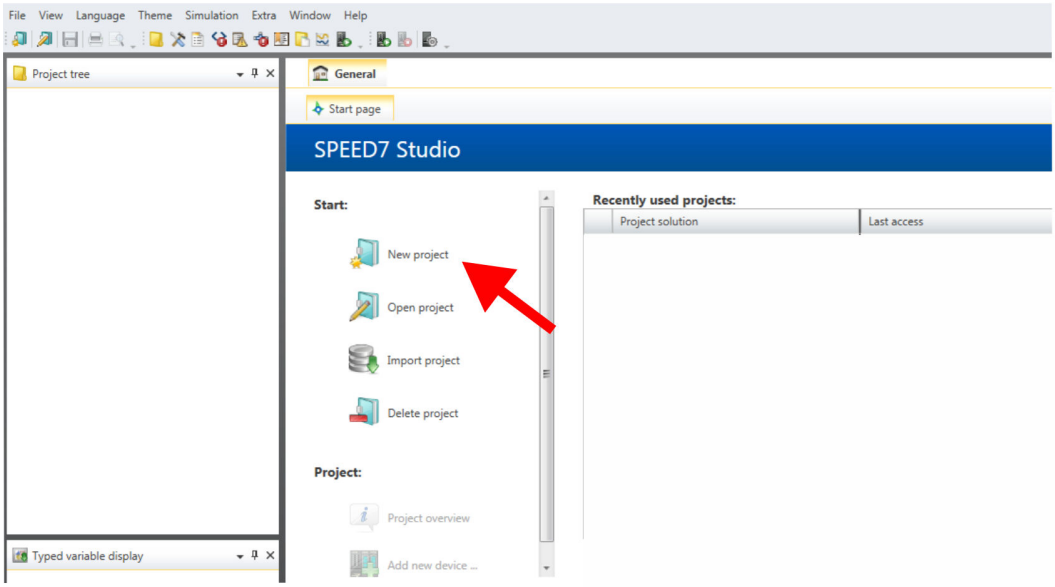
5.3 Usage in VIPA *SPEED7 Studio*

5.3.1 Hardware configuration

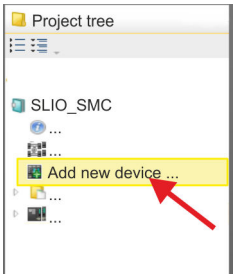
Add CPU in the project

Please use for configuration the*SPEED7 Studio* V1.6.1 and up.

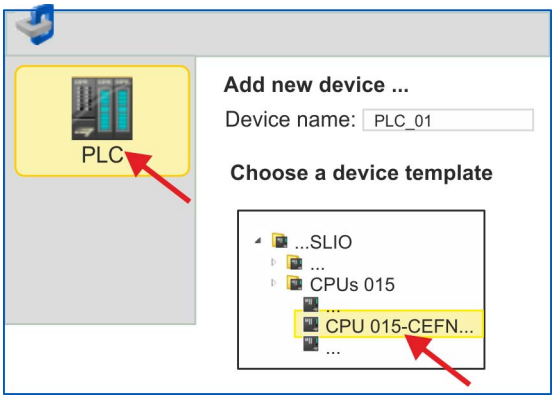
1. Start the *SPEED7 Studio*.



2. Create a new project at the start page with *'New project'*.
   ⇨ A new project is created and the view *'Devices and networking'* is shown.



3. Click in the *Project tree* at *'Add new device ...'*.

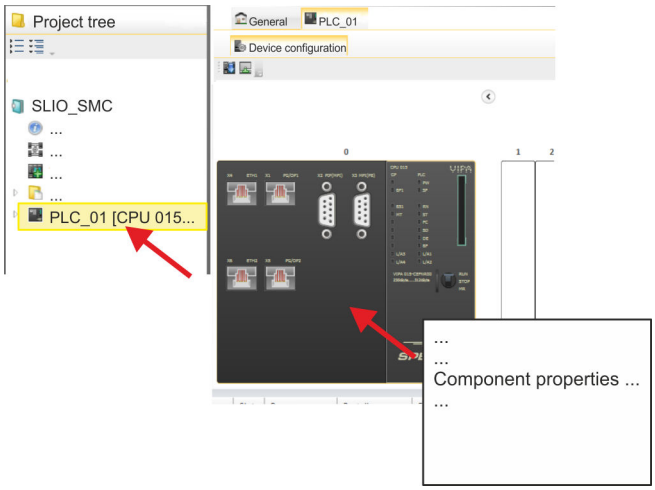


⇨ A dialog for device selection opens.

4. Select from the *'Device templates'* a CPU with EtherCAT master functions such as CPU 015-CEFNR00 and click at [OK].

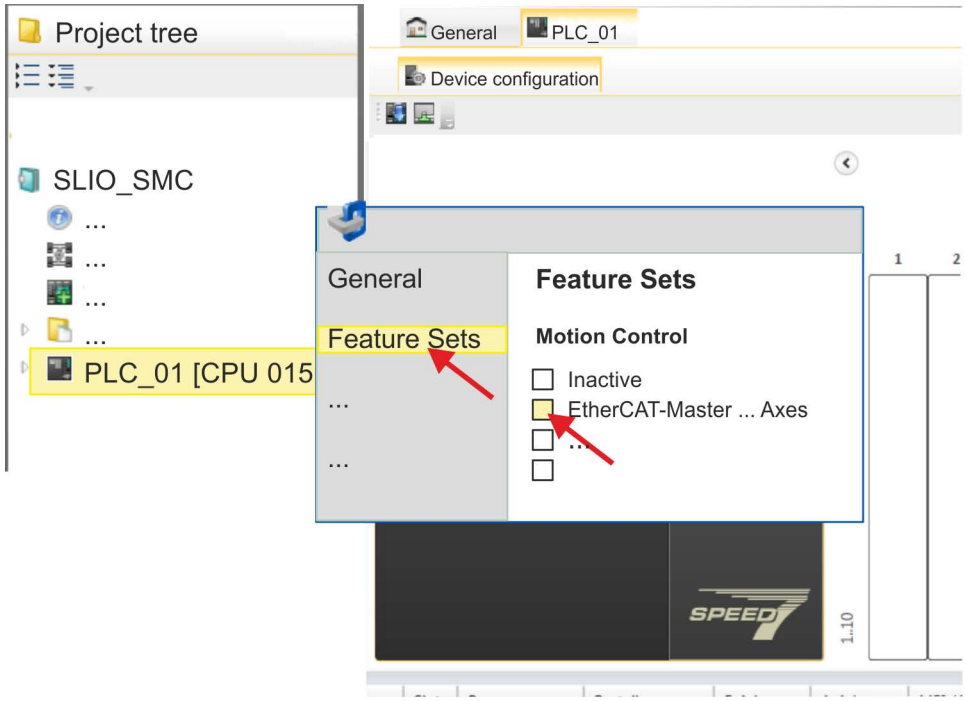
⇨ The CPU is inserted in *'Devices and networking'* and the *'Device configuration'* is opened.

Activate motion control functions



1. Click at the CPU in the *'Device configuration'* and select *'Context menu ➔ Components properties'*.

⇨ The properties dialog of the CPU is opened.

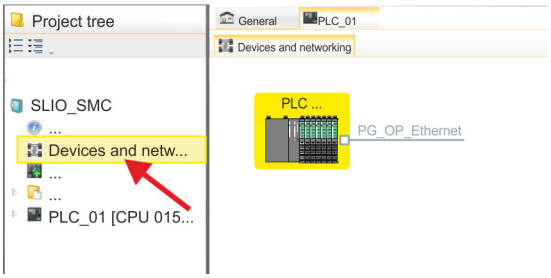


2. Click at *'Feature Sets'* and activate at *'Motion Control'* the parameter *'EtherCAT-Master... Axes'*. The number of axes is not relevant in this example.
3. Confirm your input with [OK].
   ⇨ The motion control functions are now available in your project.

CAUTION!
Please note due to the system, with every change to the feature set settings, the EtherCAT field bus system and its motion control configuration will be deleted from your project!

Configuration of Ethernet PG/OP channel

1. Click in the *Project tree* at *'Devices and networking'*.
   ⇨ You will get a graphical object view of your CPU.



2. Click at the network *'PG_OP_Ethernet'*.
3. Select *'Context menu ➔ Interface properties'*.
   ⇨ A dialog window opens. Here you can enter the IP address data for your Ethernet PG/OP channel. You get valid IP address parameters from your system administrator.

4. Confirm with [OK].

   ⇨ The IP address data are stored in your project listed in *'Devices and networking'* at *'Local components'*.

   After transferring your project your CPU can be accessed via Ethernet PG/OP channel with the set IP address data.

Installing the ESI file

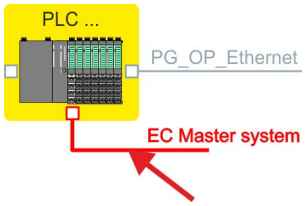
For the *Sigma-7* EtherCAT drive can be configured in the *SPEED7 EtherCAT Manager*, the corresponding ESI file must be installed. Usually, the *SPEED7 Studio* is delivered with current ESI files and you can skip this part. If your ESI file is not up-to date, you will find the latest ESI file for the *Sigma-7* EtherCAT drive under www.yaskawa.eu.com at *'Service ➔ Drives & Motion Software'*.

1. Download the according ESI file for your drive. Unzip this if necessary.
2. Navigate to your *SPEED7 Studio*.
3. Open the corresponding dialog window by clicking on *'Extra ➔ Install device description (EtherCAT - ESI)'*.
4. Under *'Source path'*, specify the ESI file and install it with [Install].

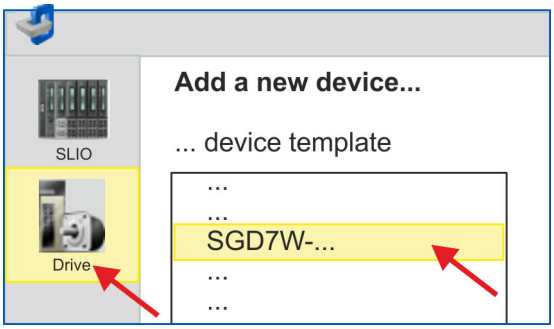
   ⇨ The devices of the ESI file are now available.

***Sigma-7W* add a double-axis drive**

1. Click in the Project tree at *'Devices and networking'*.
2. Click here at *'EC-Mastersystem'* and select *'Context menu ➔ Add new device'*.



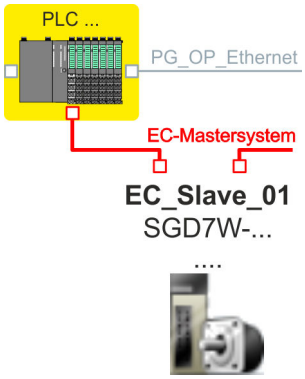
   ⇨ The device template for selecting an EtherCAT device opens.



3. Select your *Sigma-7W* double-axis drive:

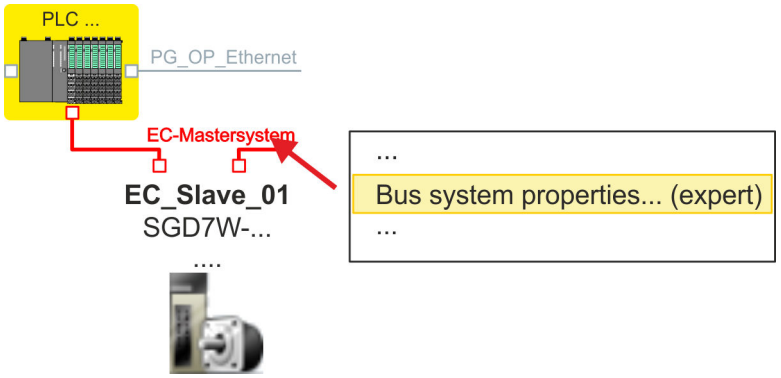
- SGD7W-xxxxA0 ...

Confirm your input with [OK]. If your drive does not exist, you must install the corresponding ESI file as described above.



⇨ The *Sigma-7W* double-axis drive is connected to your EC master system.

Configure *Sigma-7W* double-axis drive

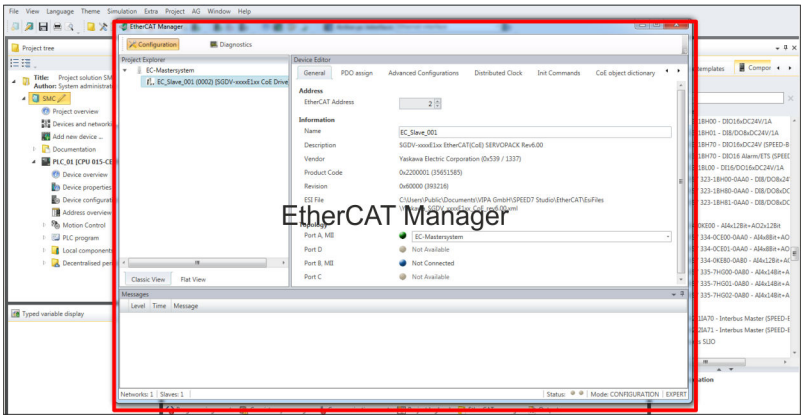


1. Click here at *'EC-Mastersystem'* and select *'Context menu ➔ Bus system properties (expert)'*.

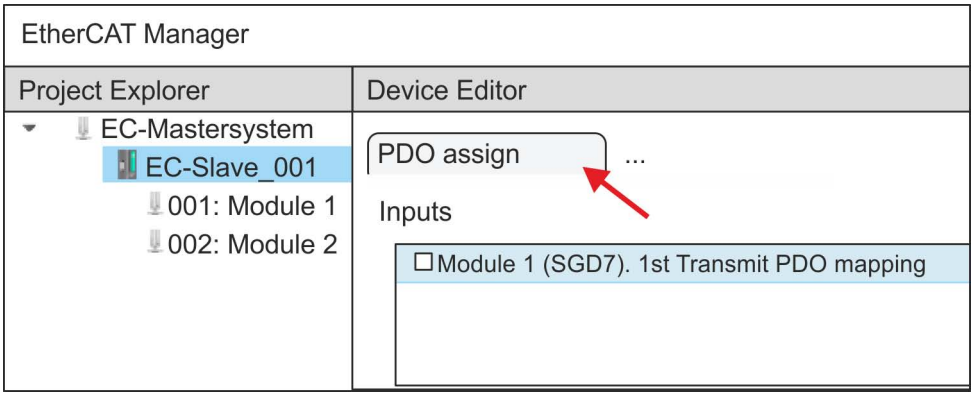
> *You can only edit PDOs in 'Expert mode'! Otherwise, the buttons are hidden.*

⇨ The *SPEED7 EtherCAT Manager* opens. Here you can configure the EtherCAT communication to your *Sigma-7W* double-axis drive.

More information about the usage of the *SPEED7 EtherCAT Manager* may be found in the online help of the *SPEED7 Studio*.



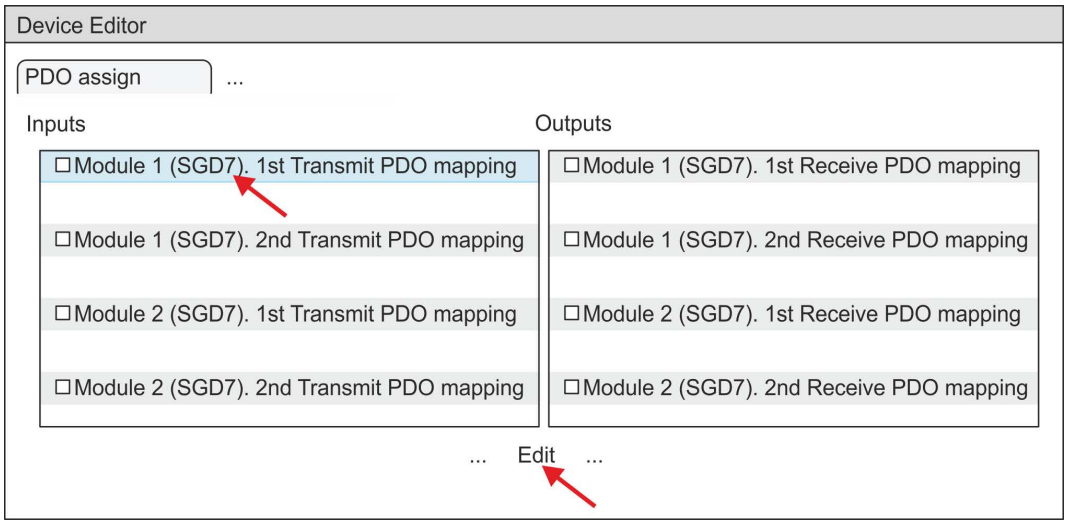
2. Click on the slave in the *SPEED7 EtherCAT Manager* and select the *'PDO assign'* tab in the *'Device editor'*.



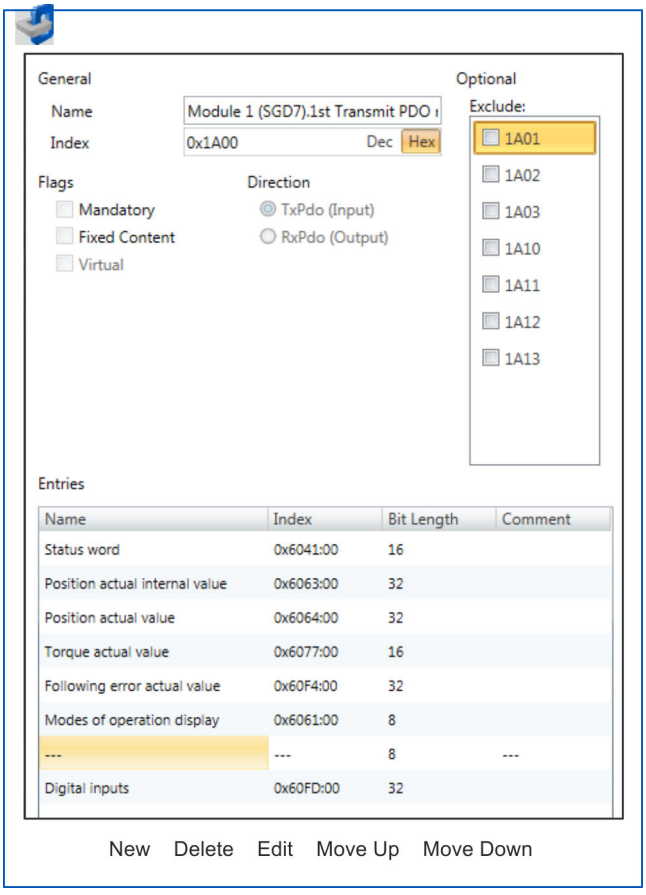
⇨ This dialogue shows a list of the PDOs for *'Module 1'* (axis 1) and *'Module 2'* (axis 2).

3. By selecting the appropriate mapping, you can edit the PDOs with [Edit]. Select the mapping *'Module 1 (SGD7). 1st Transmit PDO mapping'* and click at [Edit].

Please note that some PDOs can not be edited because of the default settings. By de-activating already activated PDOs, you can release the processing of locked PDOs.



⇨ The dialog *'Edit PDO'* is opened. Please check the PDO settings listed here and adjust them if necessary. Please also take into account the order of the *'Entries'* and add them accordingly.



The following functions are available for editing the *'Entries'*:

- New
  - Here you can create a new entry in a dialog by selecting the corresponding entry from the *'CoE object dictionary'* and making your settings. The entry is accepted with [OK] and is listed in the list of entries.
- Delete
  - This allows you to delete a selected entry.
- Edit
  - This allows you to edit the general data of an entry.
- Move Up/Down
  - This allows you to move the selected entry up or down in the list.

4. Perform the following settings for the Transmit PDOs:

Inputs: 1st Transmit PDO

Module 1 (SGD7). 1st Transmit PDO mapping	Module 2 (SGD7). 1st Transmit PDO mapping
Name: Module 1 (SGD7). 1st Transmit PDO mapping	Name: Module 2 (SGD7). 1st Transmit PDO mapping
Index: 0x1A00	Index: 0x1A10
Flags: Everything de-activated	
Direction TxPdo (Input): activated	
Exclude: 1A01: de-activated	1A11: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

Entries	Module 1 (axis 1)	Module 2 (axis 2)	Bit length
Name	Index	Index	
Status word	0x6041:00	0x6841: 00	16bit
Position actual internal value	0x6063:00	0x6863:00	32bit
Position actual value	0x6064:00	0x6864:00	32bit
Torque actual value	0x6077:00	0x6877:00	16bit
Following error actual value	0x60F4:00	0x68F4:00	32bit
Modes of operation display	0x6061:00	0x6861:00	8bit
---	---	---	8bit
Digital inputs	0x60FD:00	0x68FD:00	32bit

Inputs: 2nd Transmit PDO

Module 1 (SGD7). 2nd Transmit PDO mapping	Module 2 (SGD7). 2nd Transmit PDO mapping
Name: Module 1 (SGD7). 2nd Transmit PDO mapping	Name: Module 2 (SGD7). 2nd Transmit PDO mapping
Index: 0x1A01	Index: 0x1A11
Flags: Everything de-activated	
Direction TxPdo (Input): activated	
Exclude: 1A00, 1A02, 1A03: de-activated	1A10, 1A12, 1A13: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

Entries	Module 1 (axis 1)	Module 2 (axis 2)	Bit length
Name	Index	Index	
Touch probe status	0x60B9:00	0x68B9:00	16bit
Touch probe 1 position value	0x60BA:00	0x68BA:00	32bit
Touch probe 2 position value	0x60BC:00	0x68BC:00	32bit
Velocity actual value	0x606C:00	0x686C:00	32bit

5. Perform the following settings for the Receive PDOs:

Outputs: 1st Receive PDO

Module 1 (SGD7). 1st Receive PDO	**Module 2 (SGD7). 1st Receive PDO**
Name: Module 1 (SGD7). 1st Receive PDO mapping	Name: Module 2 (SGD7). 1st Receive PDO mapping
Index: 0x1600	Index: 0x1610
Flags: Everything de-activated	
Direction RxPdo (Output): activated	
Exclude: 1601, 1602, 1603: de-activated	1611, 1612, 1613: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

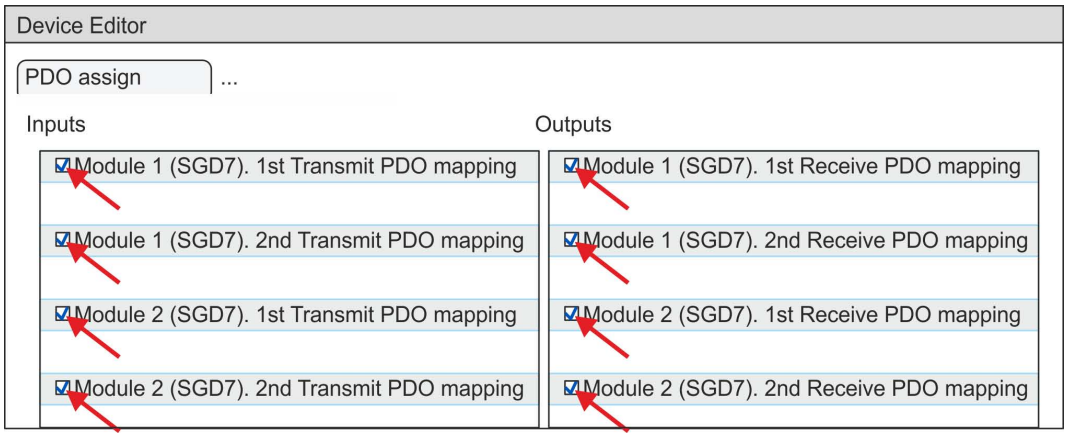
Entries	**Module 1 (axis 1)**	**Module 2 (axis 2)**	**Bit length**
Name	**Index**	**Index**	
Control word	0x6040:00	0x6840: 00	16bit
Target position	0x607A:00	0x687A: 00	32bit
Target velocity	0x60FF:00	0x68FF: 00	32bit
Modes of operation	0x6060:00	0x6860: 00	8bit
---	---	---	8bit
Touch probe function	0x60B8:00	0x68B8: 00	16bit

Outputs: 2nd Receive PDO

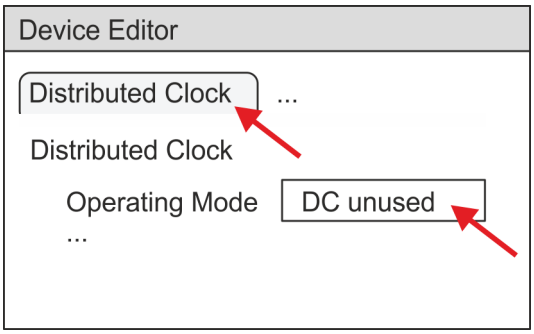
Module 1 (SGD7). 2nd Receive PDO	**Module 2 (SGD7). 2nd Receive PDO**
Name: Module 1 (SGD7). 2nd Receive PDO mapping	Name: Module 2 (SGD7). 2nd Receive PDO mapping
Index: 0x1601	Index: 0x1611
Flags: Everything de-activated	
Direction RxPdo (Output): activated	
Exclude: 1600, 1602, 1603: de-activated	1610, 1612, 1613: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

Entries	**Module 1 (axis 1)**	**Module 2 (axis 2)**	**Bit length**
Name	**Index**	**Index**	
Profile velocity	0x6081:00	0x6881: 00	32bit
Profile acceleration	0x6083:00	0x6883: 00	32bit
Profile deceleration	0x6084:00	0x6884: 00	32bit

6. For *'Module 1'* and *'Module 2'* in PDO assignment, activate the PDOs 1 and 2 for the inputs and outputs. All subsequent PDOs must remain de-activated. If this is not possible, please check the respective PDO parameter *'Exclude'*.

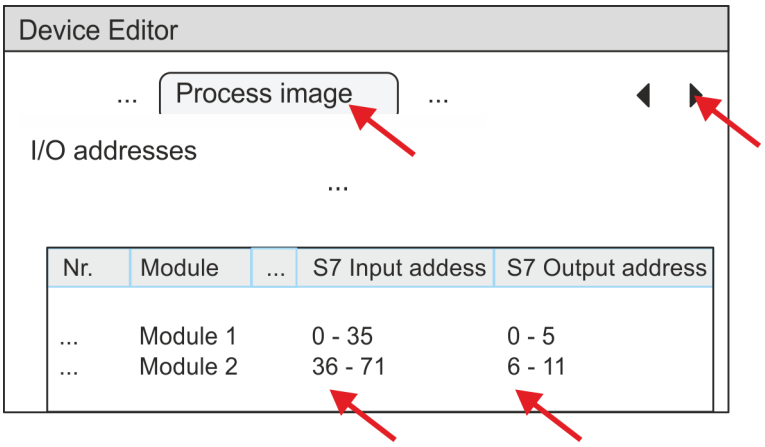


7. In the *'Device Editor'* of the *SPEED7 EtherCAT Manager*, select the *'Distributed clocks'* tab and set *'DC unused'* as *'Operating mode'*.



8. Select the *'Process image'* tab in the *'device editor'* using the arrow key and note the following PDO start addresses for the parameters of the block FB 874 - VMC_InitSigma7W_EC:

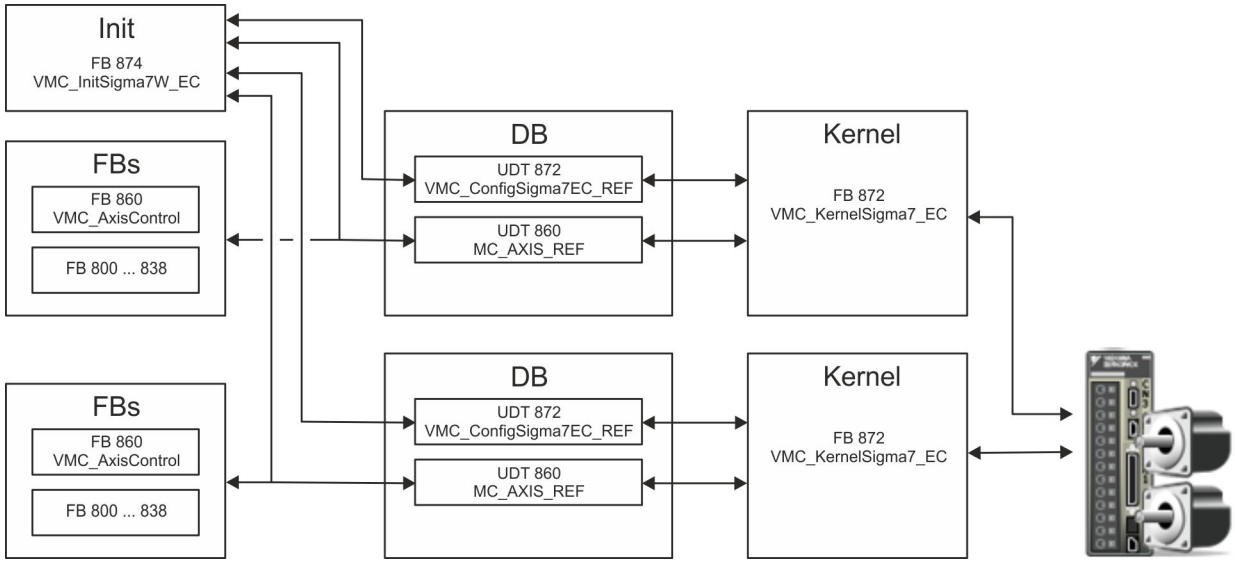
- Module 1: *'S7 Input address'* → *'M1_PdoInputs'* (here 0)
- Module 2: *'S7 Input address'* → *'M2_PdoInputs'* (here 36)
- Module 1: *'S7 Output address'* → *'M1_PdoOutputs'* (here 0)
- Module 2: *'S7 Output address'* → *'M2_PdoOutputs'* (here 36)



9. By closing the dialog of the *SPEED7 EtherCAT Manager* with [X] the configuration is taken to the *SPEED7 Studio*.

5.3.2 User program

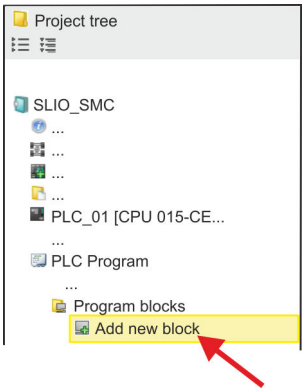
5.3.2.1 Overview



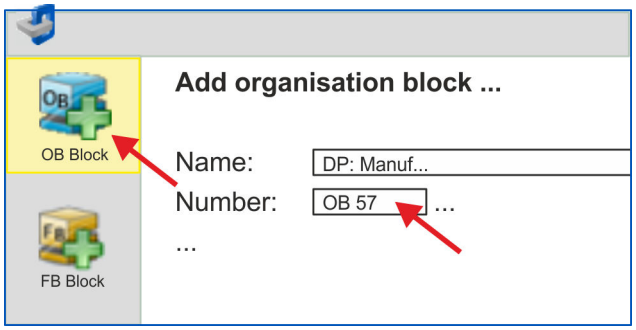
- DB
  A data block (axis DB) for configuration and status data must be created for each axis of a drive. The data block consists of the following data structures:
  – UDT 872 - *VMC_Config*Sigma7EC_REF
    The data structure describes the structure of the configuration of the drive.
    Specific data structure for *Sigma-7* EtherCAT.
  – UDT 860 - *MC_AXIS_REF*
    The data structure describes the structure of the parameters and status information of drives.
    General data structure for all drives and bus systems.
- FB 874 - *VMC_Init*Sigma7W_EC
  – The *Init*t block is used to configure the double-axis drive.
  – Specific block for *Sigma-7W* EtherCAT.
  – The configuration data for the initialization must be stored in the *axis DB*.
- FB 872 - *VMC_Kernel*Sigma7_EC
  – The *Kernel* block communicates with the drive via the appropriate bus system, processes the user requests and returns status messages.
  – The FB 872 - *VMC_Kernel*Sigma7_EC must be called for each axis.
  – Specific block for *Sigma-7* EtherCAT.
  – The exchange of the data takes place by means of the *axis DB*.
- FB 860 - *VMC_AxisControl*
  – General block for all drives and bus systems.
  – The FB 860 - *VMC_AxisControl* must be called for each axis.
  – Supports simple motion commands and returns all relevant status messages.
  – The exchange of the data takes place by means of the *axis DB*.
  – For motion control and status query, via the instance data of the block you can link a visualization.
  – In addition to the FB 860 - *VMC_AxisControl*, *PLCopen* blocks can be used.
- FB 800 ... FB 838 - PLCopen
  – The PLCopen blocks are used to program motion sequences and status queries.
  – The PLCopen blocks must be called for each axis.

5.3.2.2 Programming

Copy blocks into project

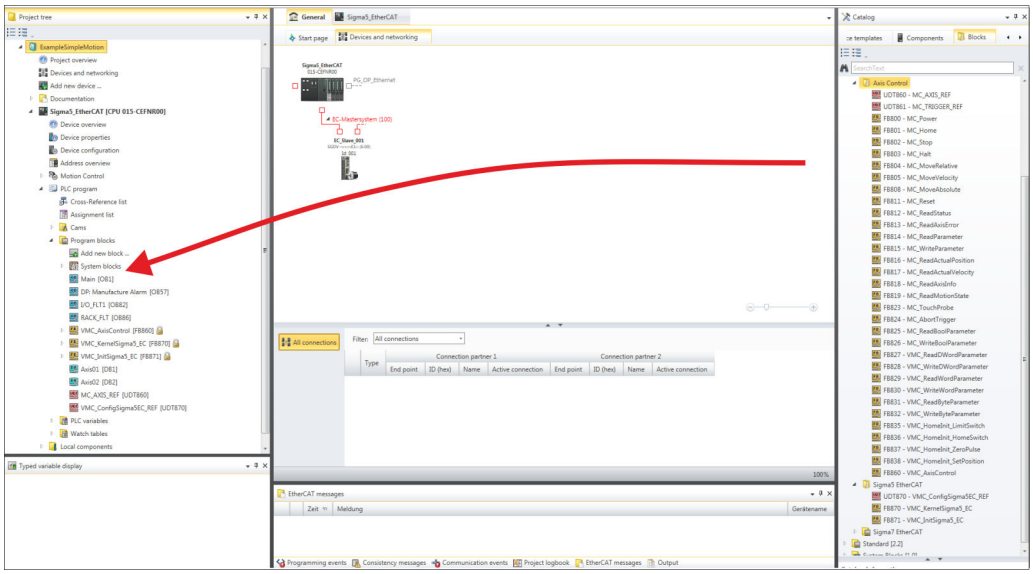


1. Click in the *Project tree* within the CPU at *'PLC program'*, *'Program blocks'* at *'Add New block'*.



⇨ The dialog *'Add block'* is opened.

2. Select the block type *'OB block'* and add one after the other OB 57, OB 82 and OB 86 to your project.



3. In the *'Catalog'*, open the *'Simple Motion Control'* library at *'Blocks'* and drag and drop the following blocks into *'Program blocks'* of the *Project tree*:
   - *Sigma-7* EtherCAT:
     - UDT 872 - VMC_ConfigSigma7EC_REF
     - FB 872 - VMC_KernelSigma7_EC
     - FB 874 - VMC_InitSigma7W_EC
   - Axis Control
     - UDT 860 - MC_AXIS_REF
     - Blocks for your movement sequences

Create axis DB for *'Module 1'*

1. Add a new DB as your *axis DB* to your project. Click in the *Project tree* within the CPU at *'PLC program'*, *'Program blocks'* at *'Add New block'*, select the block type *'DB block'* and assign the name "Axis01" to it. The DB number can freely be selected such as DB 10.

⇨ The block is created and opened.

2. ■ In "Axis01", create the variable "Config" of type UDT 872. These are specific axis configuration data.
   ■ In "Axis01", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

Axis01 [DB10]

Data block structure

		Addr...	Name	Data type	...
		...	Config	UDT	[872]
		...	Axis	UDT	[860]

Create axis DB for *'Module 2'*

1. Add another DB as your *axis DB* to your project and assign it the name "Axis02". The DB number can freely be selected such as DB 11.

   ⇨ The block is created and opened.

2. ■ In "Axis02", create the variable "Config" of type UDT 872. These are specific axis configuration data.
   ■ In "Axis02", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

Axis02 [DB11]

Data block structure

		Addr...	Name	Data type	...
		...	Config	UDT	[872]
		...	Axis	UDT	[860]

OB 1

Configuration of the double-axis

Open OB 1 and program the following FB calls with associated DBs:

FB 874 - VMC_InitSigma7W_EC, DB 874 *Chapter 5.5.3 'FB 874 - VMC_InitSigma7W_EC - Sigma-7W EtherCAT Initialization' on page 120*

At *M1/M2_PdoInputs* respectively *M1/M2_PdoOutputs*, enter the address from the *SPEED7 EtherCAT Manager* for the according axis. *95*

```
⇨ CALL  "VMC_InitSigma7W_EC" , "DI_InitSgm7WETC01"
  Enable                    :=TRUE
  LogicalAddress            :=0
  M1_PdoInputs              :=0 (EtherCAT-Manager
                                 Module1: S7 Input address)

  M1_PdoOutputs             :=0 (EtherCAT-Manager
                                 Module1: S7 Output address)
  M1_EncoderType            :=2
  M1_EncoderResolutionBits  :=20
  M1_FactorPosition         :=1.048576e+006
  M1_FactorVelocity         :=1.048576e+006
  M1_FactorAcceleration     :=1.048576e+002
  M1_OffsetPosition         :=0.000000e+000
  M1_MaxVelocityApp         :=5.000000e+001
  M1_MaxAccelerationApp     :=1.000000e+002
  M1_MaxDecelerationApp     :=1.000000e+002
  M1_MaxVelocityDrive       :=6.000000e+001
  M1_MaxAccelerationDrive   :=1.500000e+002
  M1_MaxDecelerationDrive   :=1.500000e+002
  M1_MaxPosition            :=1.048500e+003
  M1_MinPosition            :=-1.048514e+003
  M1_EnableMaxPosition      :=TRUE
  M1_EnableMinPosition      :=TRUE
  M2_PdoInputs              :=36 (EtherCAT-Manager
                                 Module2: S7 Input address)

  M2_PdoOutputs             :=36 (EtherCAT-Manager
                                 Module2: S7 Output address)
  M2_EncoderType            :=2
  M2_EncoderResolutionBits  :=20
  M2_FactorPosition         :=1.048576e+006
  M2_FactorVelocity         :=1.048576e+006
  M2_FactorAcceleration     :=1.048576e+002
  M2_OffsetPosition         :=0.000000e+000
  M2_MaxVelocityApp         :=5.000000e+001
  M2_MaxAccelerationApp     :=1.000000e+002
  M2_MaxDecelerationApp     :=1.000000e+002
  M2_MaxVelocityDrive       :=6.000000e+001
  M2_MaxAccelerationDrive   :=1.500000e+002
  M2_MaxDecelerationDrive   :=1.500000e+002
  M2_MaxPosition            :=1.048500e+003
  M2_MinPosition            :=-1.048514e+003
  M2_EnableMaxPosition      :=TRUE
  M2_EnableMinPosition      :=TRUE
  M1_MinUserPosition        :=-1000.0
  M1_MaxUserPosition        :=1000.0
  M2_MinUserPosition        :=-1000.0
  M2_MaxUserPosition        :=1000.0
  Valid                     :="InitS7WEC1_Valid"
  Error                     :="InitS7WEC1_Error"
```

```
ErrorID                 :="InitS7WEC1_ErrorID"
M1_Config               :="Axis01".Config
M1_Axis                 :="Axis01".Axis
M2_Config               :="Axis02".Config
M2_Axis                 :="Axis02".Axis
```

Connecting the kernel for the respective axis

The *Kernel* processes the user commands and passes them appropriately processed on to the drive via the respective bus system.

FB 872 - VMC_KernelSigma7_EC, DB 872 for axis 1

FB 872 - VMC_KernelSigma7_EC, DB 1872 for axis 2 ↳ *Chapter 5.5.2 'FB 872 - VMC_KernelSigma7_EC - Sigma-7 EtherCAT Kernel' on page 120*

```
⇨       CALL  "VMC_KernelSigma7_EC" , DB 872
         Init  :="KernelS7WEC1_Init"
         Config:="Axis01".Config
         Axis  :="Axis01".Axis

        CALL  "VMC_KernelSigma7_EC" , DB 1872
         Init  :="KernelS7WEC2_Init"
         Config:="Axis02".Config
         Axis  :="Axis02".Axis
```

Connecting the block for motion sequences

For simplicity, the connection of the FB 860 - VMC_AxisControl is to be shown here. This universal block supports simple motion commands and returns status messages. The inputs and outputs can be individually connected. Please specify the reference to the corresponding axis data at *'Axis'* in the *axis DB*.

FB 860 - VMC_AxisControl, DB 860 *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*

```
⇨       CALL  "VMC_AxisControl" , "DI_AxisControl01"
         SourceInputs     :="AxCtrl1_SourceInputs"
         AxisEnable       :="AxCtrl1_AxisEnable"
         AxisReset        :="AxCtrl1_AxisReset"
         HomeExecute      :="AxCtrl1_HomeExecute"
         HomePosition     :="AxCtrl1_HomePosition"
         StopExecute      :="AxCtrl1_StopExecute"
         MvVelocityExecute:="AxCtrl1_MvVelExecute"
         MvRelativeExecute:="AxCtrl1_MvRelExecute"
         MvAbsoluteExecute:="AxCtrl1_MvAbsExecute"
         PositionDistance :="AxCtrl1_PositionDistance"
         Velocity         :="AxCtrl1_Velocity"
         Acceleration     :="AxCtrl1_Acceleration"
         Deceleration     :="AxCtrl1_Deceleration"
         JogPositive      :="AxCtrl1_JogPositive"
         JogNegative      :="AxCtrl1_JogNegative"
         JogVelocity      :="AxCtrl1_JogVelocity"
         JogAcceleration  :="AxCtrl1_JogAcceleration"
         JogDeceleration  :="AxCtrl1_JogDeceleration"
         AxisReady        :="AxCtrl1_AxisReady"
         AxisEnabled      :="AxCtrl1_AxisEnabled"
         AxisError        :="AxCtrl1_AxisError"
         AxisErrorID      :="AxCtrl1_AxisErrorID"
         DriveWarning     :="AxCtrl1_DriveWarning"
         DriveError       :="AxCtrl1_DriveError"
         DriveErrorID     :="AxCtrl1_DriveErrorID"
         IsHomed          :="AxCtrl1_IsHomed"
         ModeOfOperation  :="AxCtrl1_ModeOfOperation"
         PLCopenState     :="AxCtrl1_PLCopenState"
         ActualPosition   :="AxCtrl1_ActualPosition"
         ActualVelocity   :="AxCtrl1_ActualVelocity"
         CmdDone          :="AxCtrl1_CmdDone"
         CmdBusy          :="AxCtrl1_CmdBusy"
         CmdAborted       :="AxCtrl1_CmdAborted"
         CmdError         :="AxCtrl1_CmdError"
         CmdErrorID       :="AxCtrl1_CmdErrorID"
         DirectionPositive:="AxCtrl1_DirectionPos"
         DirectionNegative:="AxCtrl1_DirectionNeg"
         SWLimitMinActive :="AxCtrl1_SWLimitMinActive"
         SWLimitMaxActive :="AxCtrl1_SWLimitMaxActive"
         HWLimitMinActive :="AxCtrl1_HWLimitMinActive"
         HWLimitMaxActive :="AxCtrl1_HWLimitMaxActive"
         Axis             :="Axis...".Axis
```

At *Axis*, enter "Axis01" for axis 1 and "Axis02" for axis 2.

For complex motion tasks, you can use the PLCopen blocks. Here you must also specify the reference to the corresponding axis data at Axis in the axis DB.

Your project now includes the following blocks:

- OB 1 - Main
- OB 57 - DP Manufacturer Alarm
- OB 82 - I/O_FLT1

- OB 86 - Rack_FLT
- FB 860 - VMC_AxisControl with instance DB
- FB 872 - VMC_KernelSigma7_EC with instance DB
- FB 874 - VMC_InitSigma7W_EC with instance DB
- UDT 860 - MC_Axis_REF
- UDT 872 - VMC_ConfigSigma7EC_REF

Sequence of operations

1. Select *'Project ➔ Compile all'* and transfer the project into your CPU. You can find more information on the transfer of your project in the online help of the *SPEED7 Studio*.

   ⇨ You can take your application into operation now.

CAUTION!
Please always observe the safety instructions for your drive, especially during commissioning!

2. Before the double-axis drive can be controlled, it must be initialized. To do this, call the *Init* block FB 874 - VMC_InitSigma7W_EC with *Enable* = TRUE.

   ⇨ The output *Valid* returns TRUE. In the event of a fault, you can determine the error by evaluating the *ErrorID*.

   You have to call the *Init* block again if you load a new axis DB or you have changed parameters on the *Init* block.

Do not continue until the Init block report an error!

3. Ensure that the *Kernel* block FB 872 - VMC_KernelSigma7_EC is called cyclically for each axis. In this way, control signals are transmitted to the drive and status messages are reported.
4. Program your application with the FB 860 - VMC_AxisControl or with the PLCopen blocks for each axis.

5.4 Usage in Siemens SIMATIC Manager

5.4.1 Precondition

Overview

- Please use for configuration the Siemens SIMATIC Manager V 5.5 SP2 and up.
- The configuration of the System SLIO CPU happens in the Siemens SIMATIC Manager by means of a virtual PROFINET IO device *'VIPA SLIO CPU'*. The *'VIPA SLIO CPU'* is to be installed in the hardware catalog by means of the GSDML.
- The configuration of the EtherCAT masters happens in the Siemens SIMATIC Manager by means of a virtual PROFINET IO device *'EtherCAT network'*. The *'EtherCAT network'* is to be installed in the hardware catalog by means of the GSDML.
- The *'EtherCAT network'* can be configured with the VIPA Tool *SPEED7 EtherCAT Manager*.
- For the configuration of the drive in the *SPEED7 EtherCAT Manager* the installation of the according ESI file is necessary.

Installing the IO device *'VIPA SLIO System'*

The installation of the PROFINET IO device *'VIPA SLIO CPU'* happens in the hardware catalog with the following approach:

1. Go to the service area of www.vipa.com.
2. Download the configuration file for your CPU from the download area via *'Config files ➔ PROFINET'*.
3. Extract the file into your working directory.
4. Start the Siemens hardware configurator.
5. Close all the projects.
6. Select *'Options ➔ Install new GSD file'*.
7. Navigate to your working directory and install the according GSDML file.
   ⇨ After the installation the according PROFINET IO device can be found at *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA SLIO System'*.

Installing the IO device EtherCAT network

The installation of the PROFINET IO devices *'EtherCAT Network'* happens in the hardware catalog with the following approach:

1. Go to the service area of www.vipa.com
2. Load from the download area at *'Config files ➔ EtherCAT'* the GSDML file for your EtherCAT master.
3. Extract the files into your working directory.
4. Start the Siemens hardware configurator.
5. Close all the projects.
6. Select *'Options ➔ Install new GSD file'*.
7. Navigate to your working directory and install the according GSDML file.
   ⇨ After the installation the *'EtherCAT Network'* can be found at *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA VIPA EtherCAT System'*.

Installing the *SPEED7 EtherCAT Manager*

The configuration of the PROFINET IO device *'EtherCAT Network'* happens by means of the *SPEED7 EtherCAT Manager* from VIPA. This may be found in the service area of www.vipa.com at *'Service/Support ➔ Downloads ➔ SPEED7'*.

The installation happens with the following proceeding:

1. Close the Siemens SIMATIC Manager.
2. Go to the service area of www.vipa.com
3. Load the *SPEED7 EtherCAT Manager* and unzip it on your PC.
4. For installation start the file EtherCATManager_v... .exe.
5. Select the language for the installation.
6. Accept the licensing agreement.
7. Select the installation directory and start the installation.
8. After installation you have to reboot your PC.
   ⇨ The *SPEED7 EtherCAT Manager* is installed and can now be called via the context menu of the Siemens SIMATIC Manager.

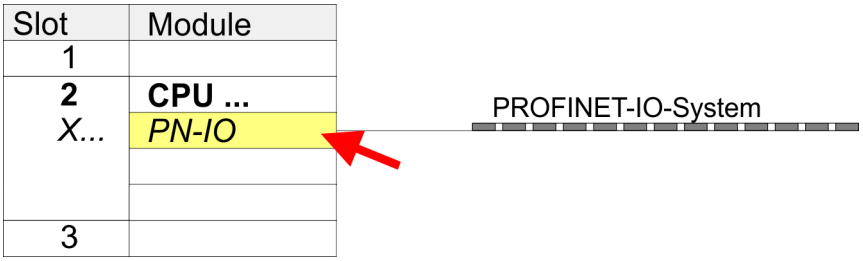
5.4.2 Hardware configuration

Configuring the CPU in the project

Slot	Module
1	
2	**CPU 315-2 PN/DP**
X1	*MPI/DP*
X2	*PN-IO*
X2...	*Port 1*
X2...	*Port 2*
3	

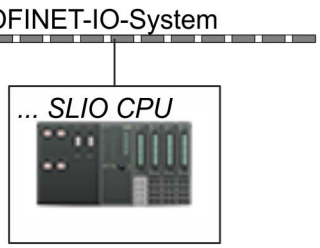
To be compatible with the Siemens SIMATIC Manager the following steps should be executed:

1. Start the Siemens hardware configurator with a new project.
2. Insert a profile rail from the hardware catalog.
3. Place at *'Slot'* number 2 the CPU 315-2 PN/DP (315-2EH14 V3.2).
4. The integrated PROFIBUS DP master (jack X3) is to be configured and connected via the sub module *'X1 MPI/DP'*.
5. The integrated EtherCAT master is to be configured via the sub module *'X2 PN-IO'* as a virtual PROFINET network.
6. Click at the sub module *'PN-IO'* of the CPU.
7. Select *'Context menu ➔ Insert PROFINET IO System'*.



8. Create with [New] a new sub net and assign valid address data
9. Click at the sub module *'PN-IO'* of the CPU and open with *'Context menu ➔ Properties'* the properties dialog.
10. Enter at *'General'* a *'Device name'*. The device name must be unique at the Ethernet subnet.

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	



Slot	Module	Order number	
0	***... SLIO CPU ...***	***015-...***	
X2	*015-...*		
1			
2			
3			
...			

11. Navigate in the hardware catalog to the directory *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA SLIO System'* and connect the IO device *'015-CFFNR00 CPU'* to your PROFINET system.

⇨ In the Device overview of the PROFINET IO device *'VIPA SLIO CPU'* the CPU is already placed at slot 0. From slot 1 you can place your System SLIO modules.

Configuration of Ethernet PG/OP channel

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	
4	343-1EX30
5	
...	

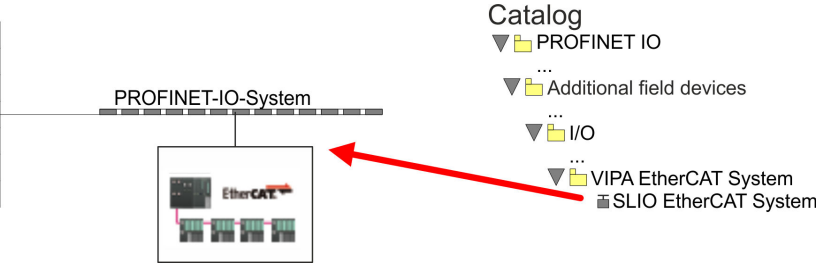
1. Place for the Ethernet PG/OP channel at slot 4 the Siemens CP 343-1 (SIMATIC 300 \ CP 300 \ Industrial Ethernet \CP 343-1 \ 6GK7 343-1EX30 0XE0 V3.0).

2. Open the properties dialog by clicking on the CP 343-1EX30 and enter for the CP at *'Properties'* the IP address data. You get valid IP address parameters from your system administrator.

3. Assign the CP to a *'Subnet'*. The IP address data are not accepted without assignment!

Insert *'EtherCAT network'*

Slot	Module
1	
2	**CPU ...**
X...	*PN-IO*
3	

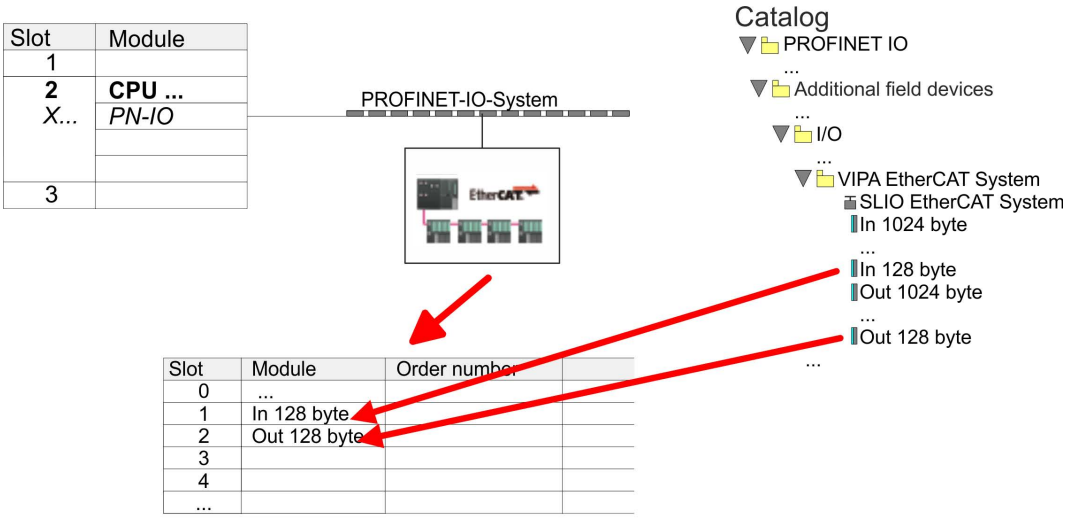


1. Navigate in the hardware catalog to the directory *'PROFINET IO ➔ Additional field devices ➔ I/O ➔ VIPA EtherCAT System'* and connect the IO device *'SLIO EtherCAT System'* to your PROFINET system.

2. Click at the inserted IO device *'EtherCAT Network'* and define the areas for in and output by drag and dropping the according *'Out'* or *'In'* area to a slot.

Create the following areas:

- In 128byte
- Out 128byte

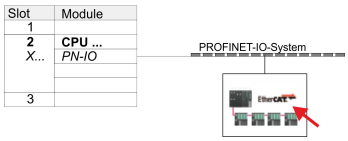


3. Select *'Station ➔ Save and compile'*

Configure *Sigma-7W* EtherCAT double-axis drive

The double-axis drive is configured in the *SPEED7 EtherCAT Manager*.

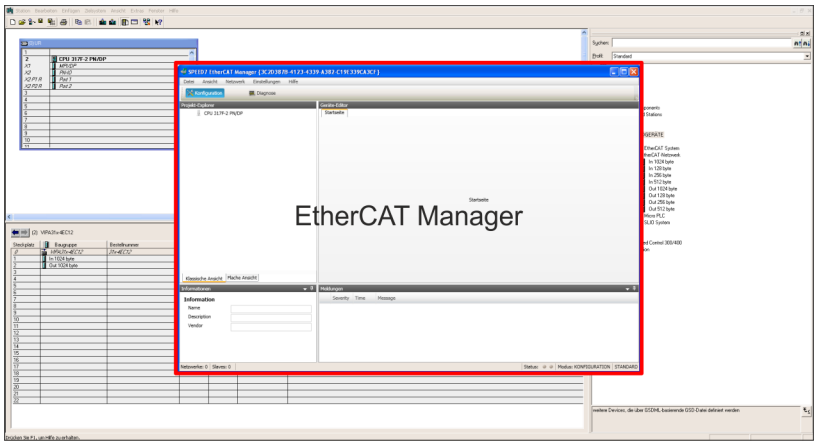
> *Before calling the SPEED7 EtherCAT Manager you have always to save your project with 'Station ➔ Save and compile'.*



1. Click at an inserted IO device *'EtherCAT Network'* and select *'Context menu ➔ Start Device-Tool ➔ SPEED7 EtherCAT Manager'*.

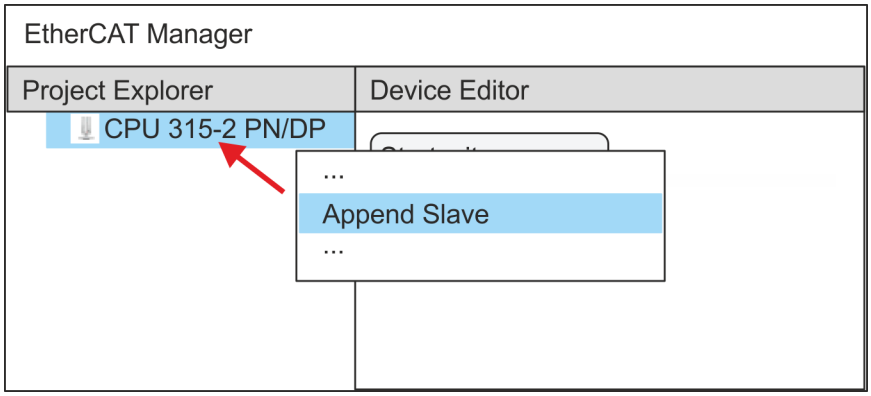
⇨ The *SPEED7 EtherCAT Manager* opens. Here you can configure the EtherCAT communication to your *Sigma-7W* EtherCAT double-axis drive.

More information about the usage of the *SPEED7 EtherCAT Manager* may be found in the according manual or online help.



3. For the *Sigma-7W* EtherCAT drive to be configured in the *SPEED7 EtherCAT Manager*, the corresponding ESI file must be installed. The ESI file for the *Sigma-7W* EtherCAT double-axis drive can be found under www.yaskawa.eu.com at *'Service ➔ Drives & Motion Software'*. Download the according ESI file for your drive. Unzip this if necessary.
4. Open in the *SPEED7 EtherCAT Manager* via *'File ➔ ESI Manager'* the dialogue window *'ESI Manager'*.
5. In the *'ESI Manager'* click at [Add File] and select your ESI file. With [Open], the ESI file is installed in the *SPEED7 EtherCAT Manager*.
6. Close the *'ESI Manager'*.

   ⇨ Your *Sigma-7W* EtherCAT double-axis drive is now available for configuration.

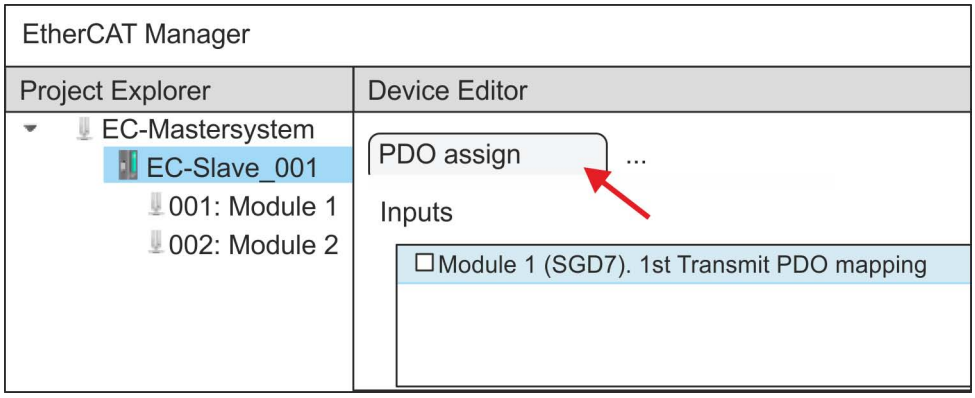


7. In the EtherCAT Manager, click on your CPU and open via *'Context menu ➔ Append Slave'* the dialog box for adding an EtherCAT slave.

   ⇨ The dialog window for selecting an EtherCAT slave is opened.
8. Select your *Sigma-7W* EtherCAT double-axis drive and confirm your selection with [OK].

   ⇨ The *Sigma-7W* EtherCAT double-axis drive is connected to the master and can now be configured.
9. *You can only edit PDOs in 'Expert mode'! Otherwise, the buttons are hidden. By activating the 'Expert mode' you can switch to advanced setting.*

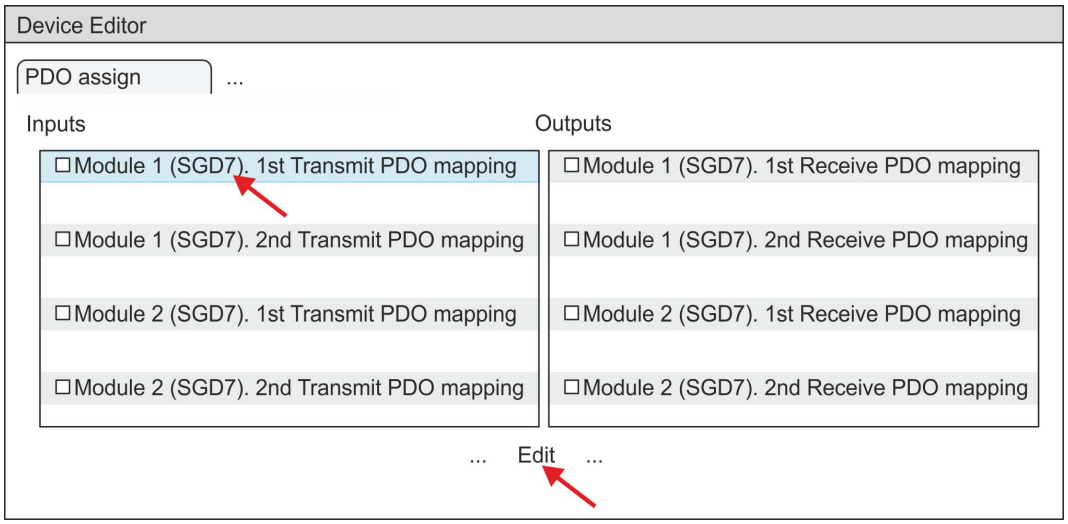
   By activating *'View ➔ Expert'* you can switch to the *Expert mode*.
10. Click on the *Sigma-7W* EtherCAT Slave in the *SPEED7 EtherCAT Manager* and select the *'PDO assign'* tab in the *'Device editor'*.



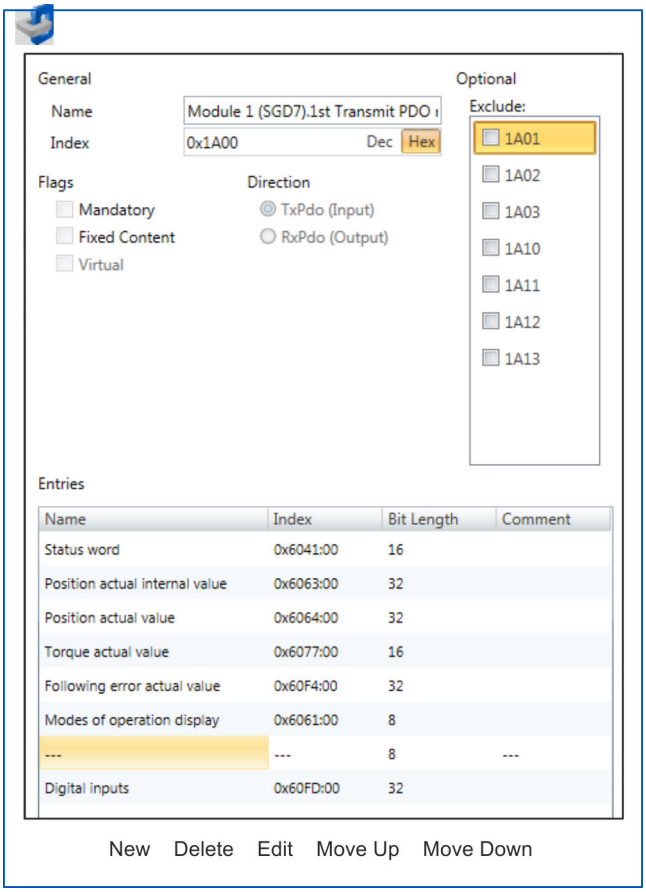
   ⇨ This dialogue shows a list of the PDOs.

11. By selecting the appropriate mapping, you can edit the PDOs with [Edit]. Select the mapping *'Module 1 (SGD7). 1st Transmit PDO mapping'* and click at [Edit].

Please note that some PDOs can not be edited because of the default settings. By de-activating already activated PDOs, you can release the processing of locked PDOs.



⇨ The dialog *'Edit PDO'* is opened. Please check the PDO settings listed here and adjust them if necessary. Please also take into account the order of the *'Entries'* and add them accordingly.



The following functions are available for editing the *'Entries'*:

- New
  - Here you can create a new entry in a dialog by selecting the corresponding entry from the *'CoE object dictionary'* and making your settings. The entry is accepted with [OK] and is listed in the list of entries.
- Delete
  - This allows you to delete a selected entry.
- Edit
  - This allows you to edit the general data of an entry.
- Move Up/Down
  - This allows you to move the selected entry up or down in the list.

12. Perform the following settings for the Transmit PDOs:

Inputs: 1st Transmit PDO

Module 1 (SGD7). 1st Transmit PDO mapping	**Module 2 (SGD7). 1st Transmit PDO mapping**
Name: Module 1 (SGD7). 1st Transmit PDO mapping	Name: Module 2 (SGD7). 1st Transmit PDO mapping
Index: 0x1A00	Index: 0x1A10
Flags: Everything de-activated	
Direction TxPdo (Input): activated	
Exclude: 1A01: de-activated	1A11: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

Entries	**Module 1 (axis 1)**	**Module 2 (axis 2)**	**Bit length**
Name	**Index**	**Index**	
Status word	0x6041:00	0x6841: 00	16bit
Position actual internal value	0x6063:00	0x6863:00	32bit
Position actual value	0x6064:00	0x6864:00	32bit
Torque actual value	0x6077:00	0x6877:00	16bit
Following error actual value	0x60F4:00	0x68F4:00	32bit
Modes of operation display	0x6061:00	0x6861:00	8bit
---	---	---	8bit
Digital inputs	0x60FD:00	0x68FD:00	32bit

Inputs: 2nd Transmit PDO

Module 1 (SGD7). 2nd Transmit PDO mapping	**Module 2 (SGD7). 2nd Transmit PDO mapping**
Name: Module 1 (SGD7). 2nd Transmit PDO mapping	Name: Module 2 (SGD7). 2nd Transmit PDO mapping
Index: 0x1A01	Index: 0x1A11
Flags: Everything de-activated	
Direction TxPdo (Input): activated	
Exclude: 1A00, 1A02, 1A03: de-activated	1A10, 1A12, 1A13: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

Entries	**Module 1 (axis 1)**	**Module 2 (axis 2)**	**Bit length**
Name	**Index**	**Index**	
Touch probe status	0x60B9:00	0x68B9:00	16bit
Touch probe 1 position value	0x60BA:00	0x68BA:00	32bit
Touch probe 2 position value	0x60BC:00	0x68BC:00	32bit
Velocity actual value	0x606C:00	0x686C:00	32bit

13. Perform the following settings for the Receive PDOs:

Outputs: 1st Receive PDO

Module 1 (SGD7). 1st Receive PDO	Module 2 (SGD7). 1st Receive PDO
Name: Module 1 (SGD7). 1st Receive PDO mapping	Name: Module 2 (SGD7). 1st Receive PDO mapping
Index: 0x1600	Index: 0x1610
Flags: Everything de-activated	
Direction RxPdo (Output): activated	
Exclude: 1601, 1602, 1603: de-activated	1611, 1612, 1613: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

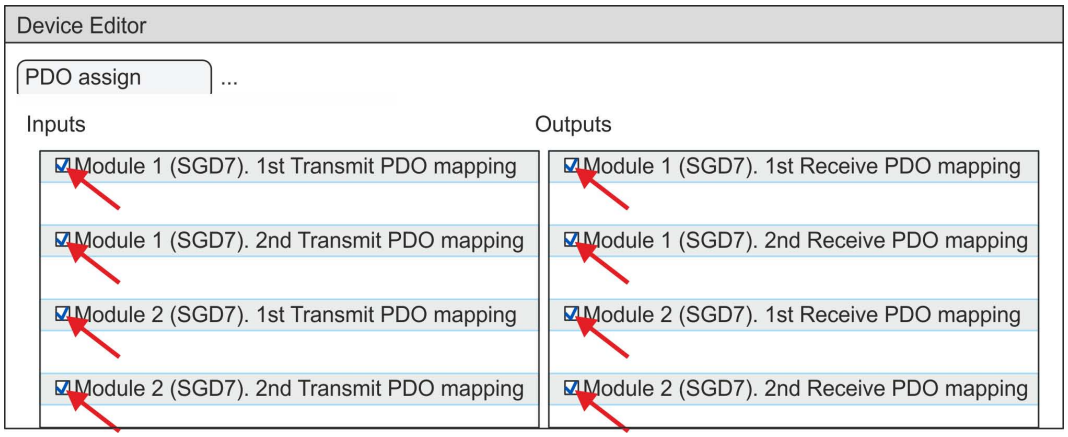
Entries	Module 1 (axis 1)	Module 2 (axis 2)	Bit length
Name	Index	Index	
Control word	0x6040:00	0x6840: 00	16bit
Target position	0x607A:00	0x687A: 00	32bit
Target velocity	0x60FF:00	0x68FF: 00	32bit
Modes of operation	0x6060:00	0x6860:00	8bit
---	---	---	8bit
Touch probe function	0x60B8:00	0x68B8: 00	16bit

Outputs: 2nd Receive PDO

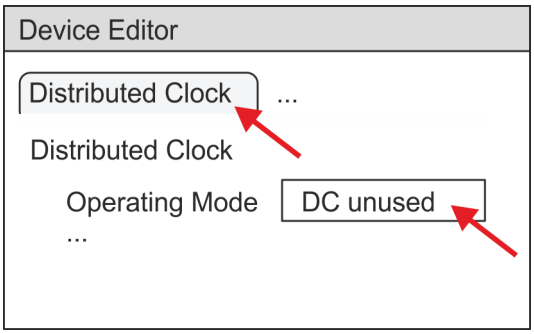
Module 1 (SGD7). 2nd Receive PDO	Module 2 (SGD7). 2nd Receive PDO
Name: Module 1 (SGD7). 2nd Receive PDO mapping	Name: Module 2 (SGD7). 2nd Receive PDO mapping
Index: 0x1601	Index: 0x1611
Flags: Everything de-activated	
Direction RxPdo (Output): activated	
Exclude: 1600, 1602, 1603: de-activated	1610, 1612, 1613: de-activated
Please note these settings, otherwise the PDO mappings can not be activated at the same time!	

Entries	Module 1 (axis 1)	Module 2 (axis 2)	Bit length
Name	Index	Index	
Profile velocity	0x6081:00	0x6881:00	32bit
Profile acceleration	0x6083:00	0x6883:00	32bit
Profile deceleration	0x6084:00	0x6884:00	32bit

14. For *'Module 1'* and *'Module 2'* in PDO assignment, activate the PDOs 1 and 2 for the inputs and outputs. All subsequent PDOs must remain de-activated. If this is not possible, please check the respective PDO parameter *'Exclude'*.

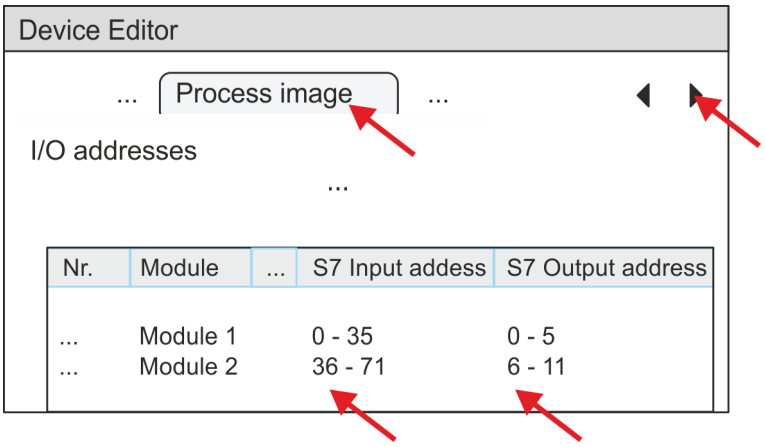


15. In the *'Device Editor'* of the *SPEED7 EtherCAT Manager*, select the *'Distributed clocks'* tab and set *'DC unused'* as *'Operating mode'*.



16. Select the *'Process image'* tab in the *'device editor'* using the arrow key and note the following PDO start addresses for the parameters of the block FB 874 - VMC_InitSigma7W_EC:

- Module 1: *'S7 Input address'* → *'M1_PdoInputs'* (here 0)
- Module 2: *'S7 Input address'* → *'M2_PdoInputs'* (here 36)
- Module 1: *'S7 Output address'* → *'M1_PdoOutputs'* (here 0)
- Module 2: *'S7 Output address'* → *'M2_PdoOutputs'* (here 36)

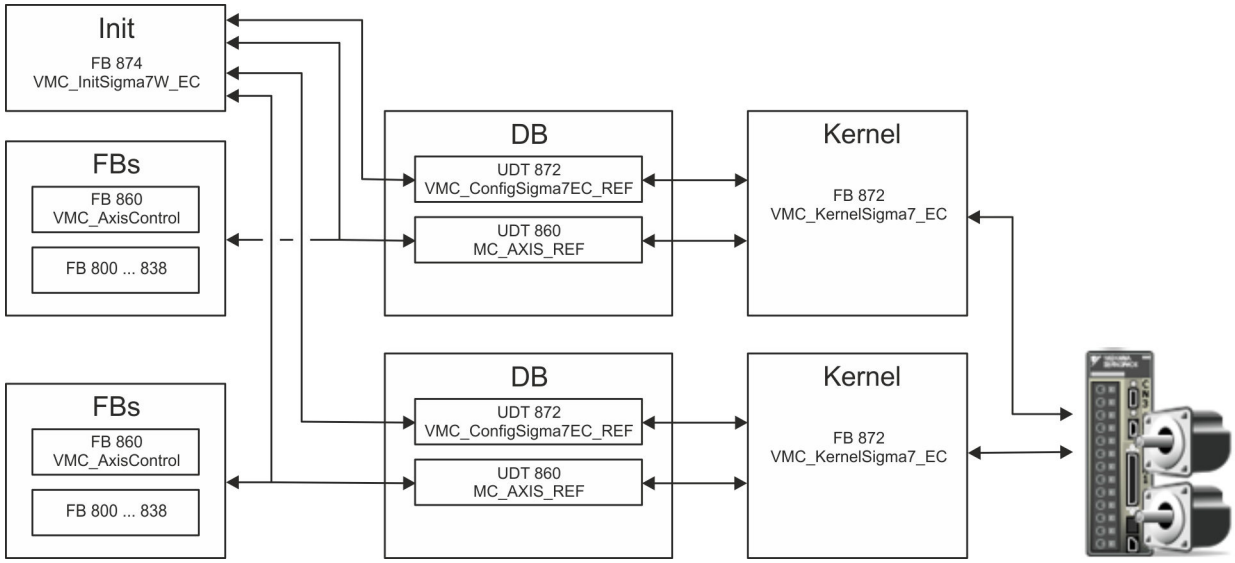


17. By closing the *SPEED7 EtherCAT Manager* the EtherCAT configuration is taken to the project. You can always edit your EtherCAT configuration in the *SPEED7 EtherCAT Manager*, since the configuration is stored in your project.

18. Save and compile your configuration.

5.4.3 User program

5.4.3.1 Overview



- DB
  A data block (axis DB) for configuration and status data must be created for each axis of a drive. The data block consists of the following data structures:
  – UDT 872 - *VMC_Config*Sigma7EC_REF
    The data structure describes the structure of the configuration of the drive.
    Specific data structure for *Sigma-7* EtherCAT.
  – UDT 860 - *MC_AXIS_REF*
    The data structure describes the structure of the parameters and status information of drives.
    General data structure for all drives and bus systems.
- FB 874 - *VMC_Init*Sigma7W_EC
  – The *Init*t block is used to configure the double-axis drive.
  – Specific block for *Sigma-7W* EtherCAT.
  – The configuration data for the initialization must be stored in the *axis DB*.
- FB 872 - *VMC_Kernel*Sigma7_EC
  – The *Kernel* block communicates with the drive via the appropriate bus system, processes the user requests and returns status messages.
  – The FB 872 - *VMC_Kernel*Sigma7_EC must be called for each axis.
  – Specific block for *Sigma-7* EtherCAT.
  – The exchange of the data takes place by means of the *axis DB*.
- FB 860 - *VMC_AxisControl*
  – General block for all drives and bus systems.
  – The FB 860 - *VMC_AxisControl* must be called for each axis.
  – Supports simple motion commands and returns all relevant status messages.
  – The exchange of the data takes place by means of the *axis DB*.
  – For motion control and status query, via the instance data of the block you can link a visualization.
  – In addition to the FB 860 - *VMC_AxisControl*, *PLCopen* blocks can be used.
- FB 800 ... FB 838 - PLCopen
  – The PLCopen blocks are used to program motion sequences and status queries.
  – The PLCopen blocks must be called for each axis.

5.4.3.2 Programming

Include library

1. Go to the service area of www.vipa.com.
2. Download the *Simple Motion Control* library from the download area at *'VIPA Lib'*.
3. Open the dialog window for ZIP file selection via *'File ➔ Retrieve'*.
4. Select the according ZIP file and click at [Open].
5. Specify a target directory in which the blocks are to be stored and start the unzip process with [OK].

Copy blocks into project

Open the library after unzipping and drag and drop the following blocks into *'Blocks'* of your project:

- *Sigma-7W* EtherCAT:
  - UDT 872 - VMC_ConfigSigma7EC_REF
  - FB 872 - VMC_KernelSigma7_EC
  - FB 874 - VMC_InitSigma7W_EC
- Axis Control
  - UDT 860 - MC_AXIS_REF
  - Blocks for your movement sequences

Create interrupt OBs

1. In your project, click at *'Blocks'* and choose *'Context menu ➔ Insert new object ➔ Organization block'*.

   ⇨ The dialog *'Properties Organization block'* opens.
2. Add OB 57, OB 82, and OB 86 successively to your project.

Create axis DB for *'Module 1'*

1. In your project, click at *'Blocks'* and choose *'Context menu ➔ Insert new object ➔ Data block'*.

   Specify the following parameters:

   - Name and type
     - The DB no. as *'Name'* can freely be chosen, such as DB 10.
     - Set *'Shared DB'* as the *'Type'*.
   - Symbolic name
     - Specify "Axis01".

   Confirm your input with [OK].

   ⇨ The block is created.
2. Open DB 10 "Axis01" by double-click.
   - In "Axis01", create the variable "Config" of type UDT 872. These are specific axis configuration data.
   - In "Axis01", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

   DB10

Address	Name	Type	...
		Struct	
...	Config	"VMC_ConfigSigma7EC_REF"	
...	Axis	"MC_AXIS_REF	
...		END_STRUCT	

Create axis DB for *'Module 2'*

1. Add another DB as your *axis DB* to your project and assign it the name "Axis02". The DB number can freely be selected such as DB11.

   ⇨ The block is created.

2. Open DB 11 "Axis02" by double-click.

   - In "Axis02", create the variable "Config" of type UDT 872. These are specific axis configuration data.
   - In "Axis02", create the variable "Axis" of type UDT 860. During operation, all operating data of the axis are stored here.

   DB 11

Address	Name	Type	...
		Struct	
...	Config	"VMC_ConfigSigma7EC_REF"	
...	Axis	"MC_AXIS_REF	
...		END_STRUCT	

OB 1

Configuration of the double-axis

Open OB 1 and program the following FB calls with associated DBs:

FB 874 - VMC_InitSigma7W_EC, DB 874 *Chapter 5.5.3 'FB 874 - VMC_Init-Sigma7W_EC - Sigma-7W EtherCAT Initialization' on page 120*

At *M1/M2_PdoInputs* respectively *M1/M2_PdoOutputs*, enter the address from the *SPEED7 EtherCAT Manager* for the according axis. *112*

```
⇨ CALL  "VMC_InitSigma7W_EC" , "DI_InitSgm7WETC01"
   Enable                    :=TRUE
   LogicalAddress            :=0
   M1_PdoInputs              :=0 (EtherCAT-Manager
                                  Module1: S7 Input address)

   M1_PdoOutputs             :=0 (EtherCAT-Manager
                                  Module1: S7 Output address)
   M1_EncoderType            :=2
   M1_EncoderResolutionBits  :=20
   M1_FactorPosition         :=1.048576e+006
   M1_FactorVelocity         :=1.048576e+006
   M1_FactorAcceleration     :=1.048576e+002
   M1_OffsetPosition         :=0.000000e+000
   M1_MaxVelocityApp         :=5.000000e+001
   M1_MaxAccelerationApp     :=1.000000e+002
   M1_MaxDecelerationApp     :=1.000000e+002
   M1_MaxVelocityDrive       :=6.000000e+001
   M1_MaxAccelerationDrive   :=1.500000e+002
   M1_MaxDecelerationDrive   :=1.500000e+002
   M1_MaxPosition            :=1.048500e+003
   M1_MinPosition            :=-1.048514e+003
   M1_EnableMaxPosition      :=TRUE
   M1_EnableMinPosition      :=TRUE
   M2_PdoInputs              :=36 (EtherCAT-Manager
                                  Module2: S7 Input address)

   M2_PdoOutputs             :=36 (EtherCAT-Manager
                                  Module2: S7 Output address)
   M2_EncoderType            :=2
   M2_EncoderResolutionBits  :=20
   M2_FactorPosition         :=1.048576e+006
   M2_FactorVelocity         :=1.048576e+006
   M2_FactorAcceleration     :=1.048576e+002
   M2_OffsetPosition         :=0.000000e+000
   M2_MaxVelocityApp         :=5.000000e+001
   M2_MaxAccelerationApp     :=1.000000e+002
   M2_MaxDecelerationApp     :=1.000000e+002
   M2_MaxVelocityDrive       :=6.000000e+001
   M2_MaxAccelerationDrive   :=1.500000e+002
   M2_MaxDecelerationDrive   :=1.500000e+002
   M2_MaxPosition            :=1.048500e+003
   M2_MinPosition            :=-1.048514e+003
   M2_EnableMaxPosition      :=TRUE
   M2_EnableMinPosition      :=TRUE
   M1_MinUserPosition        :=-1000.0
   M1_MaxUserPosition        :=1000.0
   M2_MinUserPosition        :=-1000.0
   M2_MaxUserPosition        :=1000.0
   Valid                     :="InitS7WEC1_Valid"
   Error                     :="InitS7WEC1_Error"
```

```
ErrorID                 :="InitS7WEC1_ErrorID"
M1_Config               :="Axis01".Config
M1_Axis                 :="Axis01".Axis
M2_Config               :="Axis02".Config
M2_Axis                 :="Axis02".Axis
```

Connecting the kernel for the respective axis

The *Kernel* processes the user commands and passes them appropriately processed on to the drive via the respective bus system.

FB 872 - VMC_KernelSigma7_EC, DB 872 for axis 1

FB 872 - VMC_KernelSigma7_EC, DB 1872 for axis 2 *Chapter 5.5.2 'FB 872 - VMC_KernelSigma7_EC - Sigma-7 EtherCAT Kernel' on page 120*

```
⇨       CALL  "VMC_KernelSigma7_EC" , DB 872
         Init  :="KernelS7WEC1_Init"
         Config:="Axis01".Config
         Axis  :="Axis01".Axis

        CALL  "VMC_KernelSigma7_EC" , DB 1872
         Init  :="KernelS7WEC2_Init"
         Config:="Axis02".Config
         Axis  :="Axis02".Axis
```

Connecting the block for motion sequences

For simplicity, the connection of the FB 860 - VMC_AxisControl is to be shown here. This universal block supports simple motion commands and returns status messages. The inputs and outputs can be individually connected. Please specify the reference to the corresponding axis data at *'Axis'* in the *axis DB*.

FB 860 - VMC_AxisControl, DB 860 *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*

```
⇨       CALL  "VMC_AxisControl" , "DI_AxisControl01"
         SourceInputs     :="AxCtrl1_SourceInputs"
         AxisEnable       :="AxCtrl1_AxisEnable"
         AxisReset        :="AxCtrl1_AxisReset"
         HomeExecute      :="AxCtrl1_HomeExecute"
         HomePosition     :="AxCtrl1_HomePosition"
         StopExecute      :="AxCtrl1_StopExecute"
         MvVelocityExecute:="AxCtrl1_MvVelExecute"
         MvRelativeExecute:="AxCtrl1_MvRelExecute"
         MvAbsoluteExecute:="AxCtrl1_MvAbsExecute"
         PositionDistance :="AxCtrl1_PositionDistance"
         Velocity         :="AxCtrl1_Velocity"
         Acceleration     :="AxCtrl1_Acceleration"
         Deceleration     :="AxCtrl1_Deceleration"
         JogPositive      :="AxCtrl1_JogPositive"
         JogNegative      :="AxCtrl1_JogNegative"
         JogVelocity      :="AxCtrl1_JogVelocity"
         JogAcceleration  :="AxCtrl1_JogAcceleration"
         JogDeceleration  :="AxCtrl1_JogDeceleration"
         AxisReady        :="AxCtrl1_AxisReady"
         AxisEnabled      :="AxCtrl1_AxisEnabled"
         AxisError        :="AxCtrl1_AxisError"
         AxisErrorID      :="AxCtrl1_AxisErrorID"
         DriveWarning     :="AxCtrl1_DriveWarning"
         DriveError       :="AxCtrl1_DriveError"
         DriveErrorID     :="AxCtrl1_DriveErrorID"
         IsHomed          :="AxCtrl1_IsHomed"
         ModeOfOperation  :="AxCtrl1_ModeOfOperation"
         PLCopenState     :="AxCtrl1_PLCopenState"
         ActualPosition   :="AxCtrl1_ActualPosition"
         ActualVelocity   :="AxCtrl1_ActualVelocity"
         CmdDone          :="AxCtrl1_CmdDone"
         CmdBusy          :="AxCtrl1_CmdBusy"
         CmdAborted       :="AxCtrl1_CmdAborted"
         CmdError         :="AxCtrl1_CmdError"
         CmdErrorID       :="AxCtrl1_CmdErrorID"
         DirectionPositive:="AxCtrl1_DirectionPos"
         DirectionNegative:="AxCtrl1_DirectionNeg"
         SWLimitMinActive :="AxCtrl1_SWLimitMinActive"
         SWLimitMaxActive :="AxCtrl1_SWLimitMaxActive"
         HWLimitMinActive :="AxCtrl1_HWLimitMinActive"
         HWLimitMaxActive :="AxCtrl1_HWLimitMaxActive"
         Axis             :="Axis...".Axis
```

At *Axis*, enter "Axis01" for axis 1 and "Axis02" for axis 2.

For complex motion tasks, you can use the PLCopen blocks. Here you must also specify the reference to the corresponding axis data at Axis in the axis DB.

Your project now includes the following blocks:

- OB 1 - Main
- OB 57 - DP Manufacturer Alarm
- OB 82 - I/O_FLT1

- OB 86 - Rack_FLT
- FB 860 - VMC_AxisControl with instance DB
- FB 872 - VMC_KernelSigma7_EC with instance DB
- FB 874 - VMC_InitSigma7W_EC with instance DB
- UDT 860 - MC_Axis_REF
- UDT 872 - VMC_ConfigSigma7EC_REF

Sequence of operations

1. Choose the Siemens SIMATIC Manager and transfer your project into the CPU.

   The transfer can only be done by the Siemens SIMATIC Manager - not hardware configurator!

   Since slave and module parameters are transmitted by means of SDO respectively SDO Init command, the configuration remains active, until a power cycle is performed or new parameters for the same SDO objects are transferred.

   With an overall reset the slave and module parameters are not reset!

   ⇨ You can take your application into operation now.

   CAUTION!
   Please always observe the safety instructions for your drive, especially during commissioning!

2. Before the double-axis drive can be controlled, it must be initialized. To do this, call the *Init* block FB 874 - VMC_InitSigma7W_EC with *Enable* = TRUE.

   ⇨ The output *Valid* returns TRUE. In the event of a fault, you can determine the error by evaluating the *ErrorID*.

   You have to call the *Init* block again if you load a new axis DB or you have changed parameters on the *Init* block.

   Do not continue until the Init block report an error!

3. Ensure that the *Kernel* block FB 872 - VMC_KernelSigma7_EC is called cyclically for each axis. In this way, control signals are transmitted to the drive and status messages are reported.
4. Program your application with the FB 860 - VMC_AxisControl or with the PLCopen blocks for each axis.

5.4.4 Copy project

Proceeding

In the example, the station *'Source'* is copied and saved as *'Target '*.

1. Open the hardware configuration of the *'Source'* CPU and start the *SPEED7 EtherCAT Manager*.
2. In the *SPEED7 EtherCAT Manager*, via *'File* ➔ *Save as'* save the configuration in your working directory.
3. Close the *SPEED7 EtherCAT Manager* and the hardware configurator.

4. Copy the station *'Source'* with Ctrl + C and paste it as *'Target'* into your project with Ctrl + V.
5. Select the *'Blocks'* directory of the *'Target'* CPU and delete the *'System data'*.
6. Open the hardware configuration of the *'Target'* CPU. Adapt the IP address data or re-network the CPU or the CP again.

Before calling the SPEED7 EtherCAT Manager you have always to save your project with 'Station ➔ Save and compile'.

7. Safe your project with *'Station ➔ Safe and compile'*.
8. Open the *SPEED7 EtherCAT Manager*.
9. Use *'File ➔ Open'* to load the configuration from your working directory.
10. Close the *SPEED7 EtherCAT Manager*.
11. Save and compile your configuration.

5.5 Drive specific blocks

5.5.1 UDT 872 - VMC_ConfigSigma7EC_REF - *Sigma-7* EtherCAT Data structure axis configuration

This is a user-defined data structure that contains information about the configuration data. The UDT is specially adapted to the use of a *Sigma-7* drive, which is connected via EtherCAT.

5.5.2 FB 872 - VMC_KernelSigma7_EC - *Sigma-7* EtherCAT Kernel

Description

This block converts the drive commands for a *Sigma-7* axis via EtherCAT and communicates with the drive. For each *Sigma-7* axis, an instance of this FB is to be cyclically called.

Please note that this module calls the SFB 238 internally.

In the SPEED7 Studio, this module is automatically inserted into your project.

In Siemens SIMATIC Manager, you have to copy the SFB 238 from the Motion Control Library into your project.

Parameter	Declaration	Data type	Description
Init	INPUT	BOOL	The block is internally reset with an edge 0-1. Existing motion commands are aborted and the block is initialized.
Config	IN_OUT	UDT872	Data structure for transferring axis-dependent configuration data to the *AxisKernel*.
Axis	IN_OUT	MC_AXIS_REF	Data structure for transferring axis-dependent information to the *AxisKernel* and PLCopen blocks.

5.5.3 FB 874 - VMC_InitSigma7W_EC - *Sigma-7W* EtherCAT Initialization

Description

This block is used to configure the double-axis of a *Sigma-7W* drive. The block is specially adapted to the use of a *Sigma-7W* drive, which is connected via EtherCAT.

Parameter	Declaration	Data type	Description
M1_Config	IN_OUT	UDT872	Data structure for transferring axis-dependent configuration data to the *AxisKernel* for axis 1.
M1_Axis	IN_OUT	MC_AXIS_REF	Data structure for transferring axis-dependent information to the *AxisKernel* and PLCopen blocks for axis 1.
M2_Config	IN_OUT	UDT872	Data structure for transferring axis-dependent configuration data to the *AxisKernel* for axis 2.
M2_Axis	IN_OUT	MC_AXIS_REF	Data structure for transferring axis-dependent information to the *AxisKernel* and PLCopen blocks for axis 2.
Enable	INPUT	BOOL	Release of initialization
LogicalAddress	INPUT	INT	Start address of the PDO input data
M1_PdoInputs	INPUT	INT	Start address of the input PDOs for axis 1

Parameter	Declaration	Data type	Description
M1_PdoOutputs	INPUT	INT	Start address of the output PDOs for axis 1
M1_EncoderType	INPUT	INT	Encoder type of axis 1 ■ 1: Absolute encoder ■ 2: Incremental encoder
M1_EncoderResolutionBits	INPUT	INT	Number of bits corresponding to one encoder revolution of axis 1. Default: 20
M1_FactorPosition	INPUT	REAL	Factor for converting the position of user units [u] into drive units [increments] and back of axis 1. It's valid: $p_{[increments]} = p_{[u]}$ x *FactorPosition* Please consider the factor which can be specified on the drive via the objects 0x2701: 1 and 0x2701: 2. This should be 1.
M1_FactorVelocity	INPUT	REAL	Factor for converting the speed of user units [u/s] into drive units [increments/s] and back of axis 1. It's valid: $v_{[increments/s]} = v_{[u/s]}$ x *FactorVelocity* Please also take into account the factor which you can specify on the drive via objects 0x2702: 1 and 0x2702: 2. This should be 1.
M1_FactorAcceleration	INPUT	REAL	Factor to convert the acceleration of user units [u/s²] in drive units [10^{-4} x increments/s²] and back of axis 1. It's valid: 10^{-4} x $a_{[increments/s^2]} = a_{[u/s^2]}$ x *FactorAcceleration* Please also take into account the factor which you can specify on the drive via objects 0x2703: 1 and 0x2703: 2. This should be 1.
M1_OffsetPosition	INPUT	REAL	Offset for the zero position of axis 1 [u].
M1_MaxVelocityApp	INPUT	REAL	Maximum application speed of axis 1 [u/s]. The command inputs are checked to the maximum value before execution.
M1_MaxAccelerationApp	INPUT	REAL	Maximum acceleration of application of axis 1 [u/s²]. The command inputs are checked to the maximum value before execution.
M1_MaxDecelerationApp	INPUT	REAL	Maximum acceleration of application of axis 1 [u/s²]. The command inputs are checked to the maximum value before execution.
M1_MaxPosition	INPUT	REAL	Maximum position for monitoring the software limits of axis 1 [u].
M1_MinPosition	INPUT	REAL	Minimum position for monitoring the software limits of axis 1 [u].
M1_EnableMaxPosition	INPUT	BOOL	Monitoring maximum position of axis 1 ■ TRUE: Activates the monitoring of the maximum position.
M1_EnableMinPosition	INPUT	BOOL	Monitoring minimum position of axis 1 ■ TRUE: Activation of the monitoring of the minimum position.

Parameter	Declaration	Data type	Description
M2_PdoInputs	INPUT	INT	Start address of the input PDOs for axis 2
M2_PdoOutputs	INPUT	INT	Start address of the output PDOs for axis 2
M2_EncoderType	INPUT	INT	Encoder type of axis 2 ■ 1: Absolute encoder ■ 2: Incremental encoder
M2_EncoderResolutionBits	INPUT	INT	Number of bits corresponding to one encoder revolution of axis 2. Default: 20
M2_FactorPosition	INPUT	REAL	Factor for converting the position of user units [u] into drive units [increments] and back of axis 2. It's valid: $p_{[increments]} = p_{[u]}$ x *FactorPosition* Please consider the factor which can be specified on the drive via the objects 0x2701: 1 and 0x2701: 2. This should be 1.
M2_FactorVelocity	INPUT	REAL	Factor for converting the speed of user units [u/s] into drive units [increments/s] and back of axis 2. It's valid: $v_{[increments/s]} = v_{[u/s]}$ x *FactorVelocity* Please also take into account the factor which you can specify on the drive via objects 0x2702: 1 and 0x2702: 2. This should be 1.
M2_FactorAcceleration	INPUT	REAL	Factor to convert the acceleration of user units [u/s^2] in drive units [10^{-4} x increments/s^2] and back of axis 2. It's valid: 10^{-4} x $a_{[increments/s^2]} = a_{[u/s^2]}$ x *FactorAcceleration* Please also take into account the factor which you can specify on the drive via objects 0x2703: 1 and 0x2703: 2. This should be 1.
M2_OffsetPosition	INPUT	REAL	Offset for the zero position of axis 2 [u].
M2_MaxVelocityApp	INPUT	REAL	Maximum application speed of axis 2 [u/s]. The command inputs are checked to the maximum value before execution.
M2_MaxAccelerationApp	INPUT	REAL	Maximum acceleration of application of axis 2 [u/s^2]. The command inputs are checked to the maximum value before execution.
M2_MaxDecelerationApp	INPUT	REAL	Maximum acceleration of application of axis 2 [u/s^2]. The command inputs are checked to the maximum value before execution.
M2_MaxPosition	INPUT	REAL	Maximum position for monitoring the software limits of axis 2 [u].
M2_MinPosition	INPUT	REAL	Minimum position for monitoring the software limits of axis 2 [u].
M2_EnableMaxPosition	INPUT	BOOL	Monitoring maximum position of axis 2 ■ TRUE: Activates the monitoring of the maximum position.

Parameter	Declaration	Data type	Description
M2_EnableMinPosition	INPUT	BOOL	Monitoring minimum position of axis 2 ■ TRUE: Activation of the monitoring of the minimum position.
M1_MinUserPosition	OUTPUT	REAL	Minimum user position for axis 1 based on the minimum encoder value of 0x80000000 and the *FactorPosition* [u].
M1_MaxUserPosition	OUTPUT	REAL	Maximum user position for axis 1 based on the maximum encoder value of 0x7FFFFFFF and the *FactorPosition* [u].
M2_MinUserPosition	OUTPUT	REAL	Minimum user position for axis 2 based on the minimum encoder value of 0x80000000 and the *FactorPosition* [u].
M2_MaxUserPosition	OUTPUT	REAL	Maximum user position for axis 2 based on the maximum encoder value of 0x7FFFFFFF and the *FactorPosition* [u].
Valid	OUTPUT	BOOL	Initialization ■ TRUE: Initialization is valid.
Error	OUTPUT	BOOL	■ Error – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*. The axis is disabled.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

6 Blocks for axis control

6.1 Overview

At *Axis Control* you can find the blocks for programming motion tasks and status queries.

Simple motion tasks

Block	
UDT 860 - MC_AXIS_REF - Data structure for axis	*126*
FB 860 - VMC_AxisControl - Control of drive functions and query of drive states	*126*

Complex motion tasks - PLCopen blocks

Block	See page
UDT 860 - MC_AXIS_REF - Data structure for axis	*130*
UDT 861 - MC_TRIGGER_REF - Data structure	*130*
FB 800 - MC_Power Enable respectively disable axis	*131*
FB 801 - MC_Home Homing axis	*133*
FB 802 - MC_Stop Stop axis	*135*
FB 803 - MC_Halt Stop axis	*137*
FB 804 - MC_MoveRelative Move axis relative	*139*
FB 805 - MC_MoveVelocity Drive axis with constant velocity	*141*
FB 808 - MC_MoveAbsolute Move axis to absolute position	*143*
FB 811 - MC_Reset Reset axis	*145*
FB 812 - MC_ReadStatus Read PLCopen-State of the axis	*147*
FB 813 - MC_ReadAxisError Read axis error	*149*
FB 814 - MC_ReadParameter Read axis parameter data	*151*
FB 815 - MC_WriteParameter Write parameter to axis	*153*

6.2 Simple motion tasks

6.2.1 UDT 860 - MC_AXIS_REF - Data structure axis data

This is a user-defined data structure that contains status information of the axis.

6.2.2 FB 860 VMC_AxisControl - Control block axis control

Description

With the FB *VMC_AxisControl* you can control the connected axis. You can check the status of the drive, turn the drive on or off, or execute various motion commands. A separate memory area is located in the instance data of the block. You can control your axis by means of an HMI. To do this, you must set the *SourceInputs* parameter to TRUE.

The VMC_AxisControl block should never be used simultaneously with the PLCopen module MC_Power. Since the VMC_AxisControl contains functionalities of the MC_Power and the latest command from the VMC_Kernel module is always executed, this can lead to a faulty behavior of the drive.

Parameter

CAUTION!
Parameter *SourceInputs*
Please note that switching via *SourceInputs* is only possible if the axis is in the *Disabled* state. Otherwise, this can lead to damage to man and machine!

Parameter	Declaration	Data type	Description
SourceInputs	INPUT	BOOL	■ Input for the module – FALSE: Block works with the block inputs. – TRUE: Block works with the optional variables, which are located in the static area of the instance data block (input HMI), which are used for control, e.g. from an HMI. So it is possible to switch between "automatic mode" and "manual mode".
AxisEnable	INPUT	BOOL	■ Enable/disable axis – TRUE: The axis is enabled. – FALSE: The axis is disabled.
AxisReset	INPUT	BOOL	■ Reset axis – Edge 0-1: Axis reset is performed.
HomeExecute	INPUT	BOOL	■ Homing – Edge 0-1: Homing is started.
HomePosition	INPUT	REAL	With a successful homing the current position of the axis is uniquely set to Position. Position is to be entered in the used application unit.
StopExecute	INPUT	BOOL	■ Stop axis – Edge 0-1: Stopping of the axis is started.

Parameter	Declaration	Data type	Description
MvVelocityExecute	INPUT	BOOL	■ Start moving the axis – Edge 0-1: The axis is accelerated / decelerated to the speed specified.
MvRelativeExecute	INPUT	BOOL	■ Start moving the axis – Edge 0-1: The relative positioning of the axis is started.
MvAbsoluteExecute	INPUT	BOOL	■ Start moving the axis – Edge 0-1: The absolute positioning of the axis is started.
Direction *	INPUT	BYTE	Mode for absolute positioning: ■ 0: shortest distance ■ 1: positive direction ■ 2: negative direction ■ 3: current direction
PositionDistance	INPUT	REAL	Absolute position or relative distance depending on the command in [user units].
Velocity	INPUT	REAL	Velocity setting (signed value) in [user units / s].
Acceleration	INPUT	REAL	Acceleration in [user units / s^2].
Deceleration	INPUT	REAL	Deceleration in [user units / s^2].
JogPositive	INPUT	BOOL	■ Drive axis with constant velocity in positive direction – Edge 0-1: Drive axis with constant velocity is started. – Edge 1-0: The axis is stopped.
JogNegative	INPUT	BOOL	■ Drive axis with constant velocity in negative direction – Edge 0-1: Drive axis with constant velocity is started. – Edge 1-0: The axis is stopped.
JogVelocity	INPUT	REAL	Speed setting for jogging (positive value) in [user units / s].
JogAcceleration	INPUT	REAL	Acceleration in [user units / s^2].
JogDeceleration	INPUT	REAL	Delay for jogging in [user units / s^2].
AxisReady	OUTPUT	BOOL	■ AxisReady – TRUE: The axis is ready to switch on. – FALSE: The axis is not ready to switch on. → Check and fix AxisError (see *AxisErrorID*). → Check and fix DriveError (see *DriveErrorID*). → Check initialization FB (input and output addresses or PDO mapping correct?)
AxisEnabled	OUTPUT	BOOL	■ Status axis – TRUE: Axis is switched on and accepts motion commands. – FALSE: Axis is not switched on and does not accepts motion commands.
AxisError	OUTPUT	BOOL	■ Motion axis error – TRUE: An error has occurred. Additional error information can be found in the parameter *AxisErrorID*. → The axis is disabled.

Parameter	Declaration	Data type	Description
AxisErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
DriveWarning	OUTPUT	BOOL	■ Warning – TRUE: There is a warning on the drive. Additional information can be found in the manufacturer's manual.
DriveError	OUTPUT	BOOL	■ Error on the drive – TRUE: An error has occurred. Additional error information can be found in the parameter *DriveErrorID*. → The axis is disabled.
DriveErrorID	OUTPUT	WORD	■ Error – TRUE: There is an error on the drive. Additional information can be found in the manufacturer's manual.
IsHomed	OUTPUT	BOOL	■ Information axis: homed – TRUE: The axis is homed.
ModeOfOperation	OUTPUT	INT	Drive-specific mode. For further information see drive manual. Example *Sigma-5*: 0: No mode changed/no mode assigned 1: Profile Position mode 2: Reserved (keep last mode) 3: Profile Velocity mode 4: Torque Profile mode 6: Homing mode 7: Interpolated Position mode 8: Cyclic Sync Position mode 9: Cyclic Sync Velocity mode 10: Cyclic Sync Torque mode Other Reserved (keep last mode)
PLCopenState	OUTPUT	INT	Current PLCopenState: 1: Disabled 2: Standstill 3: Homing 4: Discrete Motion 5: Continous Motion 6: Synchronised Motion 7: Stopping 8: Errorstop
ActualPosition	OUTPUT	REAL	Position of the axis in [user unit].

Parameter	Declaration	Data type	Description
ActualVelocity	OUTPUT	REAL	Velocity of the axis in [user unit / s]
CmdDone	OUTPUT	BOOL	■ Status – TRUE: Job ended without error.
CmdBusy	OUTPUT	BOOL	■ Status – TRUE: Job is running.
CmdAborted	OUTPUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job.
CmdError	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *CmdErrorID*.
CmdErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
DirectionPositive	OUTPUT	BOOL	■ Status motion job: Position increasing – TRUE: The position of the axis is increasing
DirectionNegative	OUTPUT	BOOL	■ Status motion job: Position decreasing – TRUE: The position of the axis is decreasing
SWLimitMinActive	OUTPUT	BOOL	■ Software limit switch – TRUE: Software Limit switch Minimum active (Minimum position in negative direction exceeded).
SWLimitMaxActive	OUTPUT	BOOL	■ Software limit switch – TRUE: Software limit switch Maximum active (Maximum position in positive direction exceeded).
HWLimitMinActive	OUTPUT	BOOL	■ Hardware limit switch – TRUE: Negative hardware limit switch active on the drive (NOT- Negative Overtravel).
HWLimitMaxActive	OUTPUT	BOOL	■ Hardware limit switch – TRUE: Positive hardware limit switch active on the drive (POT- Positive Overtravel).
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis.

*) This parameter is not supported by all drives, e.g. *Sigma 5 via EtherCAT* does not support this parameter.

6.3 Complex motion tasks - PLCopen blocks

6.3.1 UDT 860 - MC_AXIS_REF - Data structure axis data

This is a user-defined data structure that contains status information of the axis.

6.3.2 UDT 861 - MC_TRIGGER_REF - Data structure trigger signal

This is a user defined data structure, that contains information of the trigger signal.

6.3.3 FB 800 - MC_Power - enable/disable axis

Description With MC_Power an axis can be enabled or disabled.

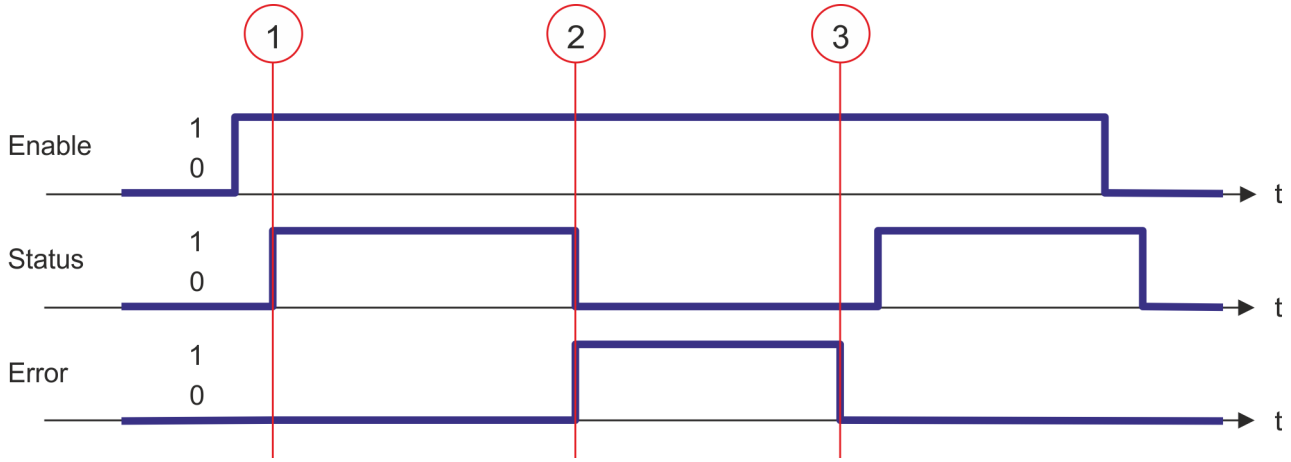
Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Enable	INPUT	BOOL	■ Enable/disable axis – TRUE: The axis is enabled – FALSE: The axis is disabled
EnablePositive	INPUT	BOOL	Parameter is currently not supported; call with FALSE
EnableNegative	INPUT	BOOL	Parameter is currently not supported; call with FALSE
Status	OUTPUT	BOOL	■ Status axis – TRUE: The axis is ready to execute motion control jobs – FALSE: The axis is not ready to execute motion control jobs
Valid	OUTPUT	BOOL	Always FALSE
Error	OUTPUT	BOOL	■ Error – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*. The axis is disabled.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

Enable axis Call MC_Power with *Enable* = TRUE. If *Status* shows a value of TRUE, the axis is enabled. In this status motion control jobs can be activated.

Disable axis Call MC_Power with *Enable* = FALSE. If *Status* shows a value of FALSE, the axis is disabled. When disabling the axis a possibly active motion job is cancelled and the axis is stopped.

Status diagram of the block parameters



(1) The axis is enabled with *Enable* = TRUE. At the time (1) it is enabled. Then motion control jobs can be activated.
(2) At the time (2) an error occurs, which causes the to disable the axis. A possibly active motion job is cancelled and the axis is stopped.
(3) The error is eliminated and acknowledged at time (3). Thus *Enable* is further set, the axis is enabled again. Finally the axis is disabled with *Enable* = FALSE.

6.3.4 FB 801 - MC_Home - home axis

Description

With MC_Home an axis can be set to a reference point. This is used to match the axis coordinates to the real, physical drive position. The homing method and its parameters must be configured directly at the drive. For this use the VMC_HomeInit_... blocks. With a virtual axis there is no configuration possible. Here, the actual position of the axis is set to input parameter *Position*.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Homing – Edge 0-1: Homing is started
Position	INPUT	REAL	With a successful homing the current position of the axis is uniquely set to *Position*. *Position* is to be entered in the used application unit.
BufferMode	INPUT	BYTE	Parameter is currently not supported; call with B#16#0
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done.
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running.
CommandAborted	OUTPUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job.
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

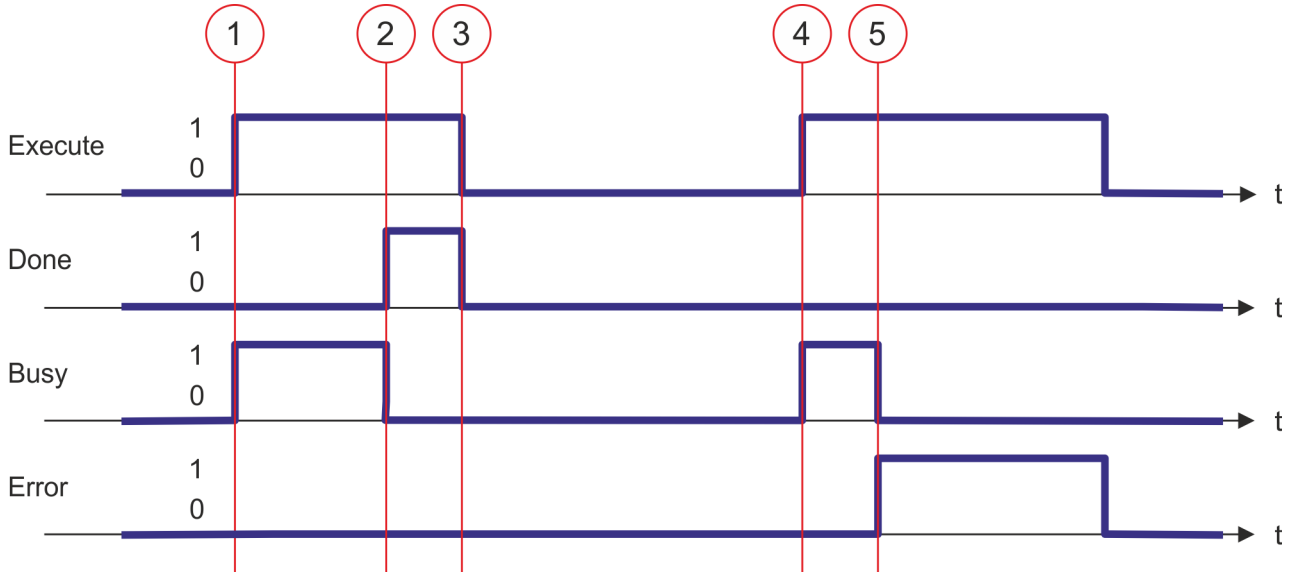
Start of the job only in the PLCopen-State *Standstill* possible.

Home axis

The homing is started with edge 0-1 at *Execute*. *Busy* is TRUE as soon as the homing is running. Once *Done* becomes TRUE, homing was successfully completed. The current position of the axis was set to the value of *Position*.

– *An active job continues to run even when Execute is set to FALSE.*
– *A running job can not be aborted by a move job (e.g. MC_MoveRelative).*

Status diagram of the block parameters



(1) The homing is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) the homing is completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.
(4) At the time (4) with an edge 0-1 at *Execute* the homing is started again and *Busy* becomes TRUE.
(5) At the time (5) an error occurs during homing. *Busy* has the value FALSE and *ERROR* den value TRUE.

6.3.5 FB 802 - MC_Stop - stop axis

Description

With MC_STOP the axis is stopped. With the parameter *Deceleration*, the dynamic behavior can be determined during stopping.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Stop axis – Edge 0-1: Stopping of the axis is started
Deceleration	INPUT	REAL	Delay in stopping in [user units/s^2]
Jerk	INPUT	REAL	Parameter is currently not supported; call with 0.0
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
CommandA-borted	OUTPUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job.
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

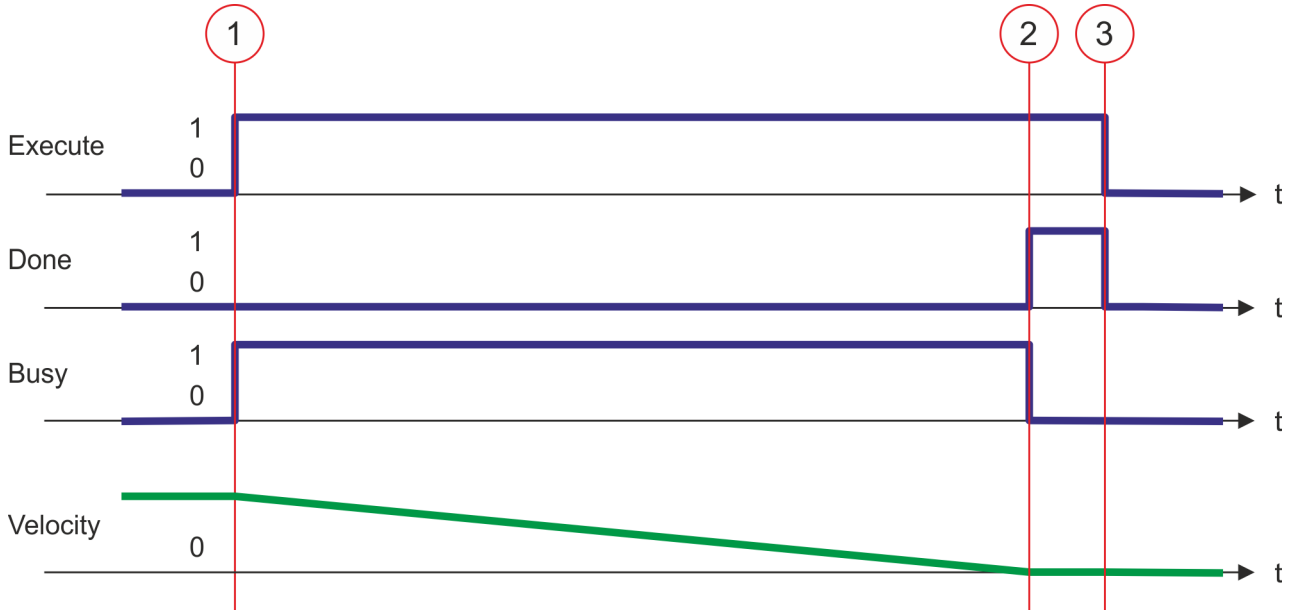
- Start of the job in the PLCopen-States *Standstill*, *Homing*, *Discrete Motion*, *Synchronized Motion* and *Continuous Motion* possible.
- MC_Stop switches the axis to the PLCopen-State *Stopping*. In *Stopping* no motion jobs can be started. As long as *Execute* is true, the axis remains in PLCopen-State *Stopping*. If *Execute* becomes FALSE, the axis switches to PLCopen-State*Standstill*. In *Standstill* motion tasks can be started.

Stop axis

The stopping of the axis is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as the stopping of the axis is running. After the axis has been stopped and thus the speed has reached 0, *Busy* with FALSE and *Done* with TRUE is returned.

– *An active job continues until the axis stops even when Execute is set to FALSE.*
– *A running job can not be aborted by a move job (e.g. MC_MoveRelative).*

Status diagram of the block parameters



(1) Stopping of the axis is started with edge 0-1 at *Execute* and *Busy* becomes TRUE. The velocity of the axis is reduced to zero, regarding the parameter *Deceleration*.
(2) At time (2) stopping the axis is completed, the axis is stopped. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.6 FB 803 - MC_Halt - holding axis

Description

With MC_Halt the axis is slowed down to standstill. With the parameter *Deceleration* the dynamic behavior can be determined during breaking.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Stop axis – Edge 0-1: Stopping of the axis is started
Deceleration	INPUT	REAL	Delay in breaking in [user units/s^2]
Jerk	INPUT	REAL	Parameter is currently not supported; call with 0.0
BufferMode	INPUT	BYTE	Parameter is currently not supported; call with B#16#0
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Active	OUTPUT	BOOL	■ Status – TRUE: Block controls the axis
CommandA-borted	OUTPUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

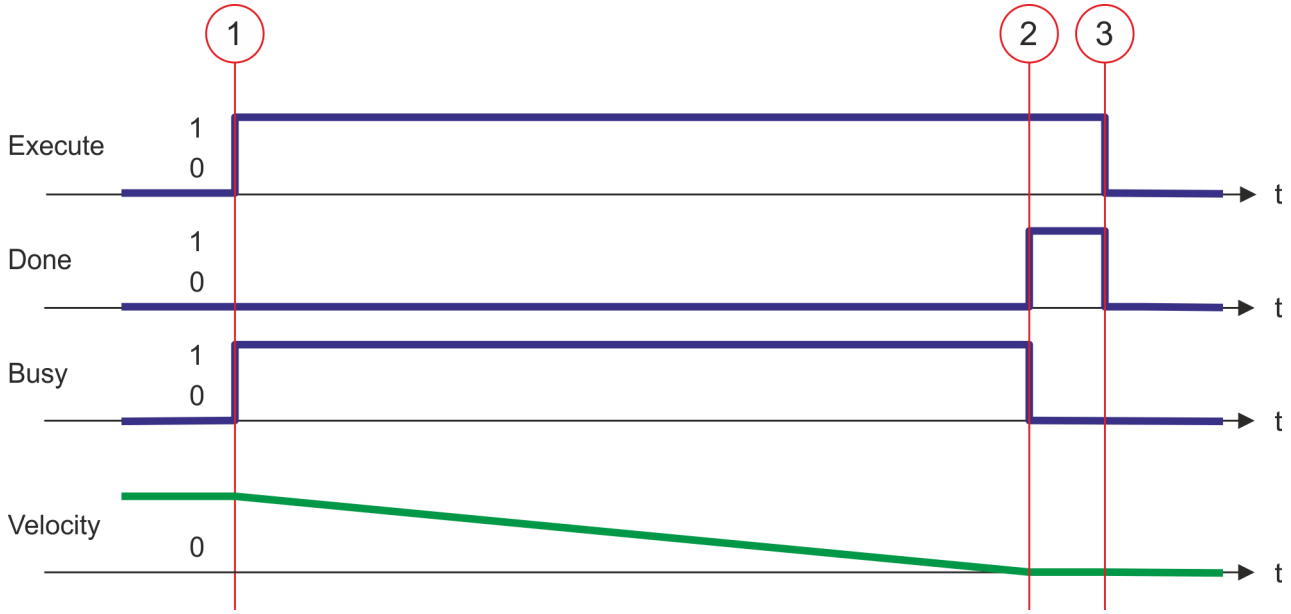
- Start of the job in the PLCopen-States *Discrete Motion*, *Synchronized Motion* and *Continuous Motion* possible.
- MC_Halt switches the axis to the PLCopen-State *Discrete Motion*.

Slow down axis

The slow down of the axis is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as the slow down of the axis is running. After the axis has been slowed down and thus the speed has reached 0, *Busy* with FALSE and *Done* with TRUE is returned.

– *An active job continues until the axis stops even when Execute is set to FALSE.*
– *A running job can be aborted by a move job (e.g. MC_MoveRelative).*

Status diagram of the block parameters



(1) Breaking the axis is started with edge 0-1 at *Execute* and *Busy* becomes TRUE. The velocity of the axis is reduced to zero, regarding the parameter *Deceleration*.
(2) At time (2) slowing down the axis is completed, the axis is stopped. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.7 FB 804 - MC_MoveRelative - move axis relative

Description

With MC_MoveRelative the axis is moved relative to the position in order to start a specified distance. With the parameters *Velocity*, *Acceleration* and *Deceleration* the dynamic behavior can be determined during the movement.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Move axis relative – Edge 0-1: The relative movement of the axis is started
ContinuousUpdate	INPUT	BOOL	Parameter is currently not supported; call with FALSE
Distance	INPUT	REAL	Relative distance in [user units]
Velocity	INPUT	REAL	Max. Velocity (needs not necessarily be reached) in [user units/s]
Acceleration	INPUT	REAL	Acceleration in [user units/s^2]
Deceleration	INPUT	REAL	Delay in breaking in [user units/s^2]
Jerk	INPUT	REAL	Parameter is currently not supported; call with 0.0
BufferMode	INPUT	BYTE	Parameter is currently not supported; call with B#16#0
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done; target position reached
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Active	OUTPUT	BOOL	■ Status – TRUE: Block controls the axis
CommandAborted	OUTPUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

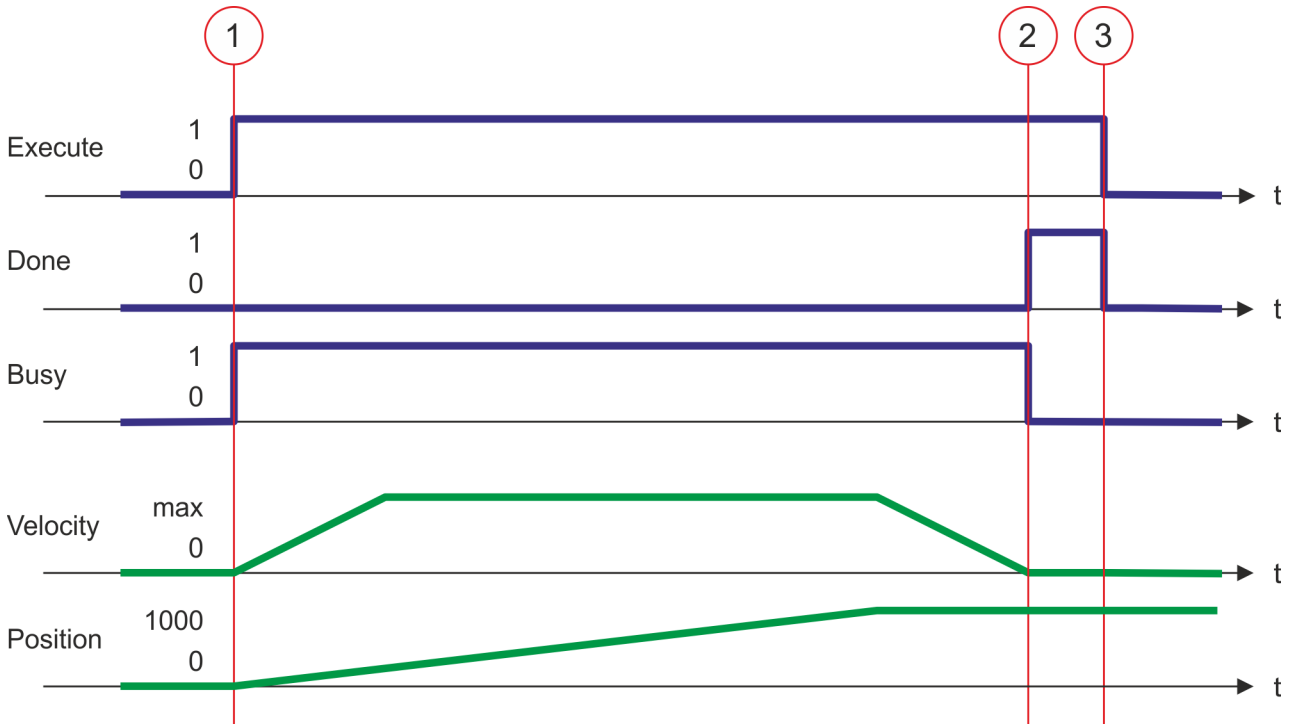
- Start of the job in the PLCopen-States *Standstill*, *Discrete Motion*, *Synchronized Motion* and *Continuous Motion* possible.
- MC_MoveRelative switches the axis to the PLCopen-State *Discrete Motion*.

Move axis relative

The movement of the axis is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as the movement of the axis is running. After the target position was reached, *Busy* with FALSE and *Done* with TRUE is returned. Then the velocity of the axis is 0.

- *An active job continues to move to target position even when Execute is set to FALSE.*
- *A running job can be aborted by a move job (e.g. MC_MoveAbsolute).*

Status diagram of the block parameters



(1) With MC_MoveRelative the axis is moved relative by a *Distance* = 1000.0 (start position at job start is 0.0). Moving the axis is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At time (2) the axis was moved by the *Distance* = 1000.0, i.e. the target position was reached. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.8 FB 805 - MC_MoveVelocity - drive axis with constant velocity

Description

With MC_MoveVelocity the axis is driven with a constant velocity. With the parameters *Velocity*, *Acceleration* and *Deceleration* the dynamic behavior can be determined during the movement.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Drive axis with constant velocity – Edge 0-1: Drive axis with constant velocity is started
ContinuousUpdate	INPUT	BOOL	Parameter is currently not supported; call with FALSE
Velocity	INPUT	REAL	Velocity setting (signed value) in [user units/s]
Acceleration	INPUT	REAL	Acceleration in [user units/s^2]
Deceleration	INPUT	REAL	Delay in breaking in [user units/s^2]
Jerk	INPUT	REAL	Parameter is currently not supported; call with 0.0
BufferMode	INPUT	BYTE	Parameter is currently not supported; call with B#16#0
InVelocity	OUTPUT	BOOL	■ Velocity setting – TRUE: Velocity setting reached
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Active	OUTPUT	BOOL	■ Status – TRUE: Block controls the axis
CommandAborted	OUTPUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

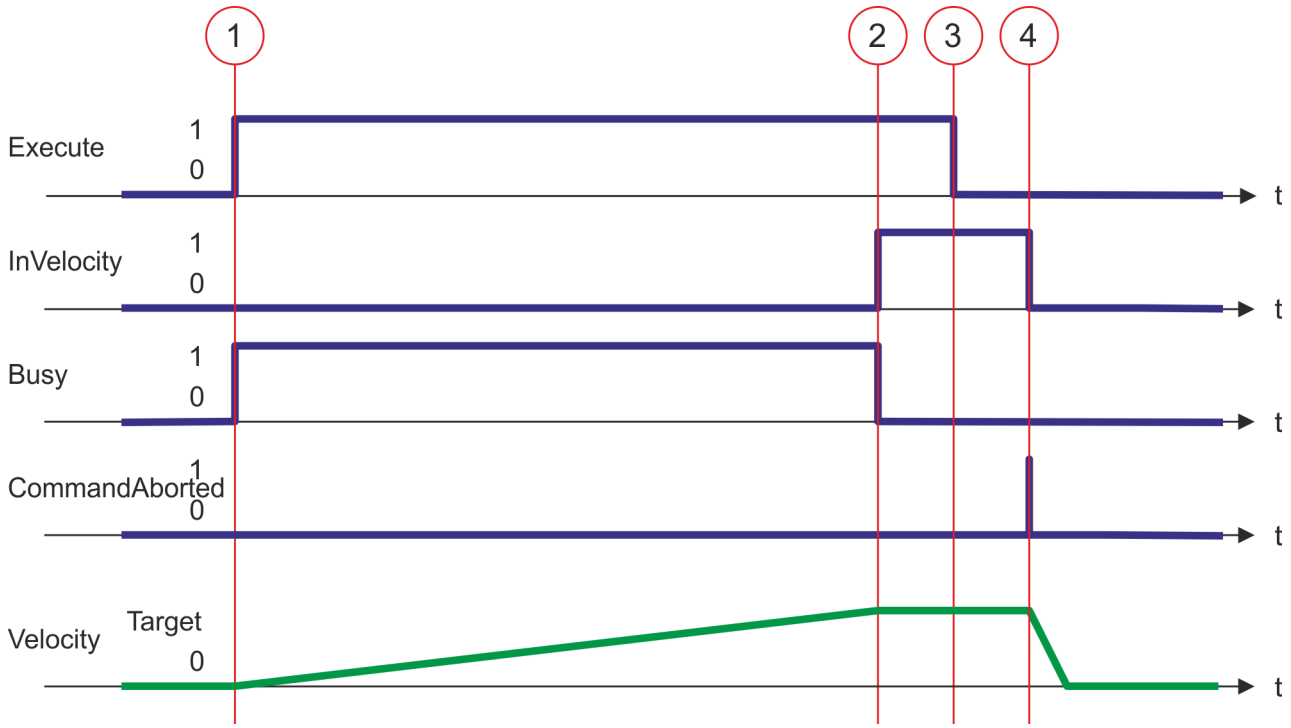
- Start of the job in the PLCopen-States *Standstill*, *Discrete Motion*, *Synchronized Motion* and *Continuous Motion* possible.
- MC_MoveVelocity switches the axis to the PLCopen-State *Continuous Motion*.

Drive axis with set velocity

The movement of the axis with set velocity is started with an edge 0-1 at *Execute*. *Busy* is TRUE and *InVelocity* FALSE as soon as the set velocity is not reached. If the set velocity is reached, *Busy* becomes FALSE and *InVelocity* TRUE. The axis is constant moved with this velocity.

– *An active job is continued, even when the set velocity is reached and even when Execute is set to FALSE.*
– *A running job can be aborted by a move job (e.g. MC_MoveAbsolute).*

Status diagram of the block parameters



(1) Moving the axis with set velocity is started with edge 0-1 at Execute and *Busy* becomes TRUE.
(2) At time (2) the axis reaches the set velocity and *Busy* has the value FALSE and *InVelocity* the value TRUE.
(3) Resetting Execute to FALSE at time (3) does not influence the axis. The axis is further moved with constant set velocity and *InVelocity* is further TRUE.
(4) At the time (4) the MC_Velocity job is aborted by a MC_Halt job. The axis is decelerated to stop.

6.3.9 FB 808 - MC_MoveAbsolute - move axis to absolute position

Description

With MC_MoveAbsolute the axis is moved to an absolute position. With the parameters *Velocity*, *Acceleration* and *Deceleration* the dynamic behavior can be determined during the movement.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Move the axis – Edge 0-1: The movement of the axis is started
ContinuousUpdate	INPUT	BOOL	Parameter is currently not supported; call with FALSE
Position	INPUT	REAL	Absolute position in [user units]
Velocity	INPUT	REAL	Maximum velocity (needs not necessarily be reached) signed value in [user units/s]
Acceleration	INPUT	REAL	Acceleration in [user units/s^2]
Deceleration	INPUT	REAL	Delay in breaking in [user units/s^2]
Jerk	INPUT	REAL	Parameter is currently not supported; call with 0.0
Direction	INPUT	Byte	■ Direction – 0: Shortest way – 1: Positive direction – 2: Negative direction – 3: Current direction
BufferMode	INPUT	BYTE	Parameter is currently not supported; call with B#16#0
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Target position was reached.
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Active	OUTPUT	BOOL	■ Status – TRUE: Block controls the axis
CommandAborted	OUTPUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

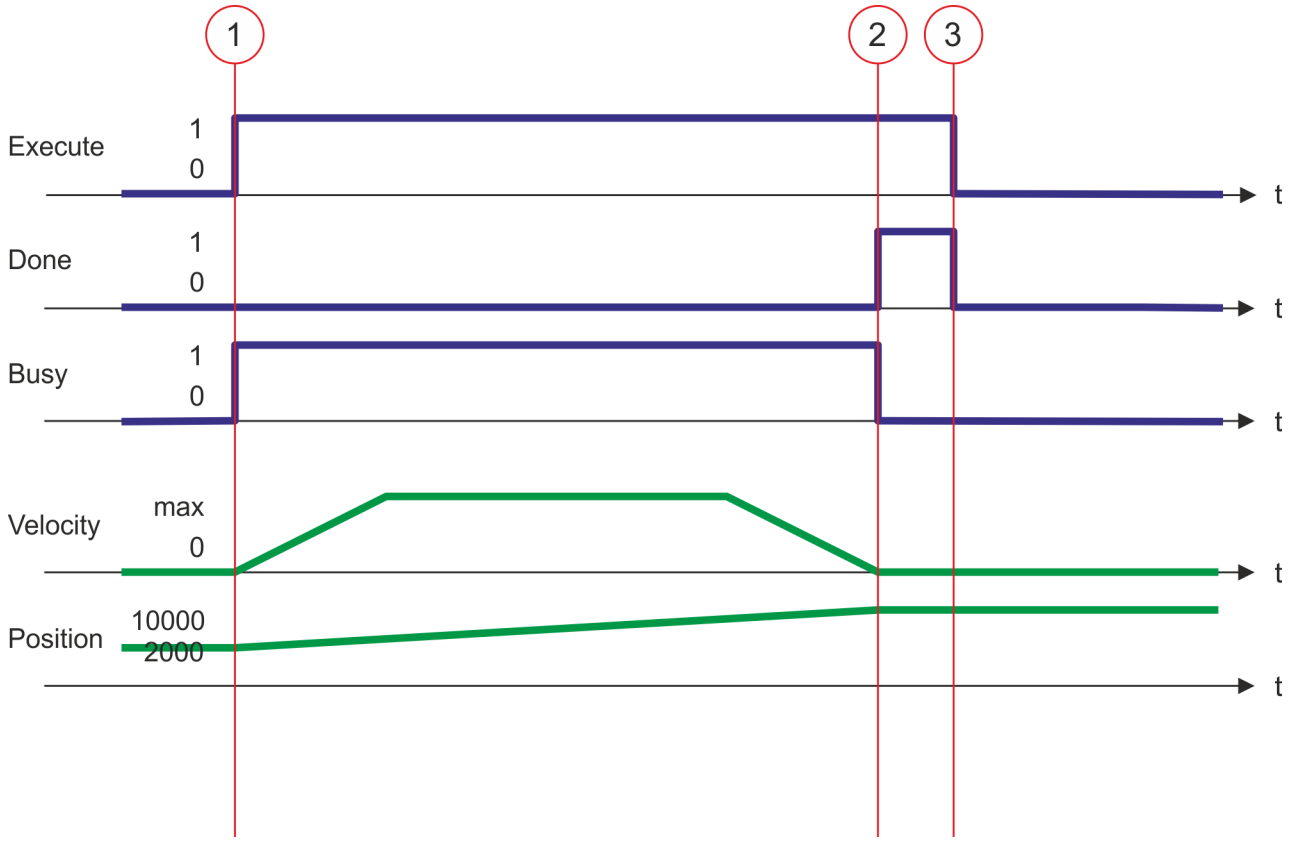
- Start of the job in the PLCopen-States *Standstill*, *Discrete Motion*, *Synchronized Motion* and *Continuous Motion* possible.
- MC_MoveVelocity switches the axis to the PLCopen-State *Discrete Motion*.

Move axis absolute

The movement of the axis is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as the movement of the axis is running. After the target position was reached, *Busy* with FALSE and *Done* with TRUE is returned. Then the velocity of the axis is 0.

- *With Sigma-5 EtherCAT the target position is always reached via the shortest way.*
- *An active job continues to move to target position even when Execute is set to FALSE.*
- *A running job can be aborted by a move job (e.g. MC_MoveVelocity).*

Status diagram of the block parameters



(1) With MC_MoveAbsolute the axis is moved to the absolute position = 10000.0 (start position at job start is 2000.0). At time (1) moving the axis is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At time (2) the axis has reached the target position. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.10 FB 811 - MC_Reset - reset axis

Description

With MC_Reset a reset (reinitialize) of the axis is done. Here all the internal errors are reset.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Reset axis – Edge 0-1: Axis reset is performed
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Reset was performed
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

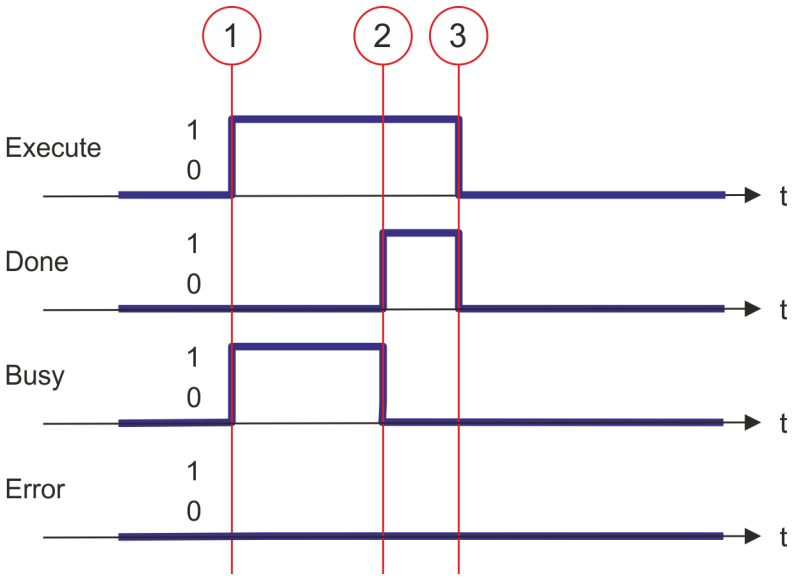
- Job start in PLCopen-State *ErrorStop* possible.
- MC_Reset switches the axis depending on MC_Power either to PLCopen-State *Standstill* (call MC_Power with *Enable* = TRUE) or *Disabled* (call MC_Power with *Enable* = FALSE).

Perform reset on axis

The reset of the axis is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as the reset of the axis is running. After axis has been reinitialized, *Busy* with FALSE and *Done* with TRUE is returned.

An active job continues until it is finished even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the reset of the axis is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) the reset is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.11 FB 812 - MC_ReadStatus - PLCopen status

Description With MC_ReadStatus the PLCopen-State of the axis can be determined

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the slave axis
Enable	INPUT	BOOL	■ Status indication – TRUE: The status is permanently displayed at the outputs – FALSE: All the outputs are FALSE respectively 0
Valid	OUTPUT	BOOL	■ State is valid – TRUE: The shown state is valid
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
ErrorStop	OUTPUT	BOOL	■ Axis errors – TRUE: An axis error has occurred, move job can not be activated
Disabled	OUTPUT	BOOL	■ Status axis: Disabled – TRUE: Axis is disabled, move job can not be activated
Stopping	OUTPUT	BOOL	■ Status axis: Stop – TRUE: Axis is stopped (MC_Stop is active)
Homing	OUTPUT	BOOL	■ Status axis: Homing – TRUE: Axis is just homing (MC_Homing is active)
Standstill	OUTPUT	BOOL	■ Status move job – TRUE: No move job is active; a move job can be activated
DiscreteMotion	OUTPUT	BOOL	■ Status axis motion: Discrete – TRUE: Axis is moved by a discrete movement (MC_MoveRelative, MC_MoveAbsolute or MC_Halt is active)
ContinuousMotion	OUTPUT	BOOL	■ Status axis motion: Continuous – TRUE: Axis is moved by a continuous movement (MC_MoveVelocity is active)
SynchronizedMotion	OUTPUT	BOOL	■ Status axis: Slave axis – TRUE: Axis is a slave axis (MC_CamIn or MC_GearIn is active)

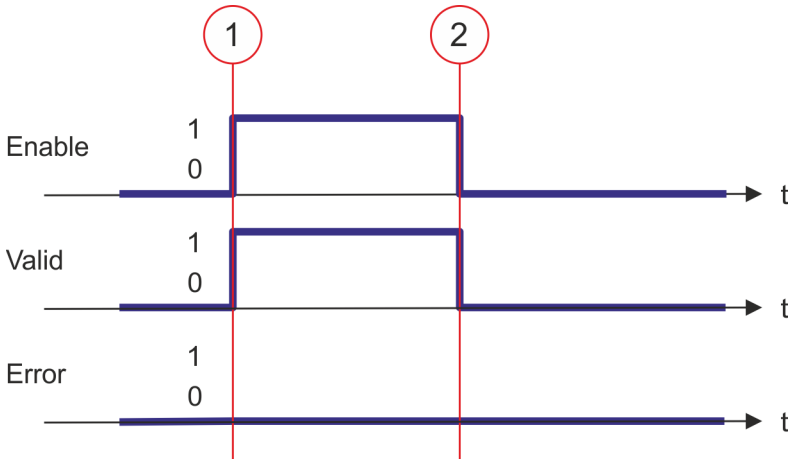
PLCopen-State

- Job start in each PLCopen-State possible.

Determine the status of the axis

With *Enable* = TRUE the outputs represent the state of the axis according to the PLCopen-State diagram.

Status diagram of the block parameters



(1) At time (1) *Enable* is set to TRUE. So *Valid* gets TRUE and the outputs correspond to the status of the PLCopen-State.
(2) At time (2) *Enable* is set to FALSE. So all the outputs are set to FALSE respectively 0.

6.3.12 FB 813 - MC_ReadAxisError - read axis error

Description With MC_ReadAxisError the current error of the axis is directly be read.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Reset axis – Edge 0-1: Axis error is read.
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Axis error read.
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running.
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*
AxisErrorID	OUTPUT	WORD	Axis error ID; the read value is vendor-specifically encoded.

PLCopen-State

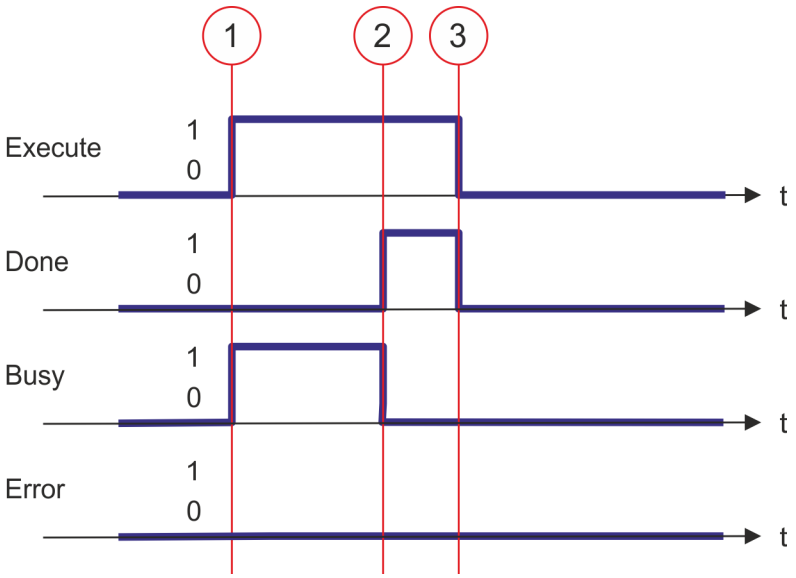
- Job start in each PLCopen-State possible.

Read error of the axis

The reading of the error of the axis is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as reading of the axis error is running. After the axis error was read, *Busy* with FALSE and *Done* with TRUE is returned. The output *AxisErrorID* shows the current axis error.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the reading of the axis error is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) reading of the axis error is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.13 FB 814 - MC_ReadParameter - read axis parameter data

Description

With MC_ReadParameter the parameter, that is defined by the parameter number, is read from the axis. *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Read axis parameter data – Edge 0-1: The parameter data is read
Parameter Number	INPUT	INT	Number of the parameter to be read. *Chapter 6.3.33 'PLCopen parameter' on page 186*
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was read
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
Value	OUTPUT	REAL	Value of the read parameter

PLCopen-State

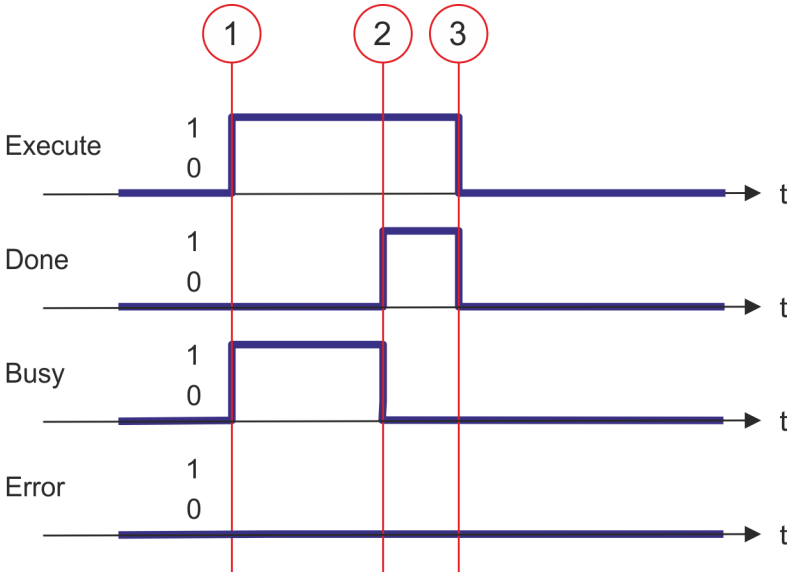
■ Job start in each PLCopen-State possible.

Read axis parameter data

The reading of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as reading of parameter data is running. After the parameter data was read, *Busy* with FALSE and *Done* with TRUE is returned. The output *Value* shows the value of the parameter.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the reading of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) reading of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.14 FB 815 - MC_WriteParameter - write axis parameter data

Description

With MC_WriteParameter the value of the parameter, that is defined by the parameter number, is written to the axis. *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Write axis parameter data – Edge 0-1: The parameter data is written
Parameter Number	INPUT	INT	Number of the parameter to be written. *Chapter 6.3.33 'PLCopen parameter' on page 186*
Value	INPUT	REAL	Value of the written parameter
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was written
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

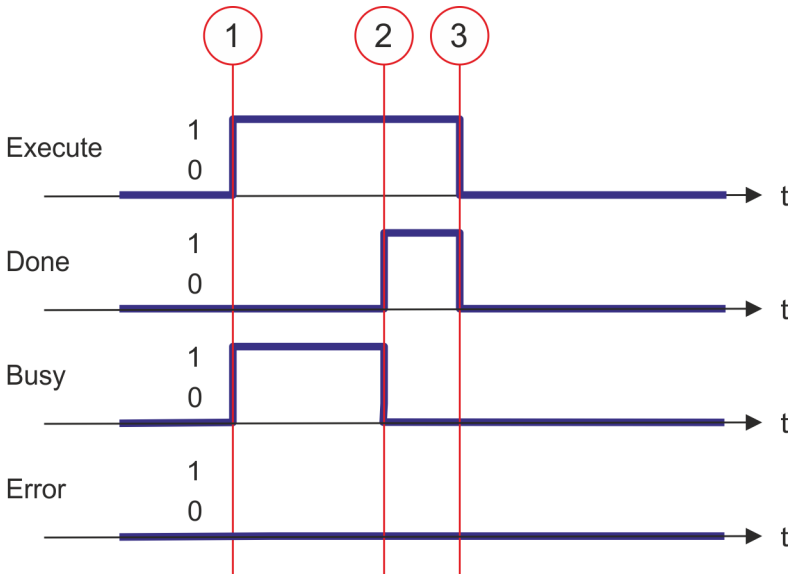
■ Job start in each PLCopen-State possible.

Write axis parameter data

The writing of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as writing of parameter data is running. After the parameter data was written, *Busy* with FALSE and *Done* with TRUE is returned.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the writing of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) writing of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.15 FB 816 - MC_ReadActualPosition - reading current axis position

Description With MC_ReadActualPosition the current position of the axis is read.

Parameter

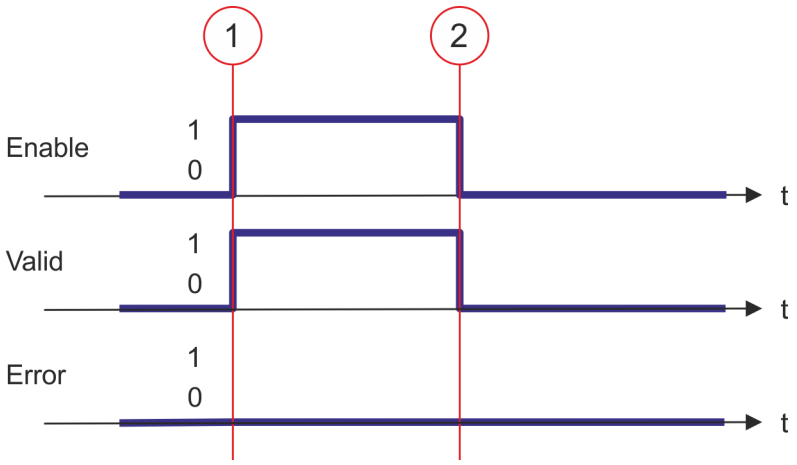
Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Enable	INPUT	BOOL	■ Read axis position – TRUE: The position of the axis is continuously read – FALSE: All the outputs are FALSE respectively 0
Valid	OUTPUT	BOOL	■ Position valid – TRUE: The read position is valid
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
Position	OUTPUT	REAL	Position of the axis [user unit]

PLCopen-State

- Job start in each PLCopen-State possible.

Read axis position The current axis position is determined and stored at *Position* with *Enable* set to TRUE.

Status diagram of the block parameters



(1) At time (1) *Enable* is set to TRUE. So *Valid* gets TRUE and output *Position* corresponds to the current axis position.
(2) At time (2) *Enable* is set to FALSE. So all the outputs are set to FALSE respectively 0.

6.3.16 FB 817 - MC_ReadActualVelocity - read axis velocity

Description With MC_ReadActualVelocity the current velocity of the axis is read.

Parameter

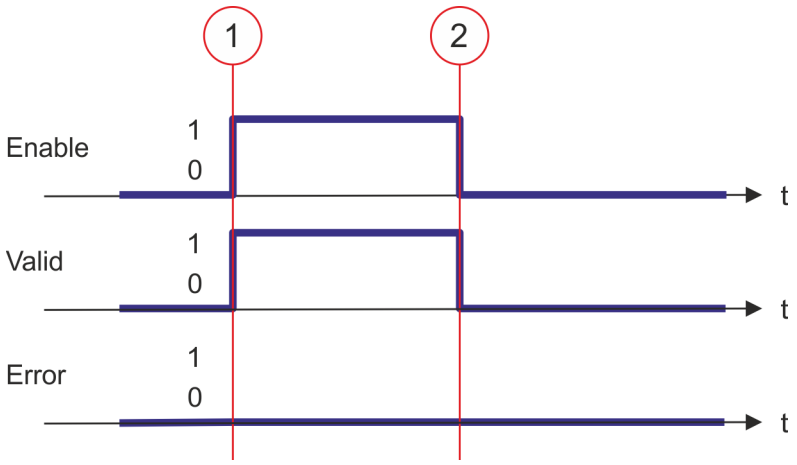
Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Enable	INPUT	BOOL	■ Read axis velocity – TRUE: The velocity of the axis is continuously read – FALSE: All the outputs are FALSE respectively 0
Valid	OUTPUT	BOOL	■ Velocity valid – TRUE: The read velocity is valid
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*
Velocity	OUTPUT	REAL	Velocity of the axis [user unit/s]

PLCopen-State

- Job start in each PLCopen-State possible.

Read axis velocity The current axis velocity is determined and stored at *Velocity* with *Enable* set to TRUE.

Status diagram of the block parameters



(1) At time (1) *Enable* is set to TRUE. So *Valid* gets TRUE and output *Velocity* corresponds to the current axis velocity.
(2) At time (2) *Enable* is set to FALSE. So all the outputs are set to FALSE respectively 0.

6.3.17 FB 818 - MC_ReadAxisInfo - read additional axis information

Description With MC_ReadAxisInfo some additional information of the axis are shown.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Enable	INPUT	BOOL	■ Read additional information from axis – TRUE: The additional information of the axis are read – FALSE: All the outputs are FALSE respectively 0
Valid	OUTPUT	BOOL	■ Additional information valid – TRUE: The read additional information are valid
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
HomeAbsSwitch	OUTPUT	BOOL	Homing switch ■ TRUE: Homing switch is activated
LimitSwitchPos	OUTPUT	BOOL	Limit switch positive direction ■ TRUE: Limit switch positive direction is activated
LimitSwitchNeg	OUTPUT	BOOL	Limit switch negative direction (NOT bit of the drive) ■ TRUE: Limit switch negative direction is activated
Simulation	OUTPUT	BOOL	Parameter is currently not supported; always FALSE
Communication-Ready	OUTPUT	BOOL	■ Information axis: Data exchange – TRUE: Data exchange with axis is initialized; axis is ready for communication
ReadyForPo-werOn	OUTPUT	BOOL	■ Information axis: Enable possible – TRUE: Enabling the axis is possible
PowerOn	OUTPUT	BOOL	■ Information axis: Enabled – TRUE: Enabling of the axis is carried out
IsHomed	OUTPUT	BOOL	■ Information axis: Homed – TRUE: The axis is homed
AxisWarning	OUTPUT	BOOL	■ Information axis: Error – TRUE: At least 1 error is reported from the axis

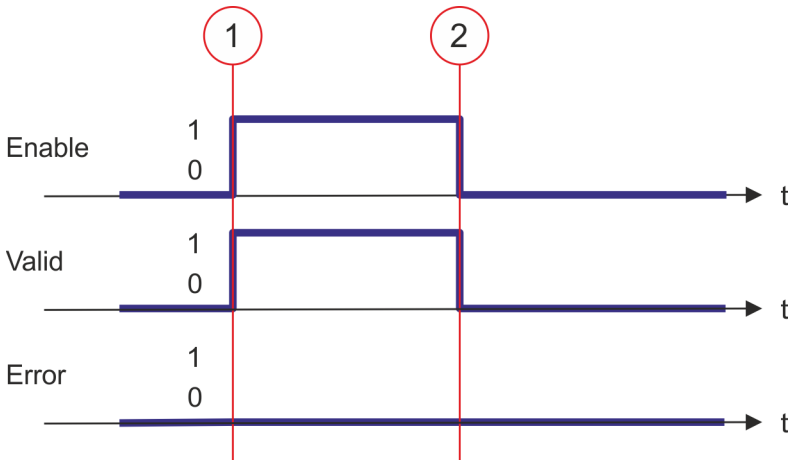
PLCopen-State

■ Job start in each PLCopen-State possible.

Determine the status of the axis

The additional information of the axis are shown at the outputs with *Enable* set to TRUE.

Status diagram of the block parameters



(1) At time (1) *Enable* is set to TRUE. So *Valid* gets TRUE and the outputs show the additional information of the axis.
(2) At time (2) *Enable* is set to FALSE. So all the outputs are set to FALSE respectively 0.

6.3.18 FB 819 - MC_ReadMotionState - read status motion job

Description With MC_ReadMotionState the current status of the motion job is shown.

Parameter

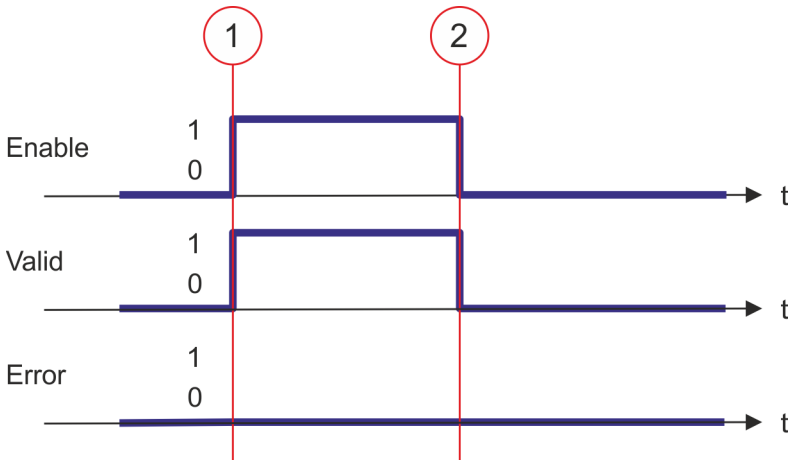
Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Enable	INPUT	BOOL	■ Read motion state – TRUE: The status of the motion job is continuously read – FALSE: All the outputs are FALSE respectively 0
Source	INPUT	Byte	Only Source = 0 is supported; at the outputs the current status of the motion job is shown.
Valid	OUTPUT	BOOL	■ Status valid – TRUE: The read status of the motion job is valid
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
ConstantVelocity	OUTPUT	BOOL	■ Status motion job: Velocity – TRUE: Velocity is constant
Acceleration	OUTPUT	BOOL	■ Status motion job: Acceleration – TRUE: The axis is accelerated; the velocity of the axis is increasing
Decelerating	OUTPUT	BOOL	■ Status motion job: Braking process – TRUE: Axis is decelerated; the velocity of the axis is getting smaller
DirectionPositive	OUTPUT	BOOL	■ Status motion job: Position increasing – TRUE: The position of the axis is increasing
DirectionNegative	OUTPUT	BOOL	■ Status motion job: Position decreasing – TRUE: The position of the axis is decreasing

PLCopen-State

- Job start in each PLCopen-State possible.

Read status of the motion job With *Enable* = TRUE the outputs represent the status of the motion job of the axis.

Status diagram of the block parameters



(1) At time (1) *Enable* is set to TRUE. So *Valid* gets TRUE and the outputs correspond to the status of motion job.
(2) At time (2) *Enable* is set to FALSE. So all the outputs are set to FALSE respectively 0.

6.3.19 FB 823 - MC_TouchProbe - record axis position

Description

This function block is used to record an axis position at a trigger event. The trigger signal can be configured via the variable specified at the input *TriggerInput*. As trigger signal can serve e.g. a digital input or a encoder zero track.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis.
TriggerInput	IN_OUT	MC_TRIGGER_REF	Reference to the trigger input. Structure ■ .Probe – 01: TouchProbe register 1 – 02: TouchProbe register 2 ■ .TriggerSource – 00: Input – 00: Encoder zero pulse ■ .Triggermode – 00: SingleTrigger (fix) ■ .Reserved (0 fix)
Execute	IN	BOOL	The recording of the axis position is activated with edge 0-1 at *Execute*.
Done	OUT	BOOL	■ Status – TRUE: Job successfully done. The axis position was recorded.
Busy	OUT	BOOL	■ Status – TRUE: Job is running.
CommandA-borted	OUT	BOOL	■ Status – TRUE: The job was aborted during processing by another job.
Error	OUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
RecordedPosi-tion	OUT	REAL	Recorded axis position where trigger event occurred [user units].

- *An active job continues to run until this is completed, even when Execute is set to FALSE. The detected axis position is the output at RecordedPosition for one cycle.* *Chapter 7.3 'Behavior of the inputs and outputs' on page 193*
- *Thus the job can be executed, the communication to the axis must be OK and the PLCopen-State must be unequal Homing.*
- *A running job can be aborted with a new MC_TouchProbe job for the same axis.*
- *A running job can be aborted by MC_AbortTrigger.*
- *A running job can be aborted by MC_Home.*

Recording the axis position

The recording of the axis position is activated with edge 0-1 at *Execute*. *Busy* is TRUE as soon as the job is running. After processing the job, *Busy* with FALSE and *Done* with TRUE is returned. The recorded value can be found in *RecordedPosition*.

6.3.20 FB 824 - MC_AbortTrigger - abort recording axis position

Description

This block aborts the recording of the axis position, which was started via MC_TouchProbe.

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis.
TriggerInput	IN_OUT	MC_TRIGGER_REF	Reference to the trigger input. Structure ■ .Probe – 01: TouchProbe register 1 – 02: TouchProbe register 2 ■ .TriggerSource – 00: Input – 00: Encoder zero pulse ■ .Triggermode – 00: SingleTrigger (fix) ■ .Reserved (0 fix)
Execute	IN	BOOL	The recording of the axis position is aborted with edge 0-1 at *Execute*.
Done	OUT	BOOL	■ Status – TRUE: Job successfully done. The recording of the axis position was aborted.
Busy	OUT	BOOL	■ Status – TRUE: Job is running.
Error	OUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

Thus the job can be executed, the communication to the axis must be OK.

Abort the recording of the axis position

The recording of the axis position is aborted with edge 0-1 at *Execute*. *Busy* is TRUE as soon as the job is running. After processing the job, *Busy* with FALSE and *Done* with TRUE is returned.

6.3.21 FB 825 - MC_ReadBoolParameter - read axis boolean parameter data

Description

With MC_ReadBoolParameter the parameter of data type BOOL, that is defined by the parameter number, is read from the axis. ↳ *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Read axis parameter data – Edge 0-1: The parameter data is read
Parameter Number	INPUT	INT	Number of the parameter to be read. ↳ *Chapter 6.3.33 'PLCopen parameter' on page 186*
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was read
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*
Value	OUTPUT	BOOL	Value of the read parameter

PLCopen-State

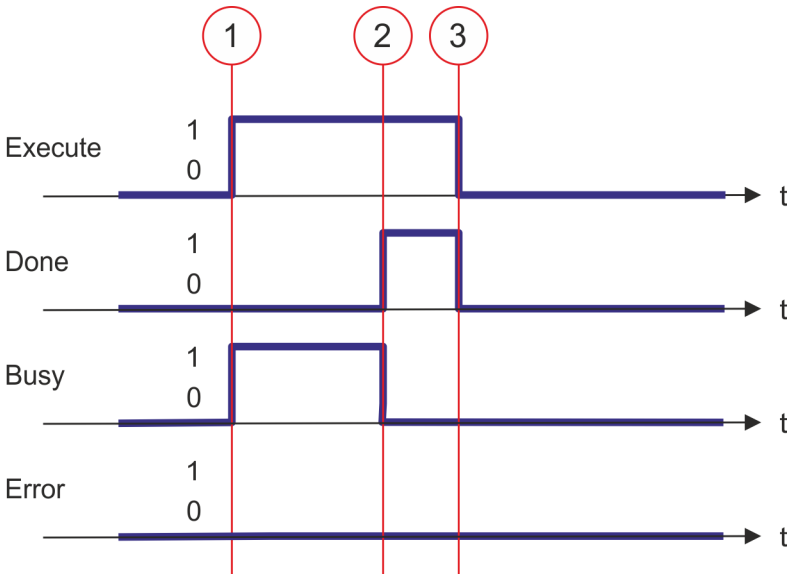
■ Job start in each PLCopen-State possible.

Read axis parameter data

The reading of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as reading of parameter data is running. After the parameter data was read, *Busy* with FALSE and *Done* with TRUE is returned. The output *Value* shows the value of the parameter.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the reading of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) reading of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.22 FB 826 - MC_WriteBoolParameter - write axis boolean parameter data

Description

With MC_WriteBoolParameter the value of the parameter of data type BOOL, that is defined by the parameter number, is written to the axis. ↳ *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Write axis parameter data – Edge 0-1: The parameter data is written
Parameter Number	INPUT	INT	Number of the parameter to be written. ↳ *Chapter 6.3.33 'PLCopen parameter' on page 186*
Value	INPUT	BOOL	Value of the written parameter
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was written
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

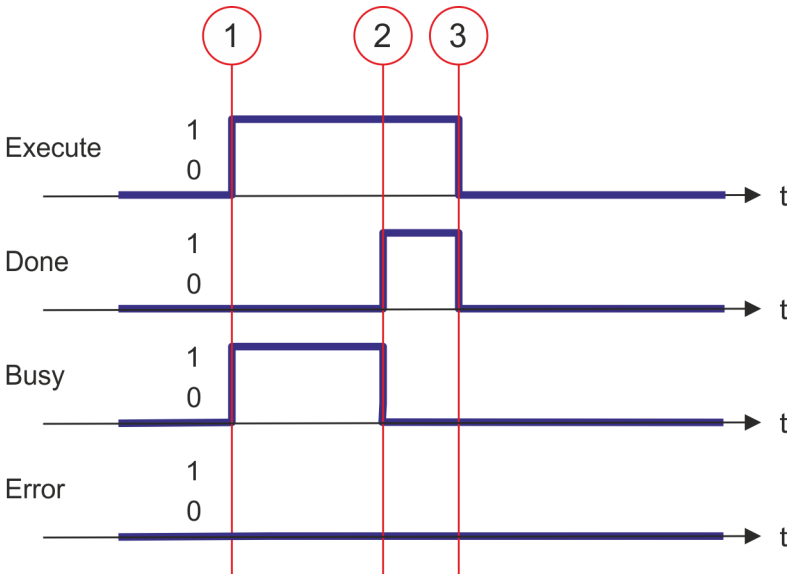
■ Job start in each PLCopen-State possible.

Write axis parameter data

The writing of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as writing of parameter data is running. After the parameter data was written, *Busy* with FALSE and *Done* with TRUE is returned.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the writing of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) writing of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.23 FB 827 - VMC_ReadDWordParameter - read axis double word parameter data

Description

With MC_ReadDWordParameter the parameter of data type DWORD, that is defined by the parameter number, is read from the axis. *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Read axis parameter data – Edge 0-1: The parameter data is read
Parameter-Number	INPUT	INT	Number of the parameter to be read. *Chapter 6.3.33 'PLCopen parameter' on page 186*
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was read
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
Value	OUTPUT	DWORD	Value of the read parameter

PLCopen-State

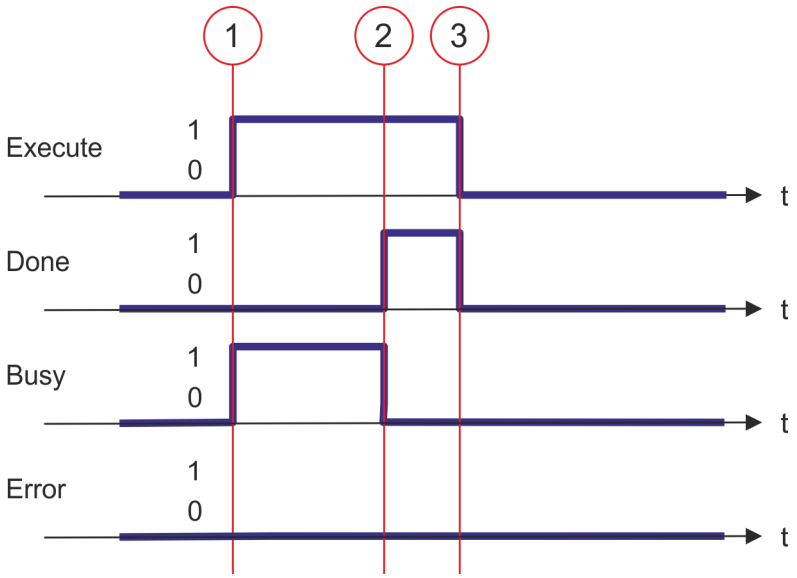
■ Job start in each PLCopen-State possible.

Read axis parameter data

The reading of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as reading of parameter data is running. After the parameter data was read, *Busy* with FALSE and *Done* with TRUE is returned. The output *Value* shows the value of the parameter.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the reading of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) reading of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.24 FB 828 - VMC_WriteDWordParameter - write axis double word parameter data

Description

With VMC_WriteDWordParameter the value of the parameter of data type DWORD, that is defined by the parameter number, is written to the axis. *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Write axis parameter data – Edge 0-1: The parameter data is written
Parameter Number	INPUT	INT	Number of the parameter to be written. *Chapter 6.3.33 'PLCopen parameter' on page 186*
Value	INPUT	DWORD	Value of the written parameter
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was written
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

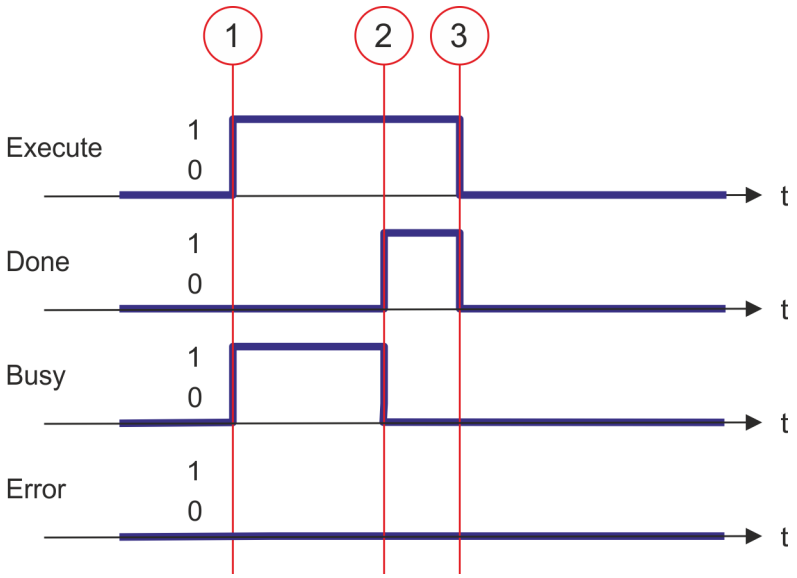
■ Job start in each PLCopen-State possible.

Write axis parameter data

The writing of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as writing of parameter data is running. After the parameter data was written, *Busy* with FALSE and *Done* with TRUE is returned.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the writing of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) writing of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.25 FB 829 - VMC_ReadWordParameter - read axis word parameter data

Description

With VMC_ReadWordParameter the parameter of data type WORD, that is defined by the parameter number, is read from the axis. *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Read axis parameter data – Edge 0-1: The parameter data is read
Parameter Number	INPUT	INT	Number of the parameter to be read. *Chapter 6.3.33 'PLCopen parameter' on page 186*
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was read
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
Value	OUTPUT	WORD	Value of the read parameter

PLCopen-State

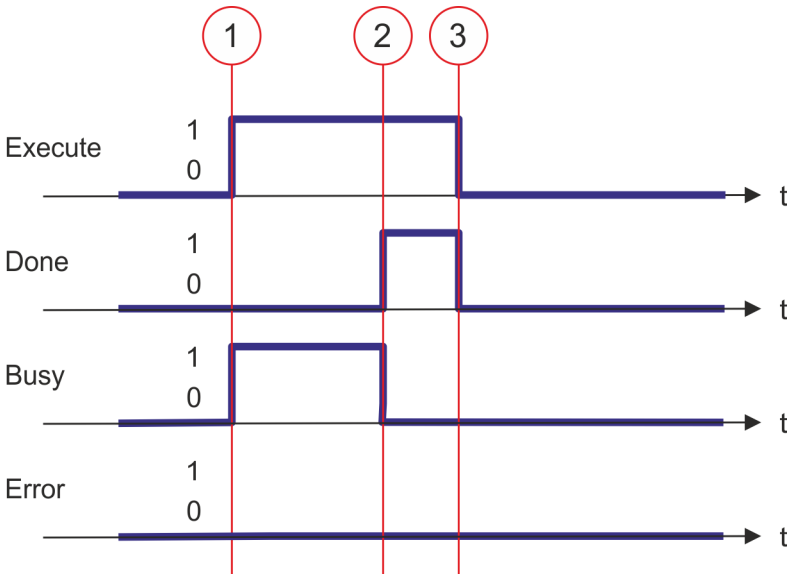
■ Job start in each PLCopen-State possible.

Read axis parameter data

The reading of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as reading of parameter data is running. After the parameter data was read, *Busy* with FALSE and *Done* with TRUE is returned. The output *Value* shows the value of the parameter.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the reading of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) reading of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.26 FB 830 - VMC_WriteWordParameter - write axis word parameter data

Description

With VMC_WriteWordParameter the value of the parameter of data type WORD, that is defined by the parameter number, is written to the axis. *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Write axis parameter data – Edge 0-1: The parameter data is written
Parameter Number	INPUT	INT	Number of the parameter to be written. *Chapter 6.3.33 'PLCopen parameter' on page 186*
Value	INPUT	WORD	Value of the written parameter
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was written
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

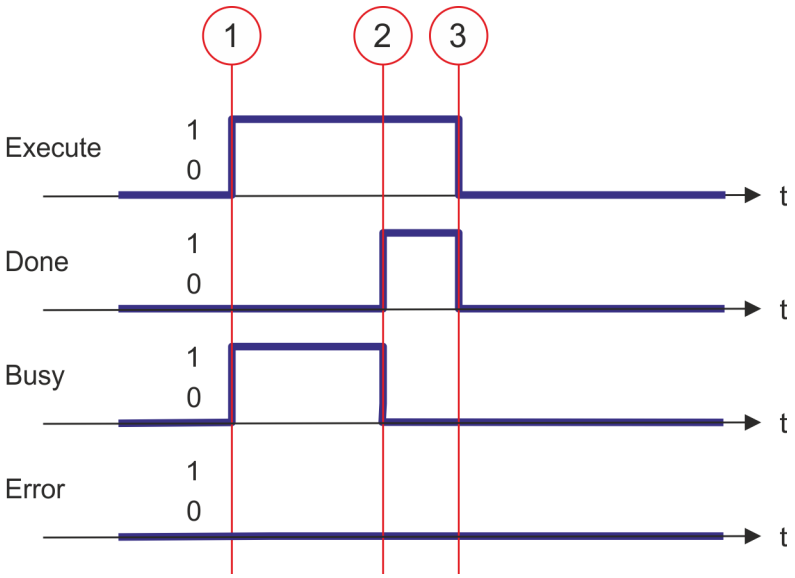
■ Job start in each PLCopen-State possible.

Write axis parameter data

The writing of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as writing of parameter data is running. After the parameter data was written, *Busy* with FALSE and *Done* with TRUE is returned.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the writing of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) writing of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.27 FB 831 - VMC_ReadByteParameter - read axis byte parameter data

Description

With VMC_ReadByteParameter the parameter of data type BYTE, that is defined by the parameter number, is read from the axis. ↳ *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Read axis parameter data – Edge 0-1: The parameter data is read
Parameter Number	INPUT	INT	Number of the parameter to be read. ↳ *Chapter 6.3.33 'PLCopen parameter' on page 186*
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was read
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*
Value	OUTPUT	BYTE	Value of the read parameter

PLCopen-State

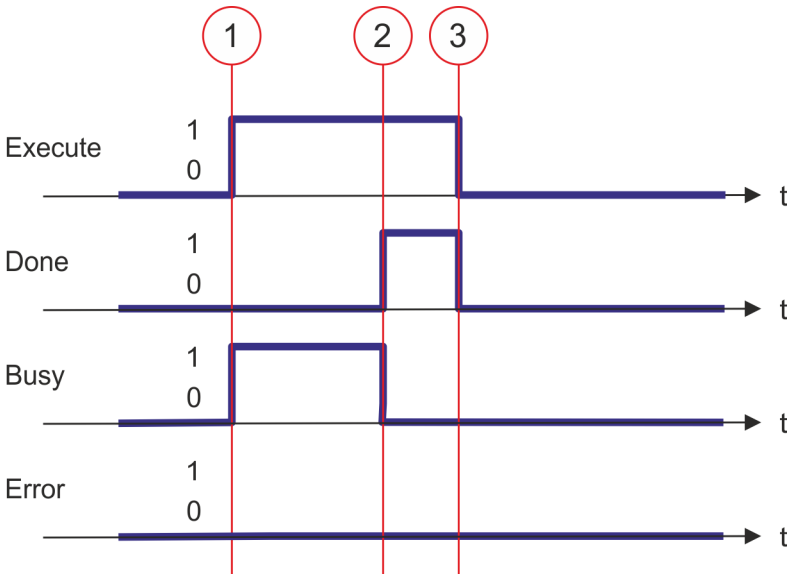
■ Job start in each PLCopen-State possible.

Read axis parameter data

The reading of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as reading of parameter data is running. After the parameter data was read, *Busy* with FALSE and *Done* with TRUE is returned. The output *Value* shows the value of the parameter.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the reading of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) reading of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.28 FB 832 - VMC_WriteByteParameter - write axis byte parameter data

Description

With VMC_WriteByteParameter the value of the parameter of data type BYTE, that is defined by the parameter number, is written to the axis. *Chapter 6.3.33 'PLCopen parameter' on page 186*

Parameter

Parameter	Declaration	Data type	Description
Axis	IN_OUT	MC_AXIS_REF	Reference to the axis
Execute	INPUT	BOOL	■ Write axis parameter data – Edge 0-1: The parameter data is written
Parameter Number	INPUT	INT	Number of the parameter to be written. *Chapter 6.3.33 'PLCopen parameter' on page 186*
Value	INPUT	BYTE	Value of the written parameter
Done	OUTPUT	BOOL	■ Status – TRUE: Job successfully done. Parameter data was written
Busy	OUTPUT	BOOL	■ Status – TRUE: Job is running
Error	OUTPUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUTPUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*

PLCopen-State

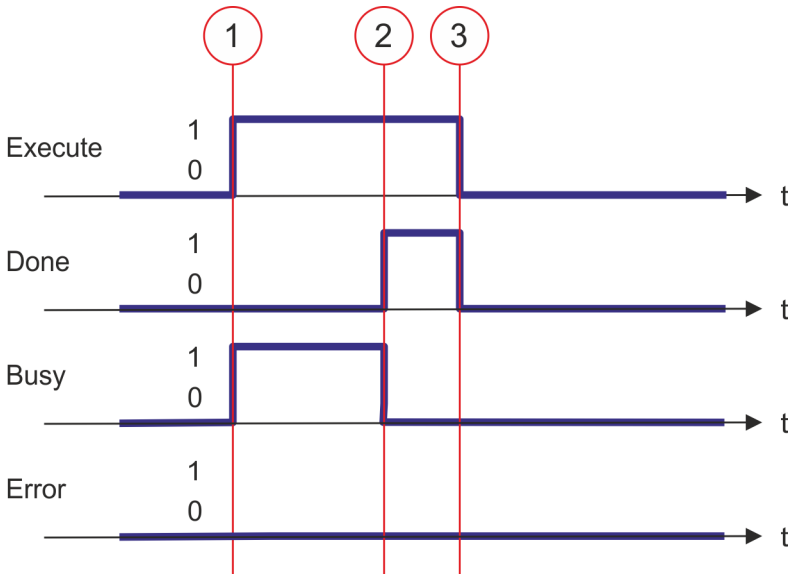
■ Job start in each PLCopen-State possible.

Write axis parameter data

The writing of the axis parameter data is started with an edge 0-1 at *Execute*. *Busy* is TRUE as soon as writing of parameter data is running. After the parameter data was written, *Busy* with FALSE and *Done* with TRUE is returned.

An active job continues to run even when Execute is set to FALSE.

Status diagram of the block parameters



(1) At time (1) the writing of the parameter data is started with edge 0-1 at *Execute* and *Busy* becomes TRUE.
(2) At the time (2) writing of the parameter data is successfully completed. *Busy* has the value FALSE and *Done* den value TRUE.
(3) At the time (3) the job is completed and *Execute* becomes FALSE and thus each output parameter FALSE respectively 0.

6.3.29 FB 835 - VMC_HomeInit_LimitSwitch - Initialisation of homing on limit switch

Description This block initialises homing on limit switch.

Parameters

Parameter	Declaration	Data type	Description
Execute	IN	BOOL	■ Initialisation of the homing method – Edge 0-1: Values of the input parameter are accepted and the initialisation of the homing method is started.
Direction	IN	BOOL	■ Direction of homing – TRUE: on positive limit switch – FALSE: on negative limit switch
Velocity-SearchSwitch	IN	REAL	Velocity for search for the switch in [user units/s]
VelocitySearch-Zero	IN	REAL	Velocity for search for zero in [user units/s]
Acceleration	IN	REAL	Acceleration in [user units/s^2]
Done	OUT	BOOL	■ Status – TRUE: Initialisation successfully done.
Busy	OUT	BOOL	■ Status – TRUE: Initialisation is active.
Error	OUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*
AXIS	IN_OUT	MC_AXIS_REF	Reference to the axis

Initialisation homing on limit switch

The values of the input parameters are accepted with an edge 0-1 at *Execute* and the initialisation of the homing method is started. As long as the initialisation is active, the output *Busy* is set to TRUE. If the initialisation has been completed successfully, the output *Done* is set to TRUE. If an error occurs during initialisation, the output *Error* is set to TRUE and an error number is output at the output *ErrorID*.

Initialisation of the homing method

1. Verify communication to the axis.
2. Check for permitted PLCopen states.
3. Check the input values:
   - Input VelocitySearchSwitch [UserUnits] > 0.0
   - VelocitySearchSwitch [InternalUnits] > 0
   - VelocitySearchSwitch [InternalUnits] ≤ VelocityMax
   - Input VelocitySearchZero [UserUnits] > 0.0
   - VelocitySearchZero [InternalUnits] > 0
   - VelocitySearchZero [InternalUnits] ≤ VelocityMax
   - Input Acceleration [UserUnits] > 0.0
   - Acceleration [InternalUnits] > 0
   - Acceleration [InternalUnits] ≤ AccelerationMax
4. Transfer of the drive parameters:
   - "Homing Method" in dependence of input "Direction" See table below!
   - "Homing Speed during search for switch" [Inc/s]
   - "Homing Speed during search for zero" [Inc/s]
   - "Homing Acceleration" [Inc/s^2]

Homing Method	Direction
1	false
2	true

6.3.30 FB 836 - VMC_HomeInit_HomeSwitch - Initialisation of homing on home switch

Description This block initialises homing on home switch.

Parameters

Parameter	Declaration	Data type	Description
Execute	IN	BOOL	■ Initialisation of the homing method – Edge 0-1: Values of the input parameter are accepted and the initialisation of the homing method is started.
InitialDirection	IN	BOOL	■ Initial direction of homing – TRUE: on positive limit switch – FALSE: on negative limit switch
WithIndexPulse	IN	BOOL	■ Homing – TRUE: homing with index pulse – FALSE: homing without index pulse
OnRisingEdge	IN	BOOL	■ Edge of home switch – TRUE: Edge 0-1 – FALSE: Edge 1-0
SameDirIndex-Pulse	IN	BOOL	■ Search for index pulse – TRUE: After detecting the home, search for index pulse without change of direction – FALSE: After detecting the home, search for index pulse with change of direction
Velocity-SearchSwitch	IN	REAL	Velocity for search for the switch in [user units/s]
VelocitySearch-Zero	IN	REAL	Velocity for search for zero in [user units/s]
Acceleration	IN	REAL	Acceleration in [user units/s^2]
Done	OUT	BOOL	■ Status – TRUE: Initialisation successfully done.
Busy	OUT	BOOL	■ Status – TRUE: Initialisation is active.
Error	OUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUT	WORD	Additional error information ↳ *Chapter 8 'ErrorID - Additional error information' on page 195*
AXIS	IN_OUT	MC_AXIS_REF	Reference to the axis

Initialisation homing on home switch

The values of the input parameters are accepted with an edge 0-1 at *Execute* and the initialisation of the homing method is started. As long as the initialisation is active, the output *Busy* is set to TRUE. If the initialisation has been completed successfully, the output *Done* is set to TRUE. If an error occurs during initialisation, the output *Error* is set to TRUE and an error number is output at the output *ErrorID*.

Initialisation of the homing method

1. Verify communication to the axis.
2. Check for permitted PLCopen states.
3. Check the input values:
   - Input VelocitySearchSwitch [UserUnits] > 0.0
   - VelocitySearchSwitch [InternalUnits] > 0
   - VelocitySearchSwitch [InternalUnits] ≤ VelocityMax
   - Input VelocitySearchZero [UserUnits] > 0.0
   - VelocitySearchZero [InternalUnits] > 0
   - VelocitySearchZero [InternalUnits] ≤ VelocityMax
   - Input Acceleration [UserUnits] > 0.0
   - Acceleration [InternalUnits] > 0
   - Acceleration [InternalUnits] ≤ AccelerationMax
4. Transfer of the drive parameters:
   - "Homing Method" in dependence of input "Direction" See Table below!
   - "Homing Speed during search for switch" [Inc/s]
   - "Homing Speed during search for zero" [Inc/s]
   - "Homing Acceleration" [Inc/s^2]

Homing Method	InitialDirection	WithIndexPulse	OnRisingEdge	SameDirIndexPulse
7	positive	true	true	false
8	positive	true	true	true
9	positive	true	false	false
10	positive	true	false	true
11	negative	true	true	false
12	negative	true	true	true
13	negative	true	false	false
14	negative	true	false	true
24	positive	false	true	false
24	positive	false	true	true
24	positive	false	false	false
24	positive	false	false	true
28	negative	false	true	false
28	negative	false	true	true
28	negative	false	false	false
28	negative	false	false	true

6.3.31 FB 837 - VMC_HomeInit_ZeroPulse - Initialisation of homing on zero puls

Beschreibung This block initialises homing on zero pulse.

Parameters

Parameter	Declaration	Data type	Description
Execute	IN	BOOL	■ Initialisation of the homing method – Edge 0-1: Values of the input parameter are accepted and the initialisation of the homing method is started.
Direction	IN	BOOL	■ Direction of homing – TRUE: Positive direction – FALSE: Negative direction
VelocitySearch-Zero	IN	REAL	Velocity for search for zero in [user units/s]
Acceleration	IN	REAL	Acceleration in [user units/s^2]
Done	OUT	BOOL	■ Status – TRUE: Initialisation successfully done.
Busy	OUT	BOOL	■ Status – TRUE: Initialisation is active.
Error	OUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter *ErrorID*.
ErrorID	OUT	WORD	Additional error information ↬ *Chapter 8 'ErrorID - Additional error information' on page 195*
AXIS	IN_OUT	MC_AXIS_REF	Reference to the axis

Initialisation homing on zero puls

The values of the input parameters are accepted with an Edge 0-1 at *Execute* and the initialisation of the homing method is started. As long as the initialisation is active, the output *Busy* is set to TRUE. If the initialisation has been completed successfully, the output *Done* is set to TRUE. If an error occurs during initialisation, the output *Error* is set to TRUE and an error number is output at the output *ErrorID*.

Initialisation of the homing method

1. Verify communication to the axis.
2. Check for permitted PLCopen states.
3. Check the input values:
   - Input VelocitySearchZero [UserUnits] > 0.0
   - VelocitySearchZero [InternalUnits] > 0
   - VelocitySearchZero [InternalUnits] ≤ VelocityMax
   - Input Acceleration [UserUnits] > 0.0
   - Acceleration [InternalUnits] > 0
   - Acceleration [InternalUnits] ≤ AccelerationMax

4. Transfer of the drive parameters:
   - "Homing Method" in dependence of input "Direction" See table below!
   - "Homing Speed during search for switch" [Inc/s]
   - "Homing Speed during search for zero" [Inc/s]
   - "Homing Acceleration" [Inc/s^2]

Homing Method	Direction
33	false
34	true

6.3.32 FB 838 - VMC_HomeInit_SetPosition - Initialisation of homing mode set position

Description This block initialises homing on current position.

Parameters

Parameter	Declaration	Data type	Description
Execute	IN	BOOL	■ Initialisation of the homing method – Edge 0-1: Values of the input parameter are accepted and the initialisation of the homing method is started.
Done	OUT	BOOL	■ Status – TRUE: Initialisation successfully done.
Busy	OUT	BOOL	■ Status – TRUE: Initialisation is active.
Error	OUT	BOOL	■ Status – TRUE: An error has occurred. Additional error information can be found in the parameter ErrorID.
ErrorID	OUT	WORD	Additional error information *Chapter 8 'ErrorID - Additional error information' on page 195*
AXIS	IN_OUT	MC_AXIS_REF	Reference to the axis

Initialisation homing on home switch

The values of the input parameters are accepted with an edge 0-1 at *Execute* and the initialisation of the homing method is started. As long as the initialisation is active, the output *Busy* is set to TRUE. If the initialisation has been completed successfully, the output *Done* is set to TRUE. If an error occurs during initialisation, the output *Error* is set to TRUE and an error number is output at the output *ErrorID*.

Initialisation of the homing method

1. Verify communication to the axis.
2. Check for permitted PLCopen states.
3. Transfer of the drive parameters:
   - "Homing Method" = 35

6.3.33 PLCopen parameter

PN	Name	Data type	R/W	Comments
1	CommandedPosition	REAL	R	Commanded position Access on: `#Axis.Status.Positioning.SetValues.CommandedPosition`
2	SWLimitPos	REAL	R/W	Positive software limit switch position Access on: `"Axis".AxisConfiguration.PositionLimits.MaxPosition`

PN	Name	Data type	R/W	Comments
3	SWLimitNeg	REAL	R/W	Negative software limit switch position Access on: `"Axis".AxisConfiguration.PositionLimits.MinPosition`
4	EnableLimitPos	BOOL	R/W	Enable positive software limit switch Access on: `"Axis".AxisConfiguration.PositionLimits.EnableMaxPos`
5	EnableLimitNeg	BOOL	R/W	Enable negative software limit switch Access on: `"Axis".AxisConfiguration.PositionLimits.EnableMinPos`
6	EnablePosLagMonitoring	BOOL	R/W	Enable monitoring of position lag Function is not supported
7	MaxPositionLag	REAL	R/W	Maximal position lag Function is not supported
8	MaxVelocitySystem	REAL	R	Maximal allowed velocity of the axis in the motion system This parameter is currently not supported
9	MaxVelocityAppl	REAL	R/W	Maximal allowed velocity of the axis in the application Access on: `#Axis.AxisConfiguration.DynamicLimits.MaxVelocityApp`
10	ActualVelocity	REAL	R	Actual velocity Access on: `#Axis.Status.Positioning.ActValues.Velocity`
11	CommandedVelocity	REAL	R	Commanded velocity Access on: `#Axis.Status.Positioning.SetValues.Velocity`
12	MaxAccelerationSystem	REAL	R	Maximal allowed acceleration of the axis in the motion system This parameter is currently not supported
13	MaxAccelerationAppl	REAL	R/W	Maximal allowed acceleration of the axis in the application Access on: `#Axis.AxisConfiguration.DynamicLimits.MaxAccelerationApp`
14	MaxDecelerationSystem	REAL	R	Maximal allowed deceleration of the axis in the motion system This parameter is currently not supported
15	MaxDecelerationAppl	REAL	R/W	Maximal allowed deceleration of the axis in the application Access on: `#Axis.AxisConfiguration.DynamicLimits.MaxDecelerationApp`

PN	Name	Data type	R/W	Comments
16	MaxJerkSystem	REAL	R	Maximum allowed jerk of the axis in the motion system This parameter is currently not supported
17	MaxJerkAppl	REAL	R/W	Maximum allowed jerk of the axis in the application This parameter is currently not supported.

6.3.34 VIPA-specific parameter

Positioning axis: Yaskawa *Sigma-5* / *Sigma-7* via EtherCAT

No.	Name	Data type	Index	Subindex	Access
900	HomingDone	BOOL	-	-	R/W [1, 2]
901	PositiveTorqueLimit	BOOL	-	-	R/W [1, 2]
902	NegativeTorqueLimit	BOOL	-	-	R/W [1, 2]
1000	ErrorCode	WORD	603F	0	R [3]
1001	HomeOffset	DWORD	607C	0	R/W [5, 6]
1002	HomingMethod	WORD	6098	0	R/W [3, 4]
1003	SpeedSearchSwitch	DWORD	6099	1	R/W [5, 6]
1004	SpeedSearchZero	DWORD	6099	2	R/W [5, 6]
1005	HomingAcceleration	DWORD	609A	0	R/W [5, 6]
1006	PositiveTorqueLimit	WORD	60E0	0	R/W [3, 4]
1007	NegativeTorqueLimit	WORD	0x60E1	0	R/W [3, 4]
1008	MotorRatedTorque	DWORD	0x6076	0	R/W [5, 6]
1009	FollowingErrorWindow	DWORD	0x6065	0	R/W [5, 6]
1010	FollowingErrorTimeOut	WORD	0x6066	0	R/W [3, 4]
1011	PositionWindow	DWORD	0x6067	0	R/W [5, 6]
1012	PositionTime	WORD	0x6068	0	R/W [3, 4]
1013	Min Position Limit	DWORD	0x607D	1	R/W [5, 6]
1014	Max Position Limit	DWORD	0x607D	2	R/W [5, 6]
1015	Digital outputs/ physical outputs	DWORD	0x60FE	1	R/W [5, 6]
1016	Digital outputs/ mask	DWORD	0x60FE	2	R/W [5, 6]

1) Access via *Chapter 6.3.21 'FB 825 - MC_ReadBoolParameter - read axis boolean parameter data' on page 164*

2) Access via *Chapter 6.3.22 'FB 826 - MC_WriteBoolParameter - write axis boolean parameter data' on page 166*

3) Access via *Chapter 6.3.25 'FB 829 - VMC_ReadWordParameter - read axis word parameter data' on page 172*

4) Access via *Chapter 6.3.26 'FB 830 - VMC_WriteWordParameter - write axis word parameter data' on page 174*

5) Access via *Chapter 6.3.23 'FB 827 - VMC_ReadDWordParameter - read axis double word parameter data' on page 168*

6) Access via *Chapter 6.3.24 'FB 828 - VMC_WriteDWordParameter - write axis double word parameter data' on page 170*

No.	Name	Data type	Index	Subindex	Access
1017	Quick stop deceleration	DWORD	0x6085	0	R/W [5, 6]
1018	Forward external torque limit	WORD	0x2404	0	R/W [3, 4]
1019	Reverse external torque limit	WORD	0x2405	0	R/W [3, 4]

1) Access via *Chapter 6.3.21 'FB 825 - MC_ReadBoolParameter - read axis boolean parameter data' on page 164*

2) Access via *Chapter 6.3.22 'FB 826 - MC_WriteBoolParameter - write axis boolean parameter data' on page 166*

3) Access via *Chapter 6.3.25 'FB 829 - VMC_ReadWordParameter - read axis word parameter data' on page 172*

4) Access via *Chapter 6.3.26 'FB 830 - VMC_WriteWordParameter - write axis word parameter data' on page 174*

5) Access via *Chapter 6.3.23 'FB 827 - VMC_ReadDWordParameter - read axis double word parameter data' on page 168*

6) Access via *Chapter 6.3.24 'FB 828 - VMC_WriteDWordParameter - write axis double word parameter data' on page 170*

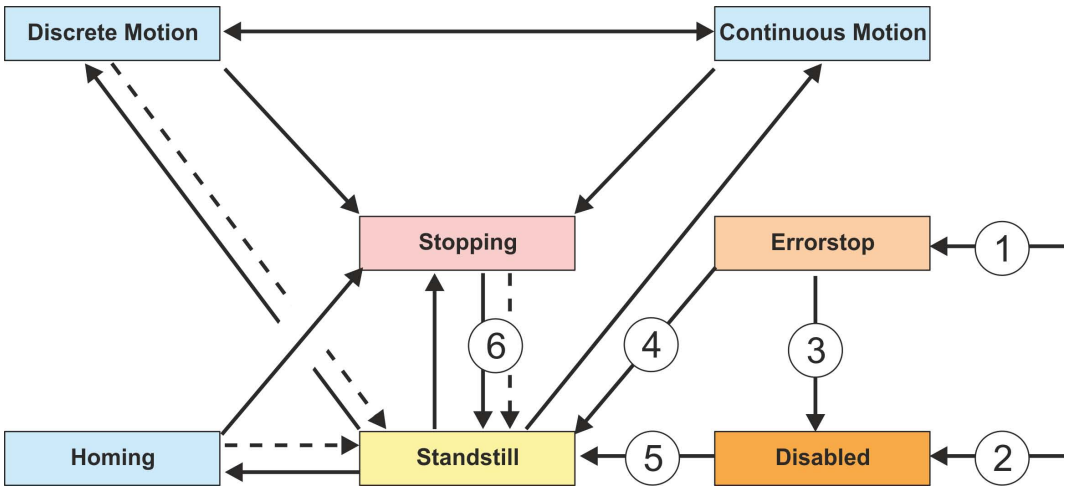
7 States and behavior of the outputs

7.1 States

State diagram

The *state diagram* shows all the states that an axis can assume. An axis is always in one of these states. Depending on the output state, a state change can take place automatically or via the blocks of the axis control. In principle, movement tasks are processed sequentially. You can use the following function blocks to query the state

- *Chapter 6.3.11 'FB 812 - MC_ReadStatus - PLCopen status' on page 147*
- Parameter *PLCopenState* from *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*



- - - ▶ Return when done
(1) From each state: An error has occurred at the axis
(2) From each state: MC_Power.Enable = FALSE and there is no error on the axis
(3) MC_Reset and MC_Power.Status = FALSE
(4) MC_Reset and MC_Power.Status = TRUE and MC_Power.Enable = TRUE
(5) MC_Power.Enable = TRUE and MC_Power.Status = TRUE
(6) MC_Stop.Done = TRUE and MC_Stop.Execute = FALSE

There are the following states

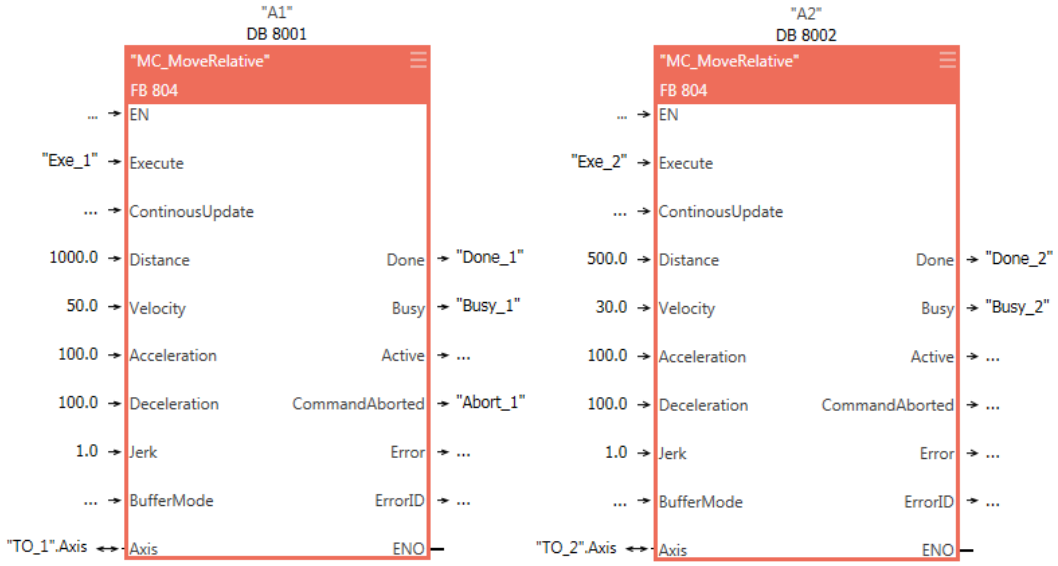
- Disabled
  - Basic state of an axis.
  - Axis can not be moved by any function block.
- Error Stop
  - An error has occurred on the axis.
  - Axis is stopped and is blocked for further motion tasks.
  - Axis remains in this state until the error is solved and a RESET is triggered.
  - Errors on an axis are also reported via the corresponding function block.
  - Errors on a function block do not lead to this state
- Stand Still
  - Ready for motion tasks
  - There is no error on the axis
  - There are no motion tasks active on the axis
  - Axis is power supplied
- Stopping
  - Axis is currently stopped:
    Chapter 6.3.5 'FB 802 - MC_Stop - stop axis' on page 135
    Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126
  - The *Stopping* state is active as long as a Stop command is active (*Execute* = 1). Even if the axis is already stopped. Then the state automatically changes to *Standstill*.

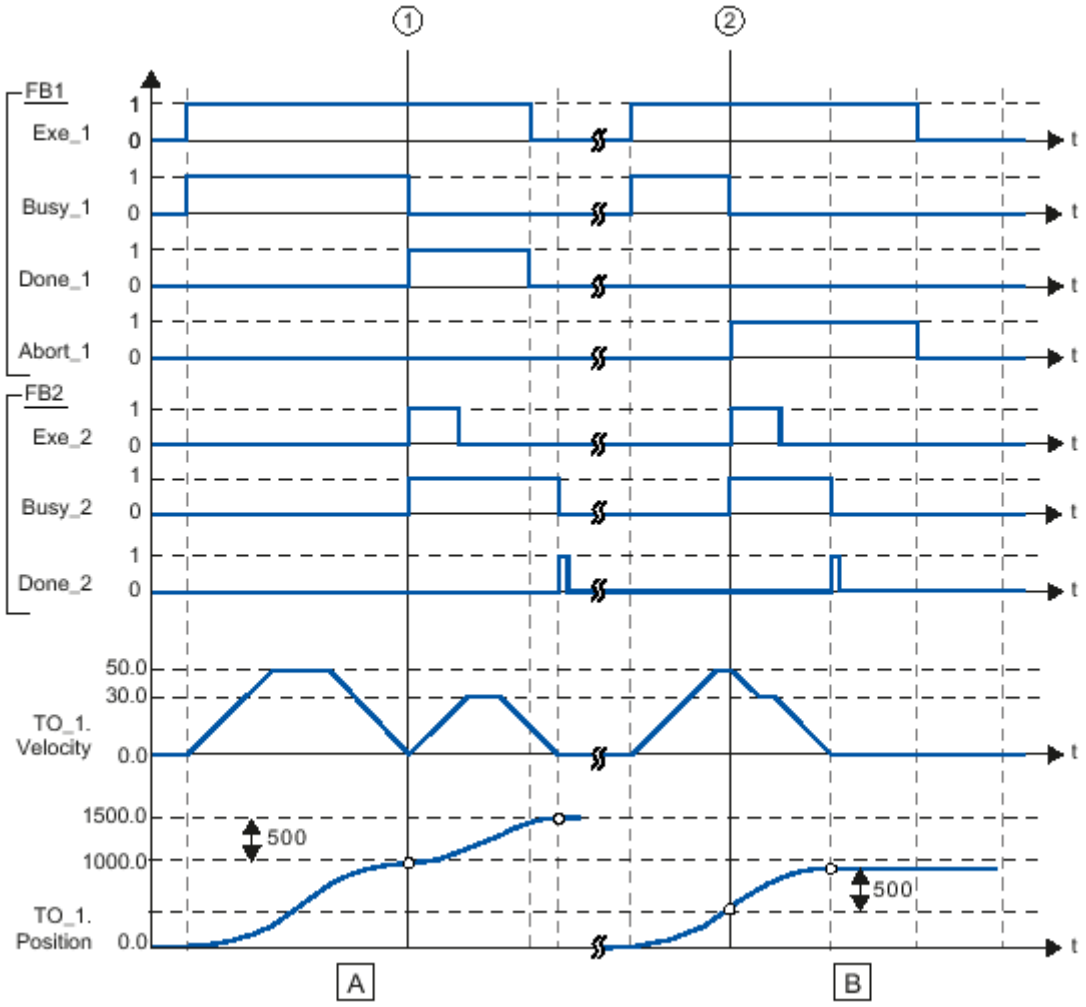
- Homing
  - The axis is currently homing:
    ↳ *Chapter 6.3.4 'FB 801 - MC_Home - home axis' on page 133*
    ↳ *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*
  - As soon as the axis is homed, the state automatically changes to *Standstill*.
- Discrete Motion
  - The axis is currently executing a motion task:
    ↳ *Chapter 6.3.9 'FB 808 - MC_MoveAbsolute - move axis to absolute position' on page 143*
    ↳ *Chapter 6.3.7 'FB 804 - MC_MoveRelative - move axis relative' on page 139*
    ↳ *Chapter 6.3.6 'FB 803 - MC_Halt - holding axis' on page 137*
    ↳ *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*
  - As soon as the target of the movement task is reached, the state automatically changes to *Standstill*.
- Continuous Motion
  - The axis performs a permanent movement task:
    ↳ *Chapter 6.3.8 'FB 805 - MC_MoveVelocity - drive axis with constant velocity' on page 141*
    ↳ *Chapter 6.2.2 'FB 860 VMC_AxisControl - Control block axis control' on page 126*

7.2 Replacement behavior of motion jobs

Example

In the following with an example of MC_MoveRelative the replacement behavior of motion jobs is explained. ↳ *Chapter 6.3.7 'FB 804 - MC_MoveRelative - move axis relative' on page 139*





(A) The axis is moved by the "MC_MoveRelative" job (A1) by the *Distance* 1000.0 (starting position is the position 0.0).

(1) Reaching the target position is reported at the time (1) *Done_1*. At this time (1) a further MC_MoveRelative order (A2) is started with the route 500.0. The successful achievement of the new target position is reported via *Done_2*. Since *Exe_2* was reset before, *Done_2* is only set for one cycle

(B) A running MC_MoveRelative job (A1) is replaced by a further MC_MoveRelative job (A2).

(2) The abort is reported at time (2) via *Abort_1*. The axis is then moved with the new velocity by the new distance *Distance* 500.0. The successful achievement of the new target position is reported via *Done_2*.

7.3 Behavior of the inputs and outputs

Exclusivity of the outputs

- The outputs *Busy*, *Done*, *Error* and *CommandAborted* exclude each other, so at a function block only one of these outputs can be TRUE at a time.
- As soon as the input *Execute* is TRUE, one of the outputs must be TRUE. Only one of the outputs *Active*, *Error*, *Done* and *CommandAborted* can be TRUE at one time.

Output status

- The outputs *Done*, *InVelocity*, *Error*, *ErrorID* and *CommandAborted* are reset with an edge 1-0 at the *Execute* input if the function block is not active (*Busy* = FALSE).
- The command execution is not affected by an edge 1-0 of *Execute*.
- If *Execute* is already reset during command execution, so it is guaranteed that one of the outputs is set at the end of the command for a PLC cycle. Only then the outputs are reset.

Input parameter

- The input parameters are taken with edge 0-1 at *Execute*.
- To change the parameters the command must be retriggered.
- If an input parameter is not passed to the function block, the last transferred value to this block remains valid.
- With the first call a sensible default value must be passed.

Position an distance

- The input *Position* designates an absolute position value.
- *Distance* designates a relative measure as distance between two positions.
- Both *Position* and *Distance* are preset in technical units e.g. [mm] or [°], in accordance to the scaling of the axis.

Parameter for the dynamic behavior

- The dynamic parameter for *Move* functions are preset in engineering units with second as the time base.
  If an axis is scaled in millimetres so the units are for *Velocity* [mm/s], *Acceleration* [mm/s^2], and *Deceleration* [mm/s^2].

Error handling

- All the function blocks have two fault outputs to indicate errors during command execution.
- *Error* indicates the error and *ErrorID* shows an additional error number.
- The outputs *Done* und*InVelocity* designate a successful command execution and are not set if *Error* becomes TRUE.

Error types

- Function block errors
  - Function block errors are errors that only concerns the function block and not the axis such as e.g. incorrect parameters.
  - Function block errors need not be explicitly reset , but will automatically reset when the input *Execute* is reset.
- Communication errors
  - Communication error such as e.g. the function block can not address the axis.
  - Communication errors often indicate an incorrect configuration or parametrization.
  - A reset is not possible, but the function block can be retriggered after the configuration has been corrected.
- Axis errors
  - Axis errors usually occur during the move such as e.g. position error.
  - An axis error must be reset by MC_Reset.

Behavior of the *Done* output

- The *Done* output is set, when a command was successfully executed.
- When operating with multiple function blocks at one axis and the current command is interrupted by another block, the *Done* output of the first block is not set.

Behavior of the *CommandAborted* output

- *CommandAborted* is set when a command is interrupted by another block.

Behavior of the *Busy* output

- The *Busy* output indicates that the function block is active.
- *Busy* is immediately set with edge 0-1 of *Execute* and will not be reset until the command was completed successfully or failed.
- As long as *Busy* is TRUE, the function block must be called cyclically to execute the command.

Behavior of the *Active* output

- If the motion of an axis is controlled by several function blocks, the *Active* output of each block indicates that the command is executed by the axis.

***Enable*-Input and *Valid* output**

- In contrast to *Execute* the *Enable* input causes that an action is permanently and continuously executed, as long as *Enable* is TRUE. MC_ReadStatus e.g. cyclically refreshes for example the status of an axis as long as *Enable* is TRUE.
- A function block with a *Enable* input indicates by the *Valid* output that the data of the outputs are valid. However, the data can constantly be updated during *Valid* is TRUE.

BufferMode

- *BufferMode* is not supported.

8 ErrorID - Additional error information

ErrorID	Description	Remark
0x0000	No Error	
0x8001	Invalid value at parameter *Position*.	
0x8002	Invalid value at parameter *Distance*.	
0x8003	Invalid value at parameter *Distance*.	
0x8004	Invalid value at parameter *Acceleration*.	
0x8005	Invalid value at parameter *Deceleration*.	
0x8007	Invalid value at parameter *ContinuousUpdate*.	
0x8008	Invalid value at parameter *BufferMode*.	
0x8009	Invalid value at parameter *EnablePositive*.	
0x800A	Invalid value at parameter *EnableNegative*.	
0x800B	Invalid value at parameter *MasterOffset*.	
0x800C	Invalid value at parameter *SlaveOffset*.	
0x800D	Invalid value at parameter *MasterScaling*.	
0x800E	Invalid value at parameter *SlaveScaling*.	
0x800F	Invalid value at parameter *StartMode*.	
0x8010	Invalid value at parameter *ActivationMode*.	
0x8011	Invalid value at parameter *Source*.	
0x8012	Invalid value at parameter *Direction*.	
0x8013	Invalid parameter of virtual axis.	Mc_ReadParameter
0x8014	Invalid parameter of physical axis.	Mc_ReadParameter
0x8015	Invalid index or subindex.	Mc_ReadParameter
0x8016	Invalid parameter length.	Mc_ReadParameter
0x8017	Invalid LADDR.	Mc_ReadParameter
0x8018	Invalid value at parameter *RatioDenominator*.	MC_GearIn
0x8019	Invalid value at parameter *RatioNumerator*.	MC_GearIn
0x801A	Unknown parameter number.	Mc_ReadParameter, MC_Write-Parameter
0x801B	Parameter can not be written, parameter is write protected	MC_WriteParameter
0x801C	Parameter communication with unknown mode.	MC_Home, MC_WriteParameter
0x801D	Parameter communication with general error. The cause of the error is not described in detail.	
0x801E	SDO parameter value out of range.	MC_Home, MC_WriteParameter
0x801F	The Type in ANY is not BYTE.	Read/write parameter
0x8020	Different configuration of the user units in cam and master axis.	
0x8021	Different configuration of the user units in cam and slave axis.	

ErrorID	Description	Remark
0x8022	There is no PROFIBUS/PROFINET device at the logical address specified in LADDR, from which you can read consistent data.	Read/write parameter
0x8023	An access error has been detected when accessing an I/O device.	Read/write parameter
0x8024	Slave error at external DP slave.	Read/write parameter
0x8025	System error at external DP slave.	Read/write parameter
0x8026	System error at external DP slave.	Read/write parameter
0x8027	The data haven't yet been read by the module.	Read/write parameter
0x8028	System error at external DP slave.	Read/write parameter
0x8029	Attempt to write a read only object.	Read/write parameter
0x802A	Attempt to read a write only object.	Read/write parameter
0x802B	Unsupported access to an object.	Read/write parameter
0x802C	Wrong data type	Read/write parameter
0x802D	Error in device profile.	Read/write parameter
0x802E	Error command type	Read/write parameter
0x802F	No system resources available.	Read/write parameter
0x8101	No cyclic communication with axis possible.	
0x8102	Command is in current PLCopen-State not allowed.	
0x8103	Command is not supported by the axis.	
0x8104	Axis is not ready to switch on, possible reasons: ■ Communication to the axis is not ready. ■ Drive is not in status *'switched on'* → Drive error possibly reset with MC_Reset ■ Communication was interrupted, e.g. by CPU power cycle. Reset error with MC_Reset.	*PreOperational* has also to be set in *Operational*.
0x8105	Command is not supported by virtual axis.	
0x8106	PLCopen-State is not defined.	
0x8201	Command cannot be executed temporarily because of lack of internal resources (no free slot in CommandBuffer).	
0x8202	Error when writing the offset for homing (no free slot in the CommandBuffer).	DriveManager → Homing (active command)
0x8301	No cyclic communication with master axis possible.	
0x8302	Command is in current PLCopen-State of the master axis not allowed.	
0x8303	Command is not supported by the master axis.	
0x8304	Master axis is not in status *Pre-Operational*.	
0x8305	Master axis data block number has been changed.	
0x8306	Communication errors at the master axis. Slave axis is stopped with fast stop.	
0x8311	No cyclic communication with slave axis possible.	

ErrorID	Description	Remark
0x8312	Command is in current PLCopen-State of the slave axis not allowed.	
0x8313	Command is not supported by the slave axis.	
0x8314	Slave axis is not in status *Pre-Operational*.	
0x8315	Slave axis data block number has been changed.	
0x8321	Coupling with *StartMode* = relative and *ActivationMode* = nextcycle is not permitted	
0x8322	Coupling or switching with *StartMode* = absolute and *Activation-Mode* = nextcycle is not permitted	
0x8323	Switching with a different *StartMode* (*StartMode* of the coupling is to be used).	
0x8331	MC_CamIn is not active.	
0x8332	MC_GearIn is not active.	
0x8340	Invalid value at TriggerInput.Probe.	
0x8341	Invalid value at TriggerInput.Source.	
0x8342	Invalid value at TriggerInput.TriggerMode.	
0x8350	Invalid value at VelocitySearchSwitch.	Homing, initialization
0x8351	Invalid value at VelocitySearchZero.	Homing, initialization
0x8352	Invalid combination of inputs.	Homing, initialization
0x8400	MC_Power: Unexpected Drive-State Drive-State <> Operation enabled	
0x8401	MC_Power: Unexpected Drive-State Drive-State = Quick stop active	
0x8402	MC_Power: Unexpected Drive-State Drive-State = Fault reaction active	
0x8403	MC_Power: Unexpected Drive-State Drive-State = Fault	
0x8500	Wrong value in *EncoderType* (1 or 2)	Init block
0x8501	Wrong value in *EncoderResolutionBits* (> 0 and ≤32)	Init block
0x8502	Incorrect value in *LogicalAddress* (≥0)	Init block
0x8503	Incorrect value in *StartInputAddress* (≥0)	Init block
0x8504	Wrong value in *StartOutputAddress* (≥0)	Init block
0x8505	Wrong value in *FactorPosition* (>0.0)	Init block
0x8506	Wrong value in *FactorVelocity* (>0.0)	Init block
0x8507	Wrong value in *FactorAcceleration* (>0.0)	Init block
0x8508	Wrong value in *MaxVelocityApp* (>0.0)	Init block
0x8509	Wrong value in *MaxAccelerationApp* (>0.0)	Init block

ErrorID	Description	Remark
0x850A	Wrong value in *MaxDecelerationApp* (>0.0)	Init block
0x850B	Wrong value in *MaxVelocityDrive* (>0.0)	Init block
0x850C	Wrong value in *MaxAccelerationDrive* (>0.0)	Init block
0x850D	Wrong value in *MaxDecelerationDrive* (>0.0)	Init block
0x850E	Wrong value in *MinPosition* (≥MinUserPos)	Init block
0x850F	Wrong value in *MaxPosition* (≥MaxUserPos)	Init block
0x8603	Error homing at the drive, speed <> 0.	MC_Home
0x8604	Error homing at the drive, speed = 0.	MC_Home