

SIEMENS



TIA Portal Programming

Train The Trainer 2021 – 2022
TIA Basic Course



SIMATIC S7

TIA Portal Programming - Basic Course

Name: _____

Course from: _____ to: _____

Instructor: _____

Location: _____

1 Firmware Update SIMATIC S7-1500

2 Unspecified Hardware Configuration with SIMATIC S7-1500

3 Decentral Hardware configuration with SIMATIC S7-1500 and ET 200SP via PROFINET

4 Basics of FC Programming with SIMATIC S7-1500

5 Basics of FB Programming with SIMATIC S7-1500

6 IEC Timers and IEC Counters Multi-instances for SIMATIC S7-1500

7 Basics of Diagnostics with SIMATIC S7-1500

8 Diagnostics via the Web with SIMATIC S7-1500

9 Analog Values for SIMATIC S7-1500

10 Global Data Blocks for the SIMATIC S7-1500

11 WinCC Advanced with TP700 Comfort and SIMATIC S7-1500

12 PID Controller for SIMATIC S7-1200/1500

13 Frequency converter G120 on PROFINET with SIMATIC S7-1500

14

15

16



Training Curriculum

TIA Portal Module 001
Firmware Update SIMATIC S7-1500

Table of contents

1	Goal	4
2	Requirement	4
3	Required hardware and software	4
4	Theory	5
4.1	SIMATIC S7-1500 automation system.....	5
4.2	Operator control and display elements of the CPU 1516F-3 PN/DP	6
4.2.1	Front view of the CPU 1516F-3 PN/DP with integrated display	6
4.2.2	Status and error displays.....	6
4.2.3	Operator control and connection elements of the CPU 1516F-3 PN/DP behind the front flap	7
4.2.4	SIMATIC Memory Card (MC)	8
4.2.5	Mode selector	8
4.3	SIMATIC STEP 7 Professional V1X (TIA Portal V1X) programming software.....	9
4.3.1	Basic settings for the TIA Portal	9
4.3.2	Setting the IP address on the programming device.....	10
4.3.3	Setting the IP address in the CPU.....	12
	Formatting the memory card in the CPU.....	14
4.3.4	Restoring the factory settings of the CPU	15
	Downloading a firmware update from the SIEMENS Support website	16
	Firmware update of the CPU	22
4.3.5	Firmware update of the display	24

Firmware Update – SIMATIC S7-1500

1 Goal

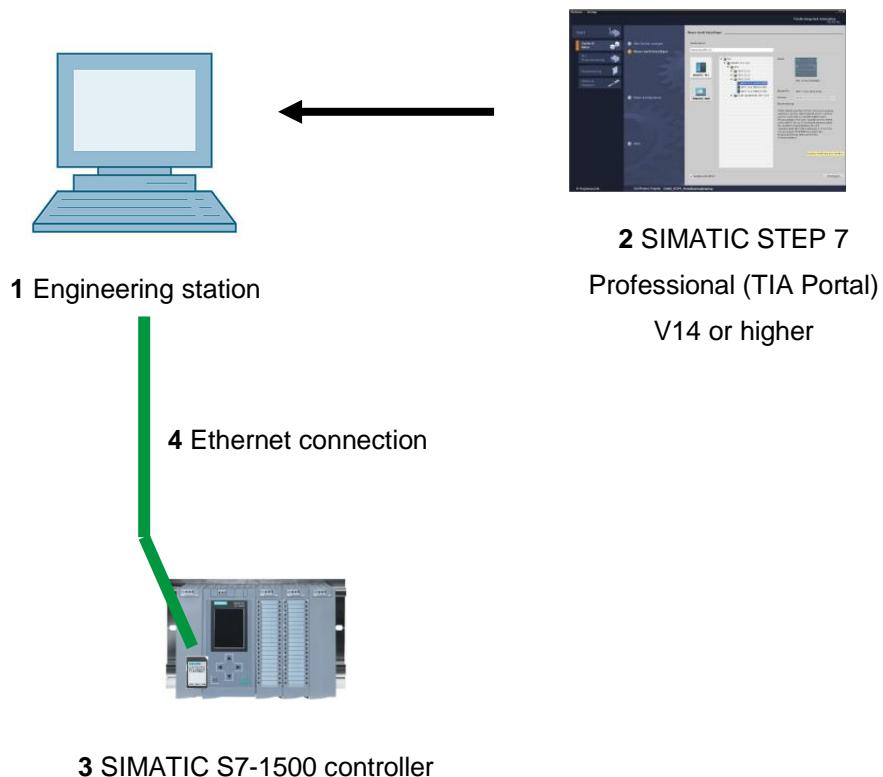
This chapter will show how the **firmware version of the CPU** of a **SIMATIC S7-1500** can be checked and upgraded using the TIA Portal.

2 Requirement

No prerequisites have to be met for successful completion of this chapter.

3 Required hardware and software

- 1 Engineering station: requirements for hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVD)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – V1X or higher
- 3 SIMATIC S7-1500 controller, e.g. CPU 1516F-3 PN/DP – Firmware V1.6 or higher with memory card
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 SIMATIC S7-1500 automation system

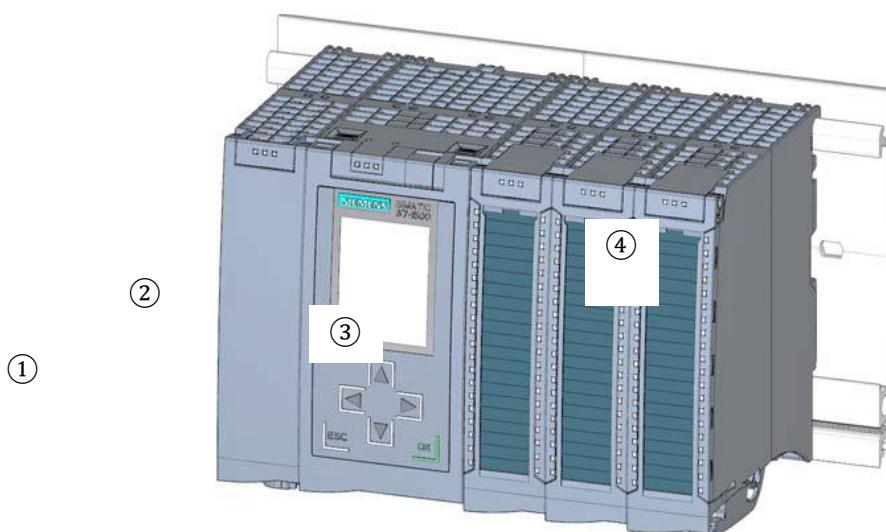
The SIMATIC S7-1500 automation system is a modular controller system for the middle to upper performance range.

A comprehensive range of modules is available to optimally adapt the system to the automation task.

SIMATIC S7-1500 is the next generation of the SIMATIC S7-300 and S7-400 automation systems with the following new performance features:

- Increased system performance
- Integrated motion control functionality
- PROFINET IO IRT
- Integrated display for machine-level operation and diagnostics
- STEP 7 language innovations while maintaining proven functions

The S7-1500 controller consists of a power supply ①, a CPU with integrated display ② and input and output modules for digital and analog signals ③. The modules are mounted on a mounting rail with integrated DIN rail profile ④. If necessary, communications processors and function modules for special tasks such as stepper motor control are also used.



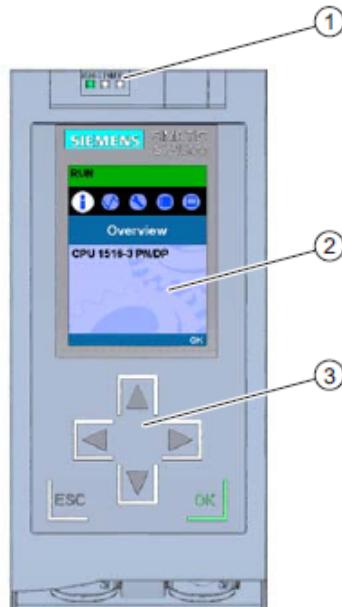
The system is programmed with the STEP 7 Professional software.

4.2 Operator control and display elements of the CPU 1516F-3 PN/DP

The figure below shows the operator control and display elements of a CPU 1516F-3 PN/DP.

The arrangement and number of elements differ from this figure for other CPUs.

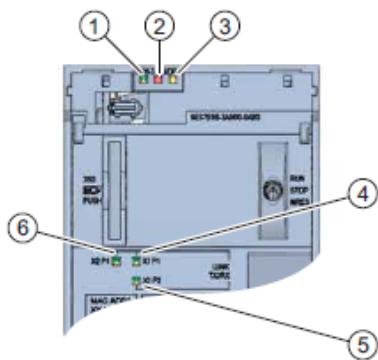
4.2.1 Front view of the CPU 1516F-3 PN/DP with integrated display



- 1) LED displays for the current operating mode and diagnostic status of the CPU
- 2) Display
- 3) Control keys

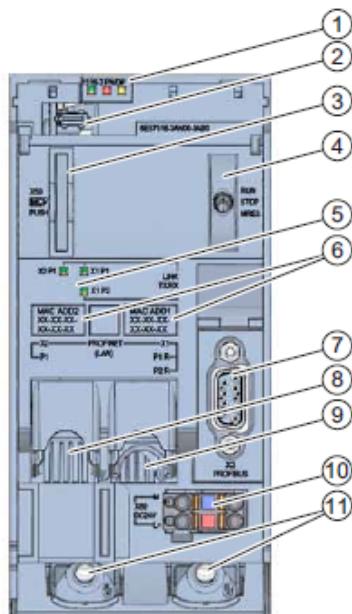
4.2.2 Status and error displays

The CPU comes with the following LED displays:



- 1) RUN/STOP LED (yellow/green LED)
- 2) ERROR LED (red LED)
- 3) MAINT LED (yellow LED)
- 4) LINK RX/TX LED for port X1 P1 (yellow/green LED)
- 5) LINK RX/TX LED for port X1 P2 (yellow/green LED)
- 6) LINK RX/TX LED for port X1 P1 (yellow/green LED)

4.2.3 Operator control and connection elements of the CPU 1516F-3 PN/DP behind the front flap



- 1) LED displays for the current operating mode and diagnostic status of the CPU
- 2) Display connection
- 3) Slot for the SIMATIC memory card
- 4) Mode selector
- 5) LED displays for the 3 ports of PROFINET interfaces X1 and X2
- 6) MAC addresses of the interfaces
- 7) Display connection
- 8) Slot for the SIMATIC Memory Card
- 9) Mode selector
- 10) LED displays for the 3 ports of PROFINET interfaces X1 and X2
- 11) MAC addresses of the interfaces

Note: The hinged front cover with the display can be removed and inserted during operation.

4.2.4 SIMATIC Memory Card (MC)

A SIMATIC Micro Memory Card (MC) is used as the memory module for the CPUs. This is a preformatted memory card that is compatible with the Windows file system. It is available with various storage capacities and can be used for the following purposes:

- Transportable data storage medium
- Program card
- Firmware update card

For operation of the CPU, the MMC **must** be inserted because the CPUs have no integrated load memory. A commercially available SD card reader is needed to write/read the SIMATIC memory card with the programming device or PC. This allows files to be copied directly to the SIMATIC memory card using Windows Explorer, for example.

Note: *It is recommended that the SIMATIC Memory Card only be removed or inserted when the CPU is in the POWER OFF state.*

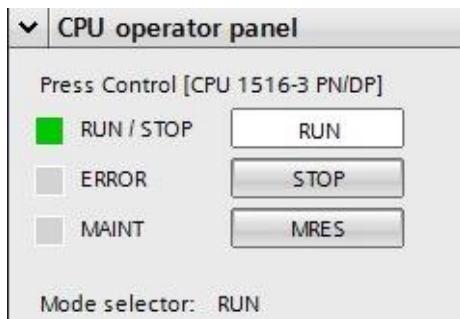
4.2.5 Mode selector

The mode selector allows you to set the operating mode of the CPU. The mode selector is designed as a toggle switch with three switch positions.

Position	Meaning	Explanation
RUN	RUN mode	The CPU is executing the user program
STOP	STOP mode	The CPU is not executing the user program
MRES	Memory reset	Position for the memory reset of the CPU

You can also use the button on the CPU operator panel of the SIMATIC STEP 7 Professional V1X software in Online & Diagnostics to switch the operating mode (**STOP** or **RUN**).

The operator panel also contains an **MRES** button for performing a memory reset and displays the status LEDs of the CPU.



4.3 SIMATIC STEP 7 Professional V1X (TIA Portal V1X) programming software

The SIMATIC STEP 7 Professional (TIA Portal) software is the programming tool for the following automation systems:

- SIMATIC S7-1500
- SIMATIC S7-1200
- SIMATIC S7-300
- SIMATIC S7-400
- SIMATIC WinAC

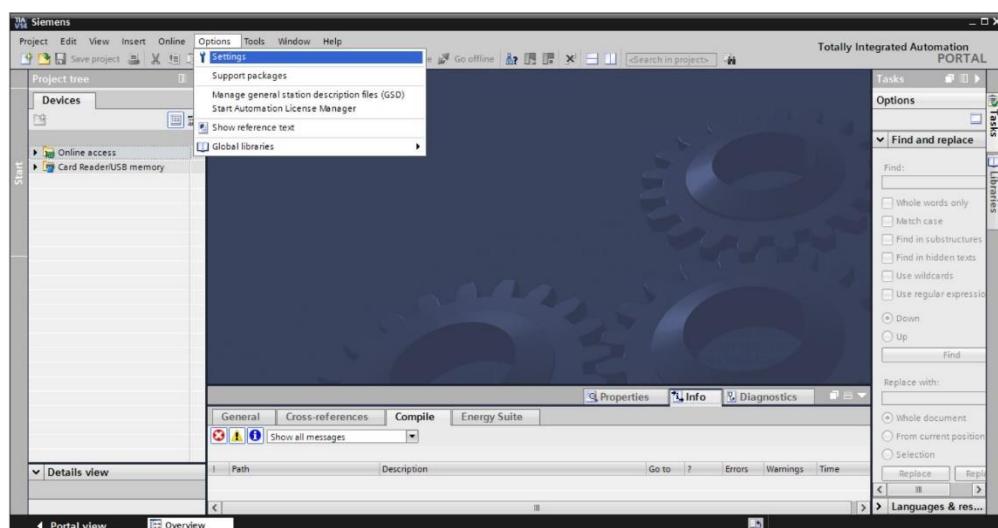
SIMATIC STEP 7 Professional provides the following functions for plant automation:

- Configuration and parameter assignment of the hardware
- Specification of the communication
- Programming
- Testing, commissioning and service with operational/diagnostic functions
- Documentation
- Creation of visualizations for SIMATIC Basic Panels with the integrated WinCC Basic software
- Visualization solutions for PCs and other panels can also be created with other WinCC software packages

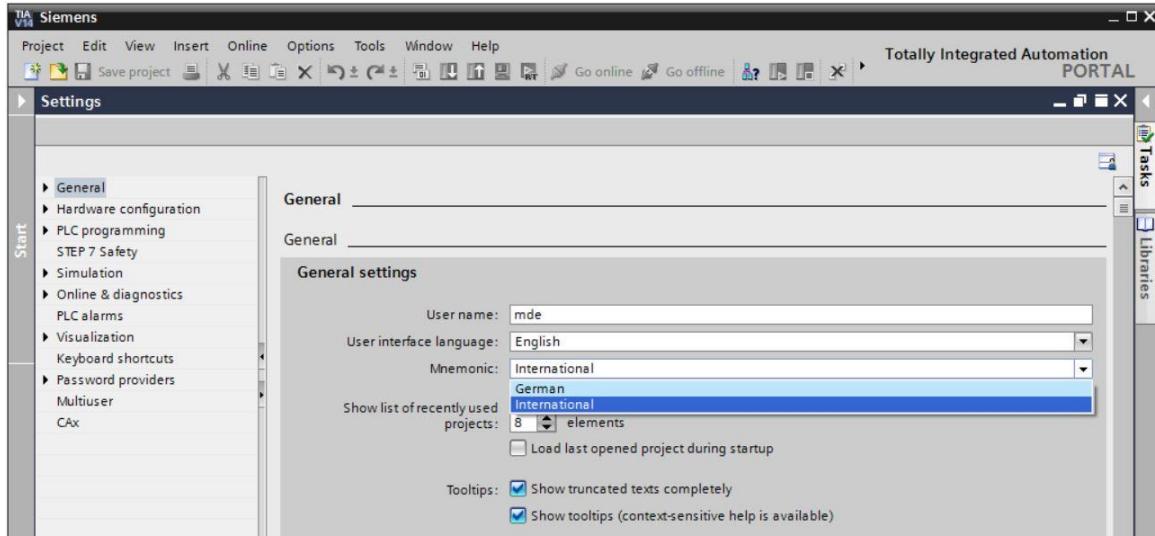
Support is provided for all functions through detailed online help.

4.3.1 Basic settings for the TIA Portal

- Users can specify their own default settings for certain settings in the TIA Portal. A few important settings are shown here.
- In the project view, select the → "Options" menu and then → "Settings".



- One basic setting is the selection of the user interface language and the language for the program display. In the curriculums to follow, "English" will be used for both settings.
- Under → "General" in "Settings", select "User interface language → English" and "Mnemonic → International".



Note: These settings can always be changed.

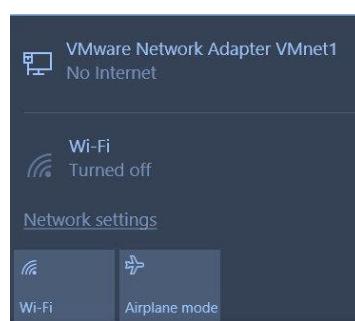
4.3.2 Setting the IP address on the programming device

You need a TCP/IP connection in order to upgrade the CPU of a SIMATIC S7-1500 controller from the PC, the programming device or a laptop.

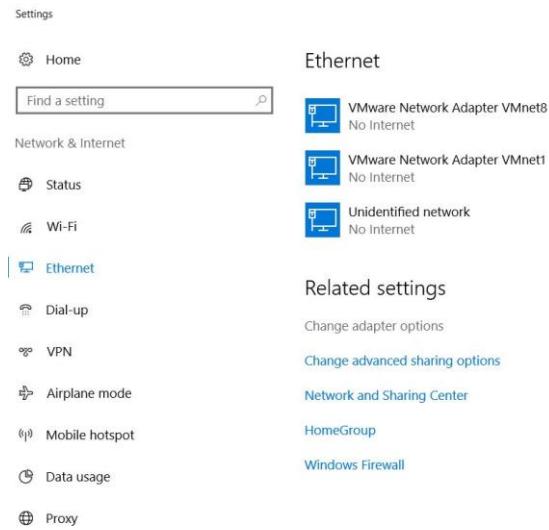
It is important that the IP addresses of both devices match for the computer and SIMATIC S7-1500 to communicate with each other via TCP/IP.

First, we will show you how to set the IP address of a computer with the Windows 10 operating system.

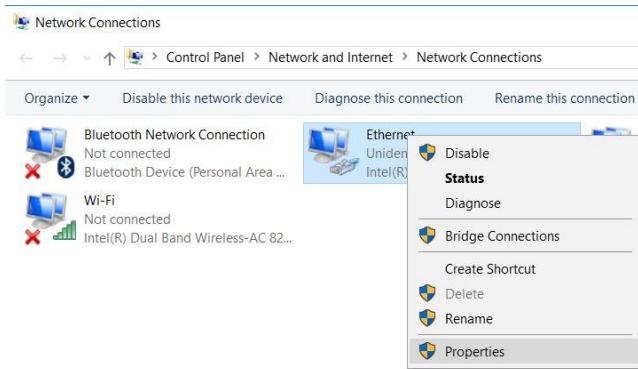
- Select the network icon in the taskbar at the bottom and click → "Network settings".



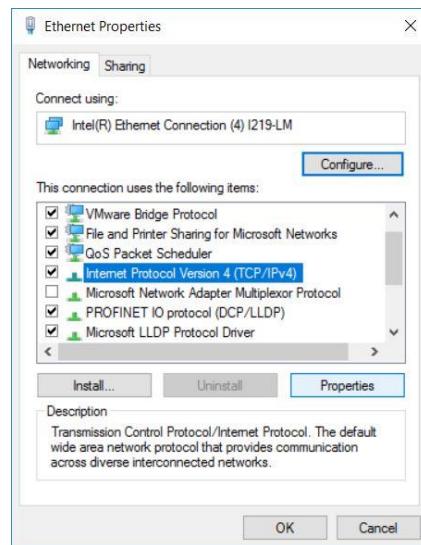
- In the network settings window that opens, click → "Ethernet" and then on → "Change adapter options".



- Select the desired → "LAN Connection" that you want to use to connect to the controller and click → "Properties".



- Select → "Properties" for → "Internet Protocol Version 4 (TCP/IPv4)".



- You can now use the following IP address, for example → IP address: 192.168.0.99 and enter the following → subnet mask 255.255.255.0. Accept the settings. (→ "OK")



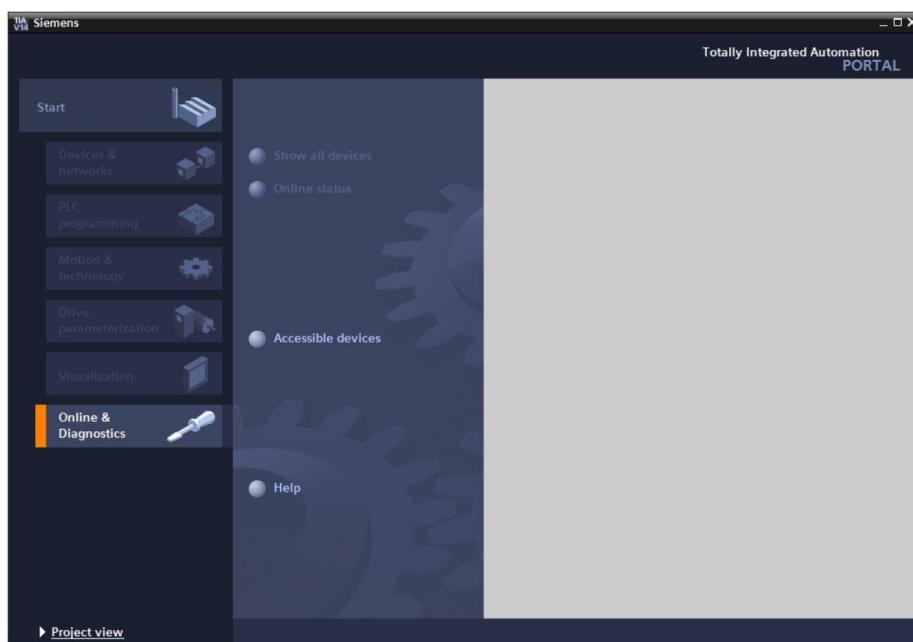
4.3.3 Setting the IP address in the CPU

Before a firmware update of the CPU can be performed, set the IP address of the SIMATIC S7-1500 correctly so that the programming device can reach the CPU via TCP (IP communication). The IP address of the SIMATIC S7-1500 is set as follows.

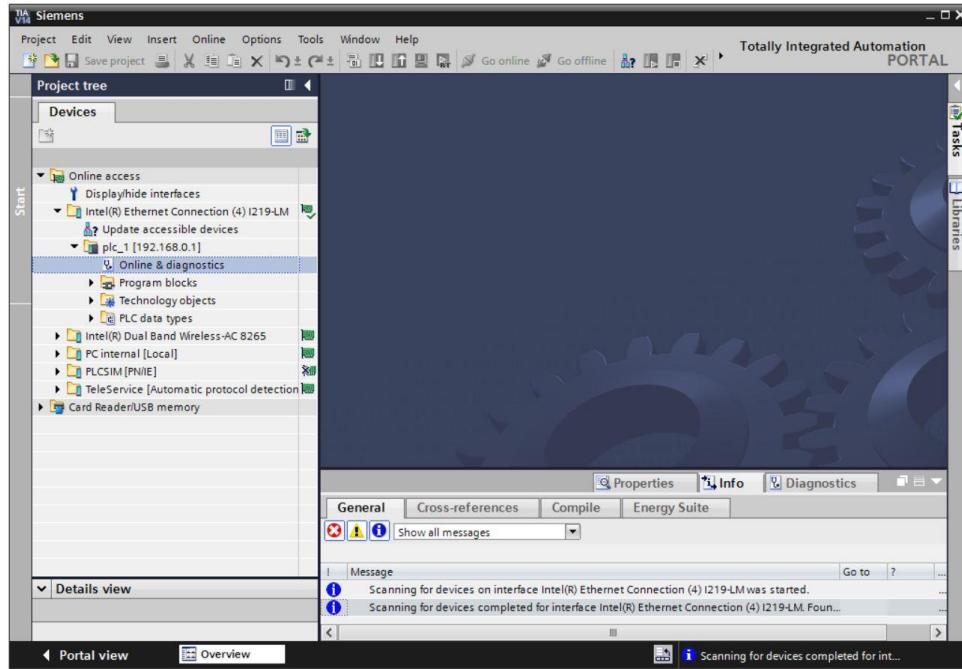
- Double-click the Totally Integrated Automation Portal to select it. (→ TIA Portal V1X)



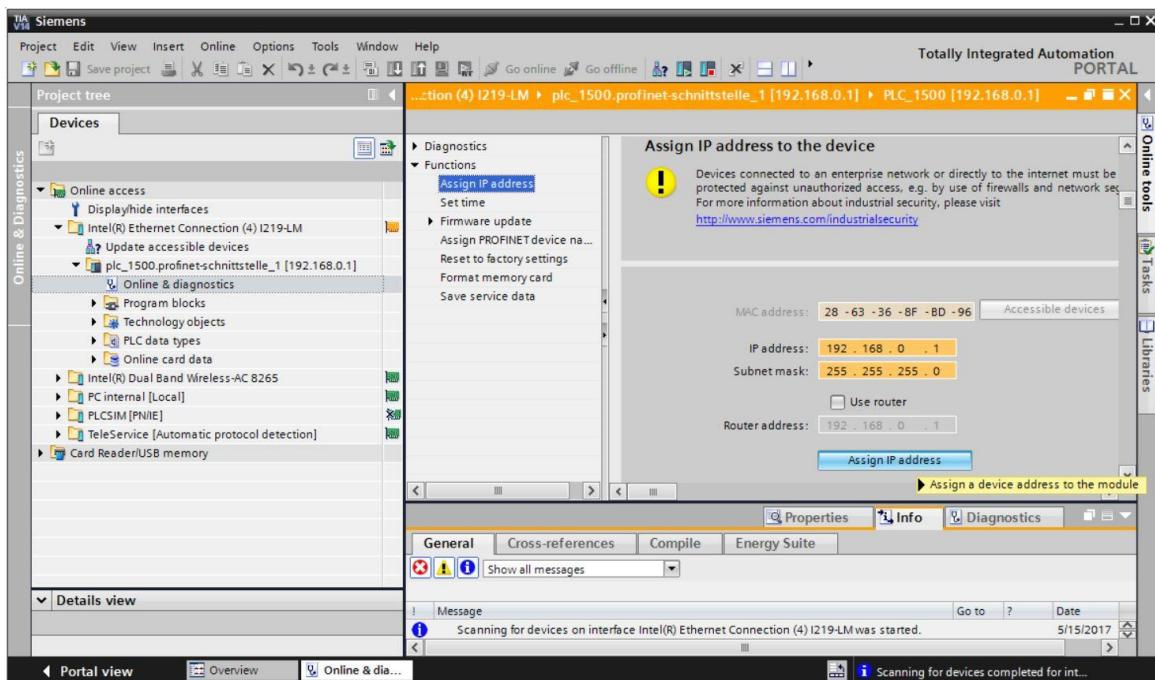
- Click → "Online & Diagnostics" and open → "Project view".



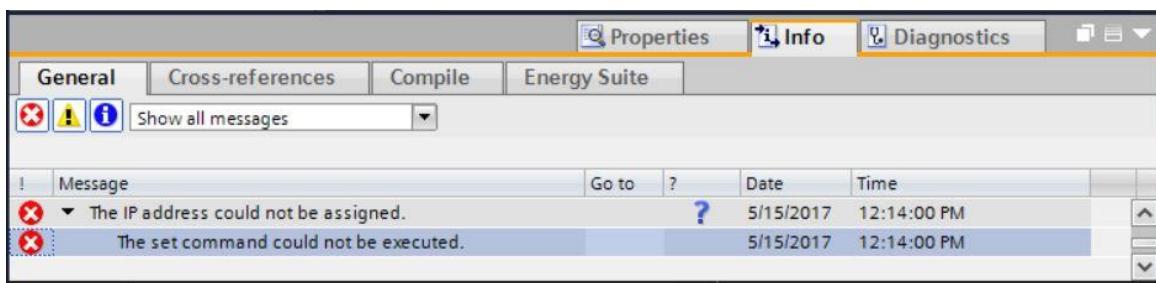
In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices", you will see the IP address (if previously set) or the MAC address (if IP address not yet assigned) of the connected SIMATIC S7-1500. Select → "Online & Diagnostics".



- Under → "Functions", you now find the → "Assign IP address" item. Enter the following IP address here (example): → IP address: 192.168.0.1 → Subnet mask 255.255.255.0. Next, click → "Assign IP address" and this new address will be assigned to your SIMATIC S7-1500.



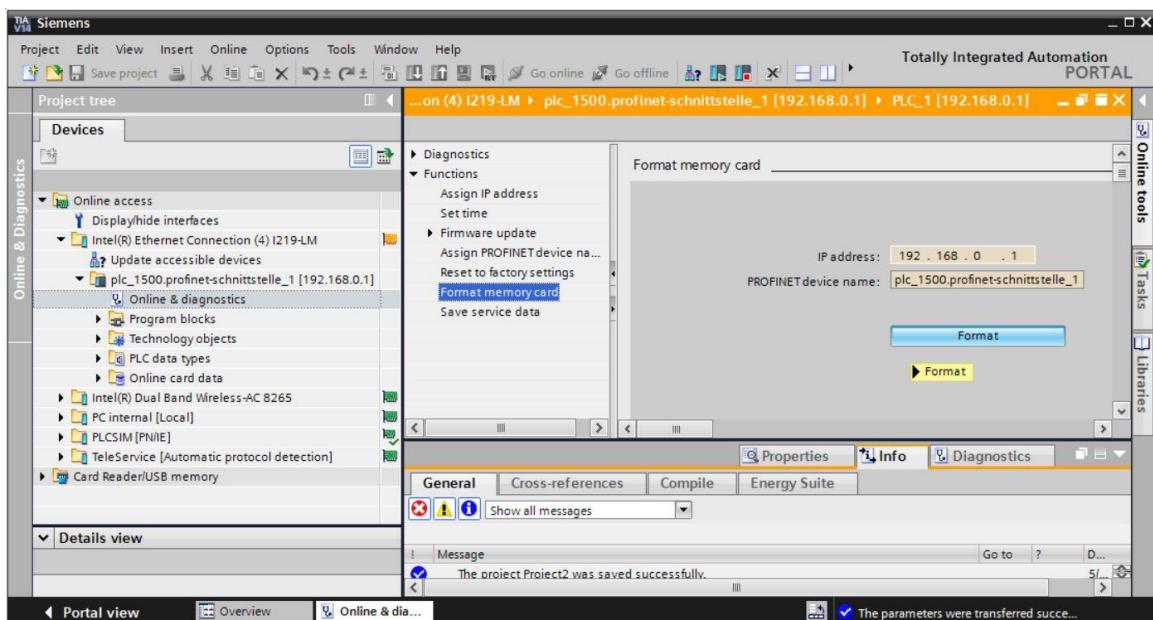
- If the IP address was not successfully assigned, you will receive a message in the → "Info" window under → "General".



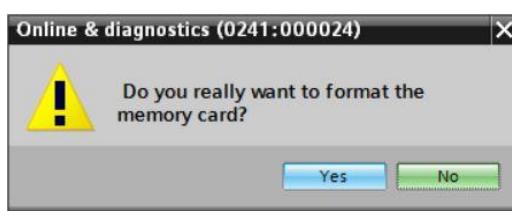
Formatting the memory card in the CPU

If the IP address could not be assigned, the program data on the CPU must be deleted. This is accomplished in two steps: → "Format memory card" and → "Reset to factory settings".

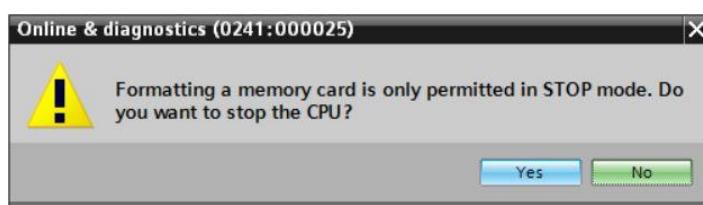
- First, select the → "Format memory card" function and press the → "Format" button.



- Confirm the prompt asking if you really want to format the memory card with → "Yes".

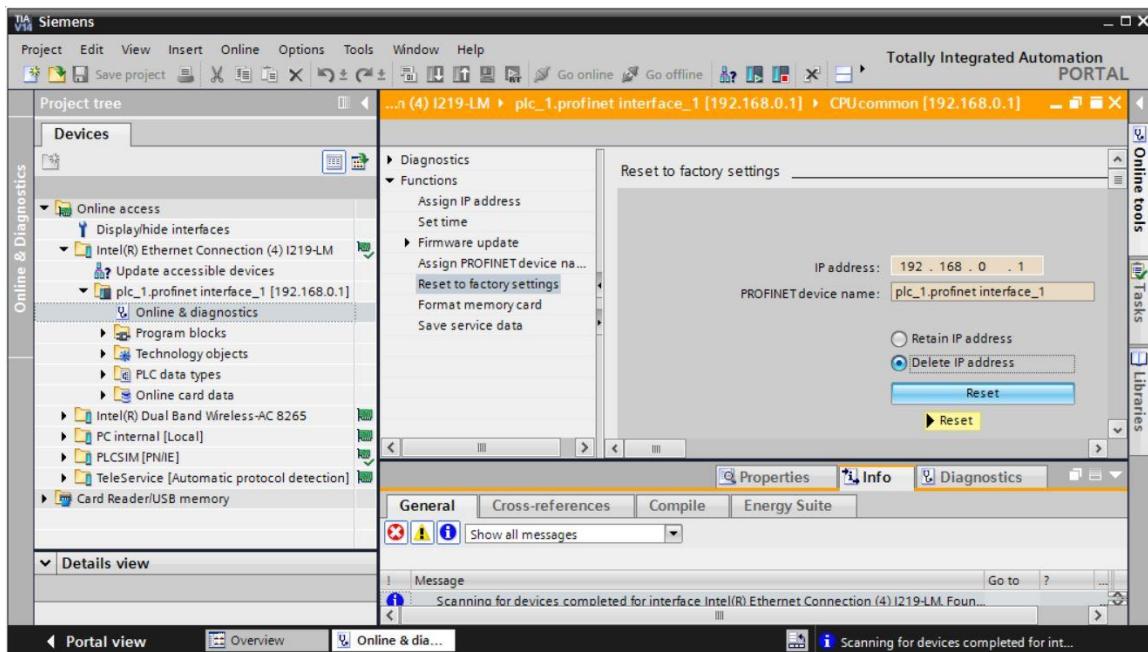


- If necessary, stop the CPU. (→ "Yes")

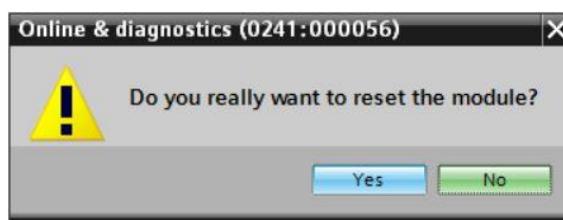


4.3.4 Restoring the factory settings of the CPU

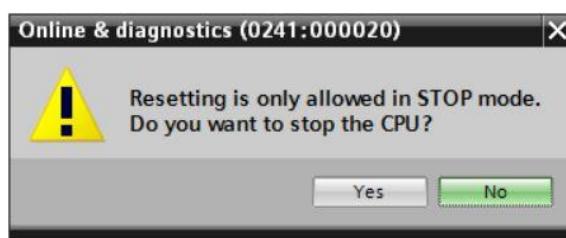
- Before you can reset the CPU, you must wait until the formatting of the CPU has finished. Afterwards, you must select → "Update accessible devices" and → "Online & Diagnostics" of your CPU again. To reset the controller, select the → "Reset to factory settings" function and then → "Delete IP address" and click → "Reset".



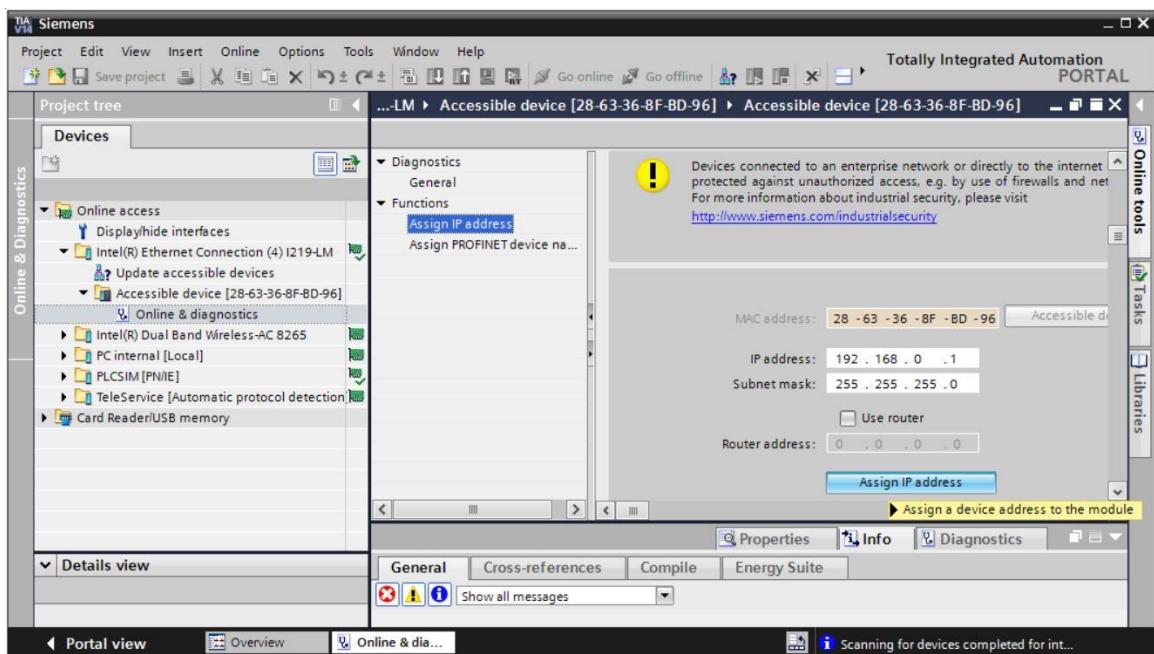
- Confirm the prompt asking if you really want to reset the module with → "Yes".



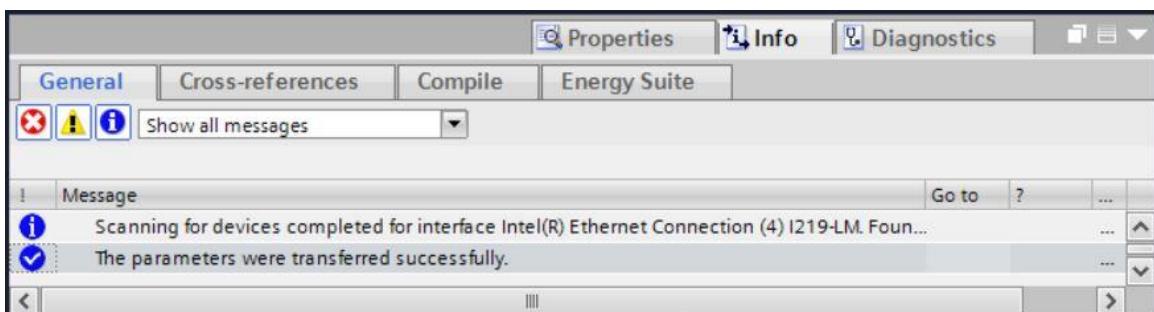
- If necessary, stop the CPU. (→ "Yes")



- Once the CPU has been reset, click → "Update accessible devices" again. The MAC address of the connected SIMATIC S7-1500 can now be seen. Select → "Assign IP address" under → "Functions" in → "Online & Diagnostics". Enter the following IP address here (example): IP address: 192.168.0.1 Subnet mask 255.255.255.0. Click "Assign IP address" and this new address will be assigned to your SIMATIC S7-1500.



- You will receive a message regarding successful transfer of parameters in the → "Info" → "General" window.



Downloading a firmware update from the SIEMENS Support website

You can download current firmware updates free of charge from the Industry Online Support of SIEMENS AG.

- Open your choice of Internet browser and enter the address → "support.automation.siemens.com".



- Select your desired language → "Language" → "English".

The screenshot shows the Industry Online Support International website. At the top, there is a navigation bar with links for Home, Contact, Help, Support Request, Site Explorer, and a search bar. A language dropdown menu is open, showing options for English, Deutsch, français, italiano, español, and 中文 (Chinese). Below the navigation, there is a section titled "articles" with three links: "i17 | Marine / Shipping, LR (Lloyds Register), LR register", "i17 | Getting Started SINAMICS S120 in Startdrive", and "i17 | Declaration of Conformity, EC/EU-Declaration of Conformity, Manufacturer". To the left, there is a sidebar with links for Home, Product Support, Application Examples, Services, Forum, and mySupport. The main content area features a search form with a placeholder "Product/Article No." and a search button. At the bottom, there is a footer with links for Imprint, Privacy policy, Cookie policy, Terms of use, and Digital ID.

In "Searching for product information", enter the CPU for which you need a firmware update. For example: → "S7-1500 CPU1516F"

The screenshot shows the Industry Online Support Germany website. The layout is similar to the international version, with a navigation bar, language dropdown (showing Deutsch selected), and a sidebar with links for Home, Product Support, Application Examples, Services, Forum, and mySupport. The main content area features a search form with a placeholder "S7-1500 CPU1516F" and a search button. The footer at the bottom includes links for Imprint, Privacy policy, Cookie policy, Terms of use, and Digital ID.

→ Under "Filter criteria for entries" select the "Entry type" → "Download" and click the entry with firmware updates for your CPU and the associated display in the selection list.

Filter criteria for entries

- All Products My Products
- Product tree
 - All
- Product
 - All
- Entry type
 - Download (18)
- Date
 - From _____ To _____

mySupport Cockpit

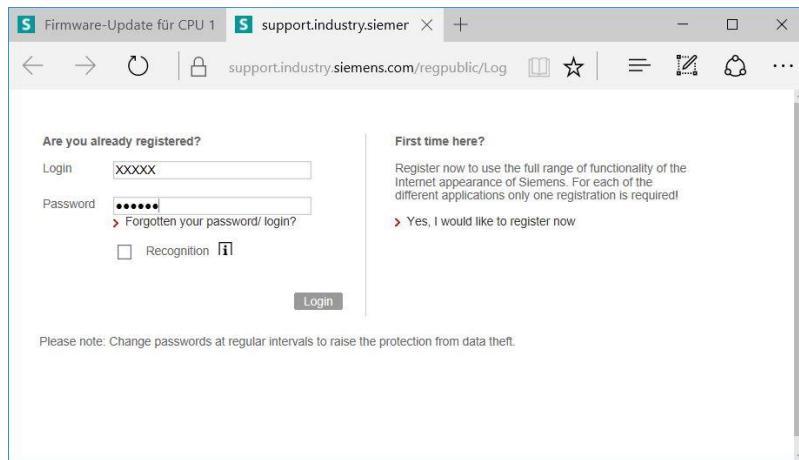
- > Favorites
- > Personal messages
- > My requests
- > CAx downloads
- > My Products / Clipboard
- > User online (444)

Various updates are offered in the next window. Select the update for your CPU that is recommended for the upgrade.

Article number	Software version	Upgrade by means of...
Latest versions of firmware for update		
6ES7510-1DJ01-0AB0 6ES7510-1SJ01-0AB0 6ES7511-1AK01-0AB0 6ES7511-1CK00-0AB0 6ES7511-1FK01-0AB0 6ES7511-1TK01-0AB0 6ES7512-1CK00-0AB0 6ES7512-1DK01-0AB0 6ES7512-1SK01-0AB0 6ES7513-1AL01-0AB0 6ES7513-1FL01-0AB0 6ES7515-2AM01-0AB0 6ES7515-2FM01-0AB0 6ES7515-2TM01-0AB0 6ES7516-3AN01-0AB0 6ES7516-3FN01-0AB0 6ES7516-2PN00-0AB0 6ES7516-2GN00-0AB0 6ES7517-3AP00-0AB0 6ES7517-3FP00-0AB0 6ES7517-3TP00-0AB0 6ES7517-3UP00-0AB0 6ES7518-4AP00-0AB0 6ES7518-4FP00-0AB0 6ES7518-4AP00-3AB0 6ES7518-4FP00-3AB0	V2.1.0	Third-party software - Licensing terms and copyright information You can find the copyright information for third-party software contained in this product, particularly open source software, as well as applicable licensing terms of such third-party software in the Readme_OSS_V210 file. Special information for resellers The information and the license terms in the Readme_OSS_V210 file must be passed on to the purchasing party to avoid license infringements by the reseller or purchasing party. ReadMe_OSS_V210.htm (3.0 MB) Recommended for update: see description Update V2.1.0 (CPUs) S7-1500_CPUs_V210.ZIP (543.7 MB)
		Third-party software - Licensing terms and copyright information

© Siemens AG 2009-2017 - Imprint | Privacy policy | Cookie policy | Terms of use | Digital ID

- If you have not already registered, do so in the next window. (→ "Yes I would like to register now"), or if already registered – log in with your "Login" and "Password". (→ "Login")



Enter the requested data for the registration, select the option "Download of export restricted software" and then save the registration. (→ → "Save")

Download of export restricted software

Access authorization for the download of export-restricted software
Software that is subject to export restrictions may not be made generally accessible. Access authorization to export-restricted software may only be granted to selected, registered users.

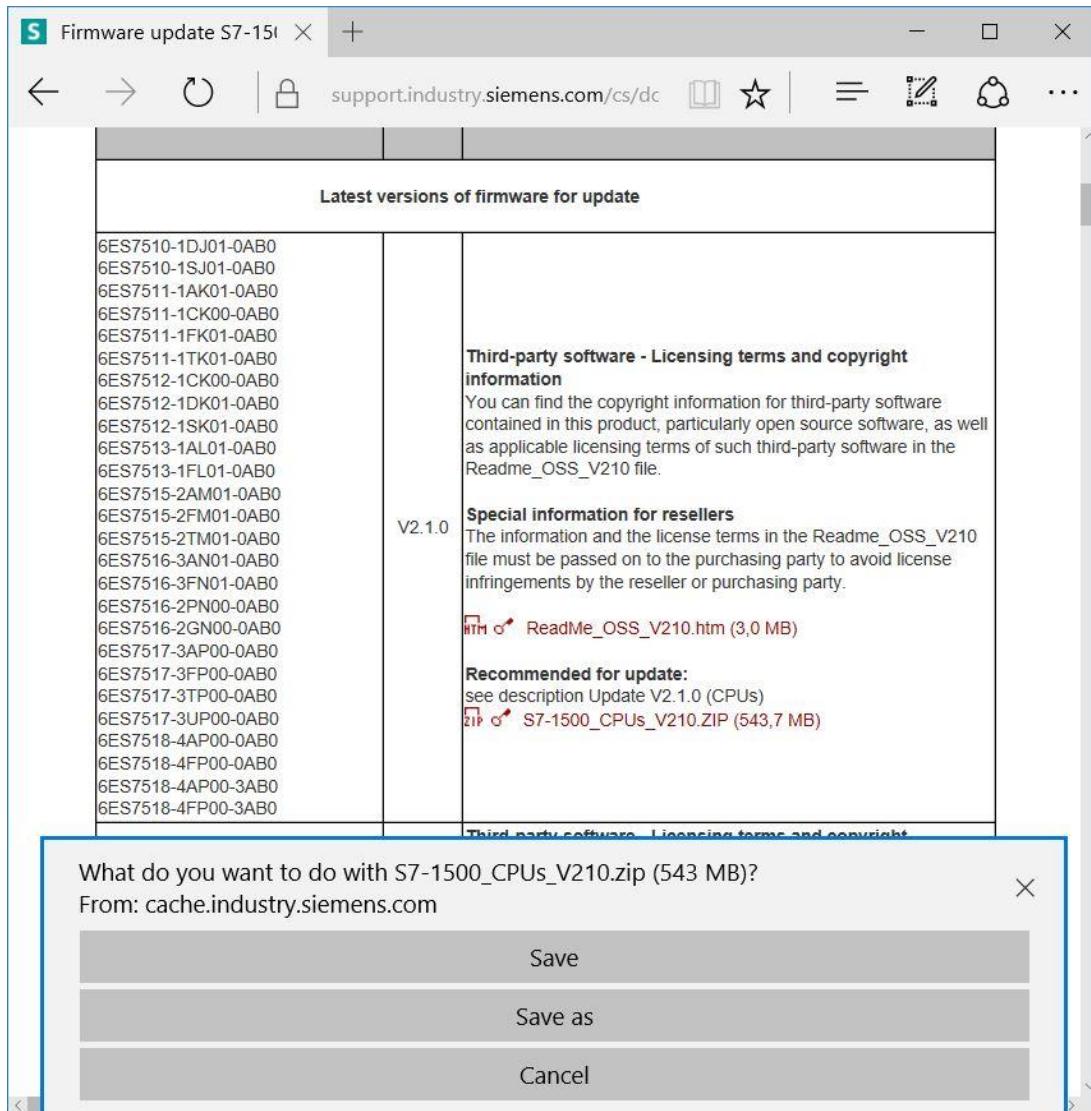
Yes, I would like to register for access to export-restricted software

* Mandatory field

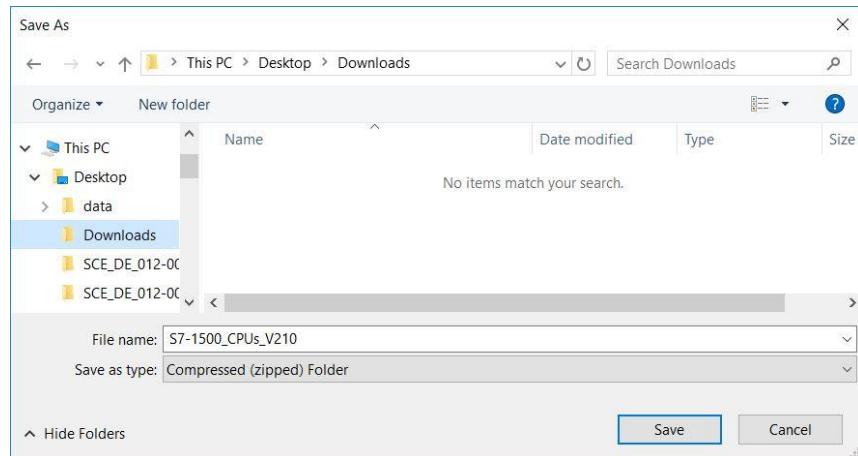
Save

Return to the login. After login you can save the firmware update on your computer.

(→ "Save as")



→ Enter the desired memory location on your computer and click → "Save".



- Back on the Industry Online Support website of SIEMENS AG, you can select the firmware update for your display. Select the update recommended for the upgrade. You can then save the firmware update on your computer. (→ "Save as")

Display for S7-1500:

CPU type	Article number	FW download
Display for CPU 1511(F), CPU 1511T, CPU 1511C, CPU 1512C and CPU 1513(F)	6ES7 591-1AA00-0AA0 6ES7 591-1AA01-0AA0	> 78301954
Display for CPU 1515(F), CPU 1515T, CPU 1516(F), CPU 1517(F), CPU 1517T(F), CPU 1518(F) and CPU 1518(F) ODK	6ES7 591-1BA00-0AA0 6ES7 591-1BA00-0AA0	> 78300948

Firmware Update for the Displays of CPUs 1515(F)/1516(F)/1517(F)/1518(F)

In this entry all firmware version are provided for the displays of CPUs 1515(F)/1516(F)/1517(F)/1518(F).

DESCRIPTION:

When updating the firmware, always update to the **latest version** available for the product and its respective article number. The previous versions of the firmware are only intended as backup to allow a downgrade to the original version. Until now this is not known to have been necessary in any case.

The respective latest version of a firmware is valid for all versions of that article number.

How to update the firmware is described in detail in entry ID > 77492231

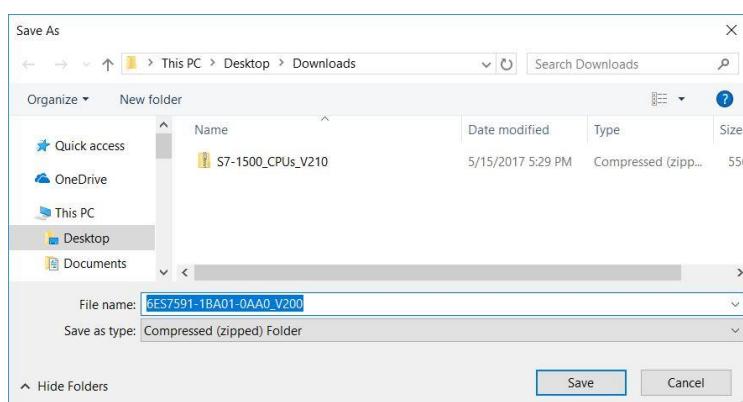
Overview of article numbers and firmware versions of the displays of CPUs 1515(F)/1516(F)/1517(F)/1518(F):

Article Number	Software- Version	Update with...
6ES7591-1BA01-0AA0	V2.0.0	Recommended for update: see description update V2.0.0 ZIP 6ES7591-1BA01-0AA0_V200.ZIP (4.5 MB)
	V1.8.0	Backup only: see description update V1.8.0 ZIP 6ES7591-1BA01-0AA0_V180.ZIP (4.4 MB)

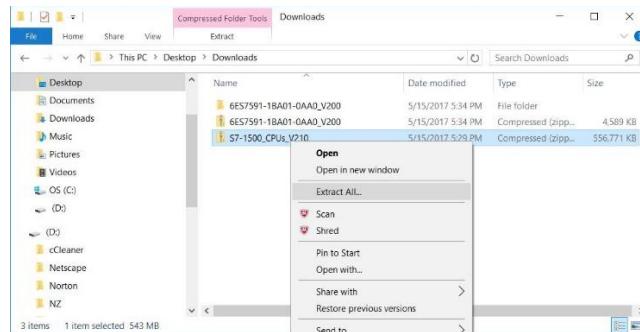
What do you want to do with 6ES7591-1BA01-0AA0_V200.zip (4.48 MB)?
From: cache.industry.siemens.com

Save
Save as
Cancel

Enter the desired memory location on your computer and click → "Save".



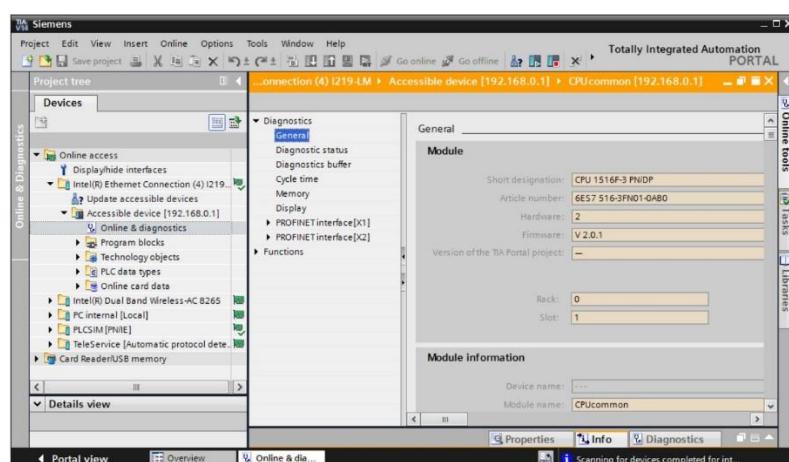
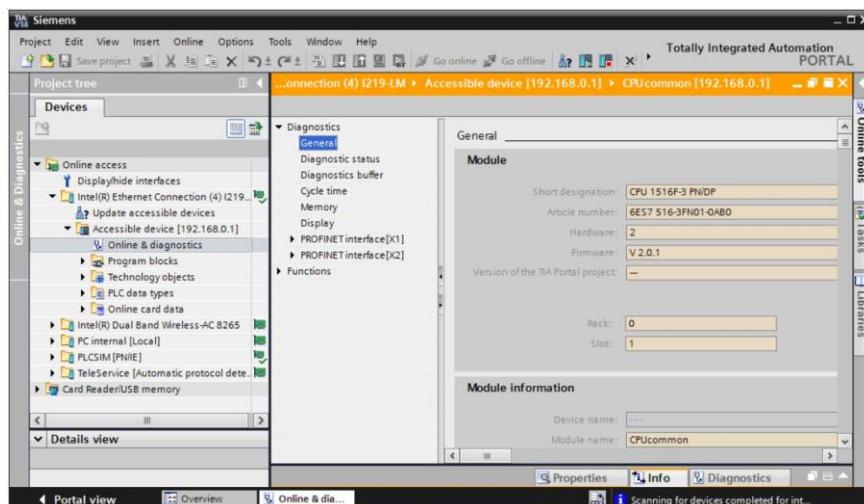
- Click the downloaded compressed files for your CPU and the display in Windows Explorer and select → "Extract All" for each compressed file.



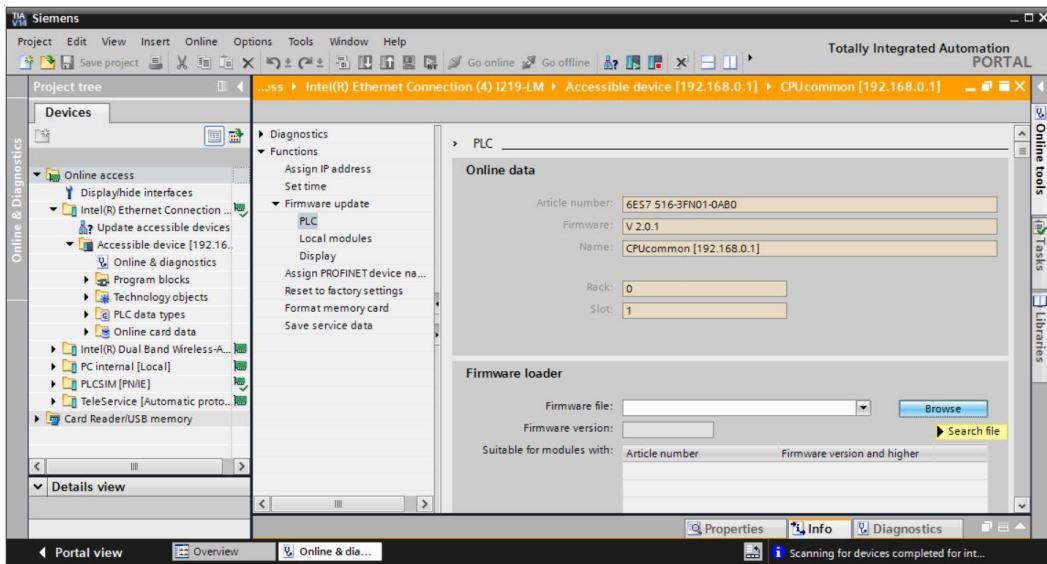
Firmware update of the CPU

The files with the firmware update can be imported into the SIMATIC S7-1500 CPU as follows.

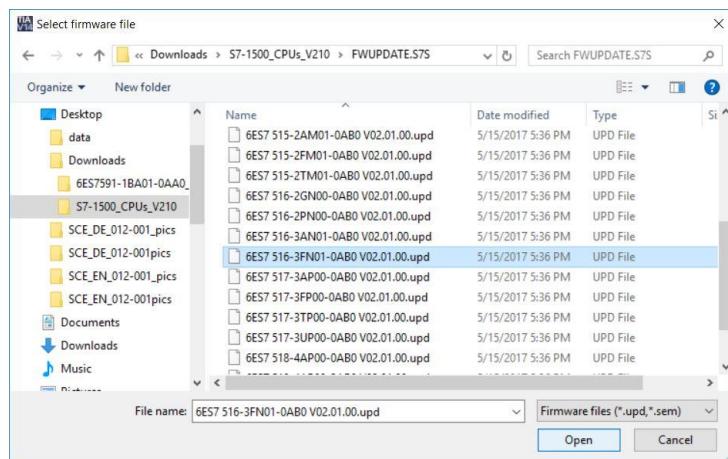
- In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices", you will see the IP address of the connected SIMATIC S7-1500. Select → "Online & Diagnostics". Under the "General" menu item, you can check the current firmware in your CPU in "Diagnostics".



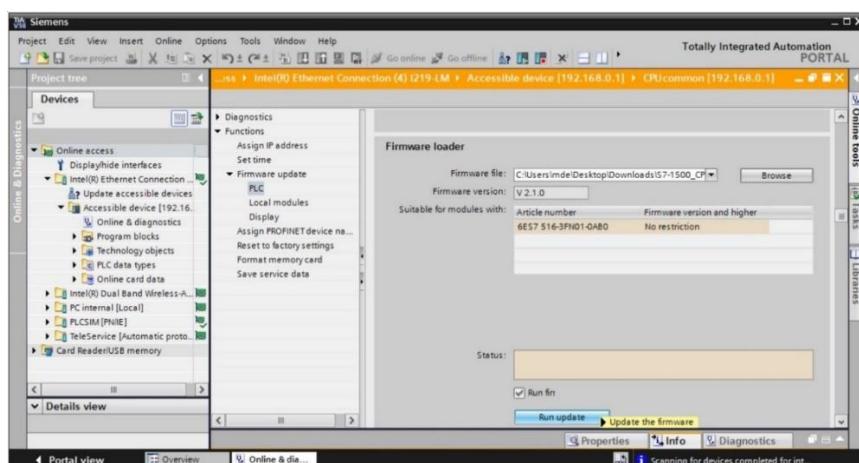
In the → "Functions" menu, change to → "Firmware update" → "PLC". In the → "Firmware loader" → "Browse".



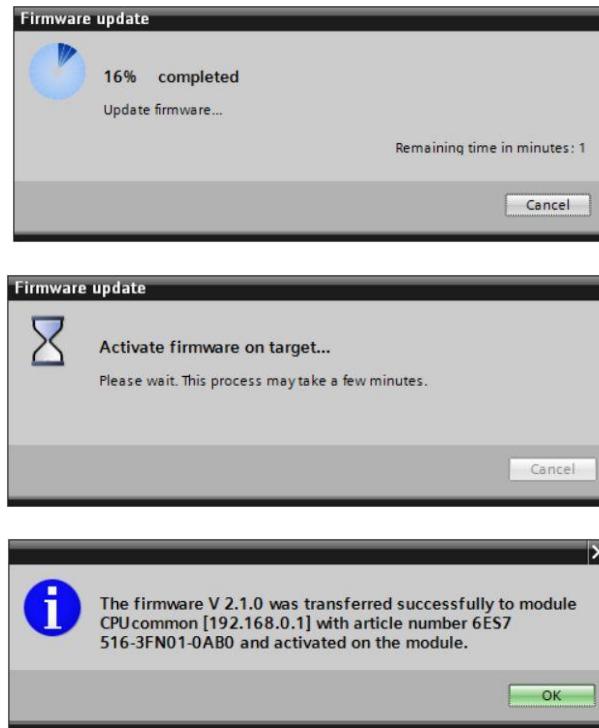
- Select the previously downloaded and extracted firmware file → "6ES7 ***-****-****.upd" on your computer and click → "Open".



- The following dialog indicates whether your firmware file is compatible with your CPU. Now start the update. (→ "Run update")



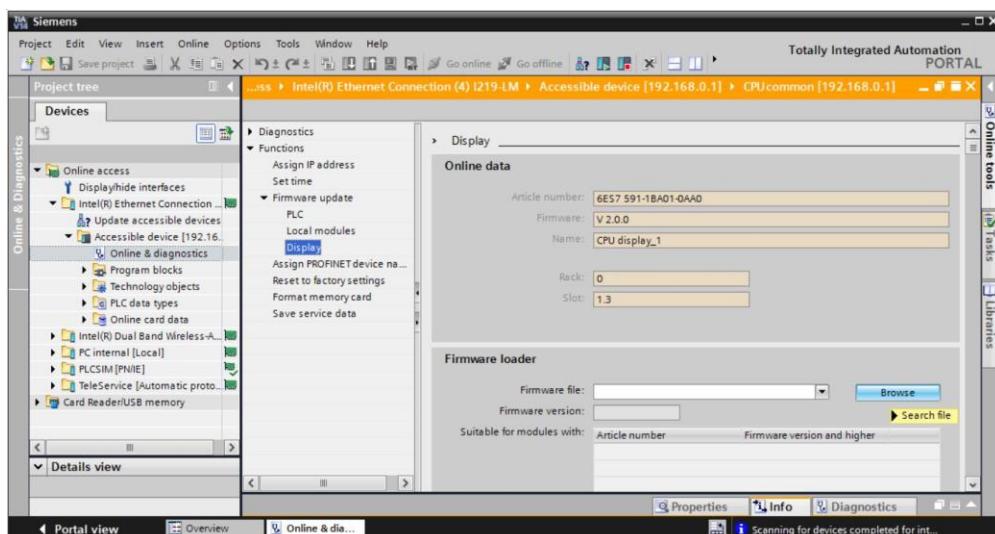
- The progress of the update and its successful completion are indicated with the following dialogs. Click → "OK" to confirm.



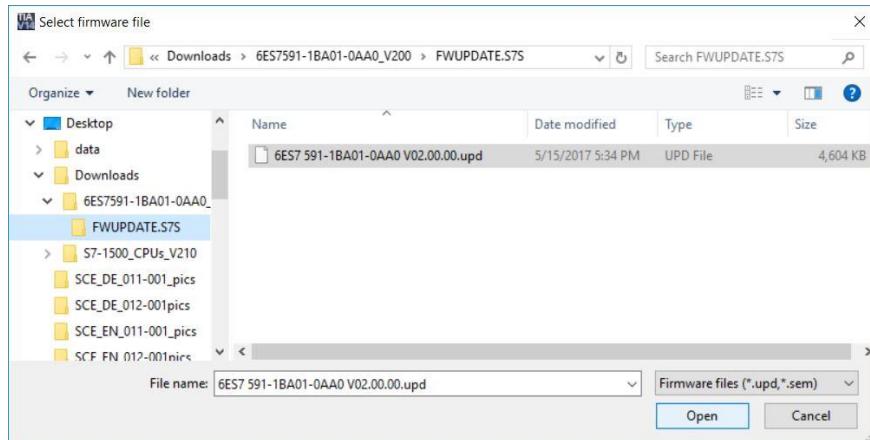
4.3.5 Firmware update of the display

Once the firmware of the SIMATIC S7-1500 CPU has been updated, it is advisable to update the firmware of the display.

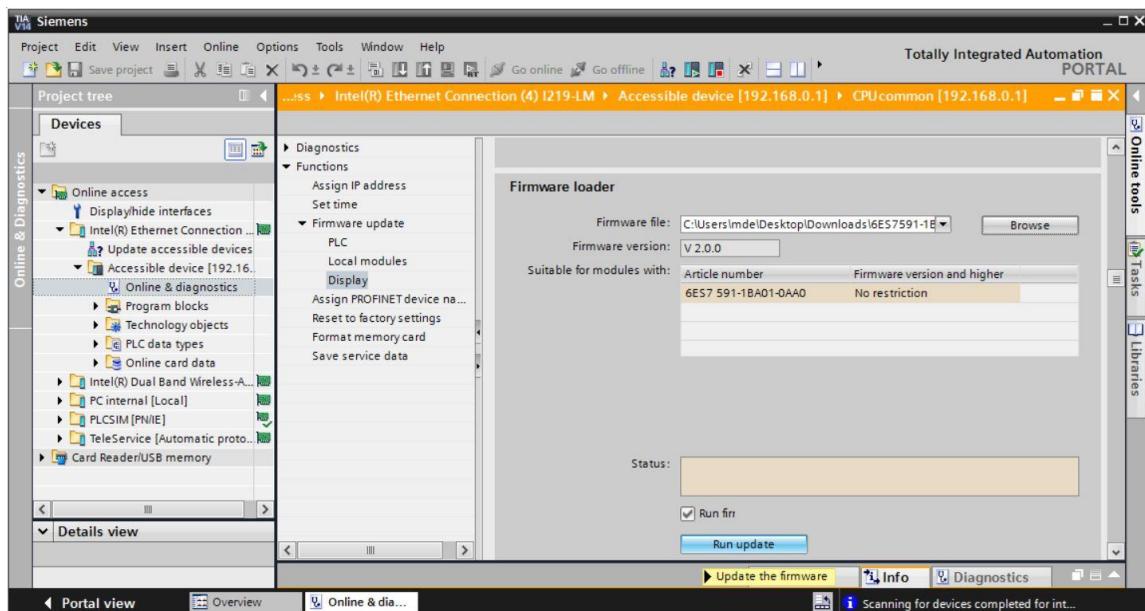
- In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices", you will see the IP address of the connected SIMATIC S7-1500. Select → "Online & Diagnostics" here. In the → "Functions" menu, change to → "Firmware update" → "Display". In the → "Firmware loader" sub-item, click → "Browse".



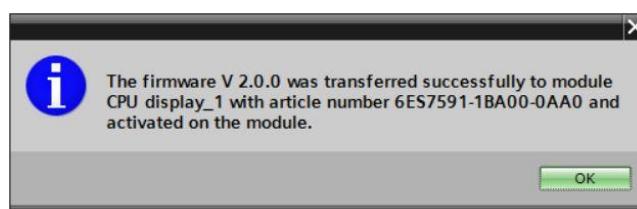
- Select the previously downloaded and extracted firmware file → "6ES7 ***-****-****.upd" on your computer and click → "Open".



- The following dialog indicates whether your firmware file is compatible with your display. Now start the update. (→ "Run update")



- The progress of the update and its successful completion are indicated with the following dialogs. Click → "OK" to confirm.





Training Curriculum

TIA Portal Module 002
Unspecified Hardware Configuration
with SIMATIC S7-1500

Table of contents

1	Goal.....	29
2	Prerequisite.....	29
3	Required hardware and software	29
4	Theory	30
4.1	SIMATIC S7-1500 automation system	30
4.1.1	Range of modules.....	31
4.1.2	Example configuration	33
4.2	Operator control and display elements of the CPU 1516F-3 PN/DP.....	34
4.2.1	Front view of the CPU 1516F-3 PN/DP with integrated display	34
4.2.2	Status and error displays	34
4.2.3	Operator control and connection elements behind the front flap	35
4.2.4	SIMATIC memory card	35
4.2.5	Mode switch	36
4.2.6	Display of the CPU	36
4.3	Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card	38
4.4	STEP 7 Professional V1X (TIA Portal V1X) programming software	40
4.4.1	Project.....	40
4.4.2	Hardware configuration.....	40
4.4.3	Central and distributed automation structure	41
4.4.4	Planning the hardware	42
4.4.5	TIA Portal – Project view and portal view	42
4.4.6	Basic settings for the TIA Portal	43
4.4.7	Setting the IP address on the programming device.....	44
4.4.8	Setting the IP address in the CPU.....	46
4.4.9	Formatting the memory card in the CPU	48
4.4.10	Resetting the CPU to factory settings.....	49
5	Task.....	51
6	Planning.....	51
7	Structured step-by-step instructions.....	52

7.1	Create a new project.....	52
7.2	Read the hardware of the SIMATIC S7-1500	53
7.3	Configure the Ethernet interface of the CPU 1516F-3 PN/DP	58
7.4	Configure the access level for the CPU 1516F-3 PN/DP	59
7.5	Insert power module PM 190W 120/230VAC	59
7.6	Configure the address areas of the digital input and output modules	60
7.7	Save and compile the hardware configuration	61
7.8	Download the hardware configuration to the device.....	62
7.9	Checklist.....	67

UNSPECIFIED HARDWARE CONFIGURATION – FOR A SIMATIC S7-1500

1 Goal

In this chapter, you will first learn how to ***create a project***. Next you will be shown in one part of the task how you can use the **TIA Portal** to detect **hardware** already installed and add it to a project. This hardware will then be configured.

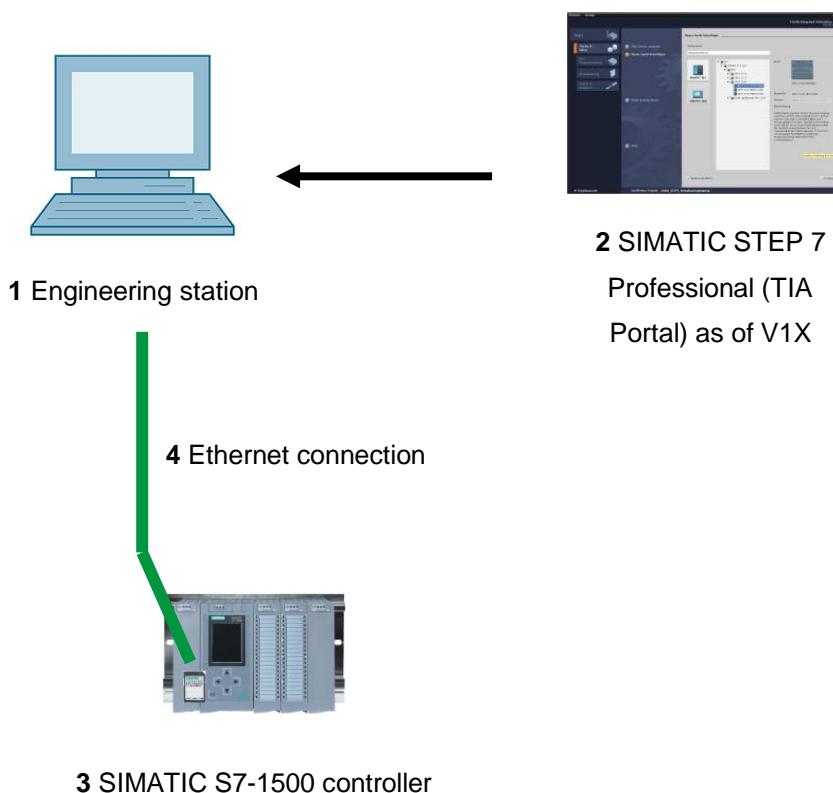
The SIMATIC S7 controllers listed in Chapter 3 can be used.

2 Prerequisite

You do not need any previous knowledge from other chapters to successfully complete this chapter. You only need an S7-1500 controller.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 SIMATIC S7-1500 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
- 4 Ethernet connection between engineering station and controller



4 Theory

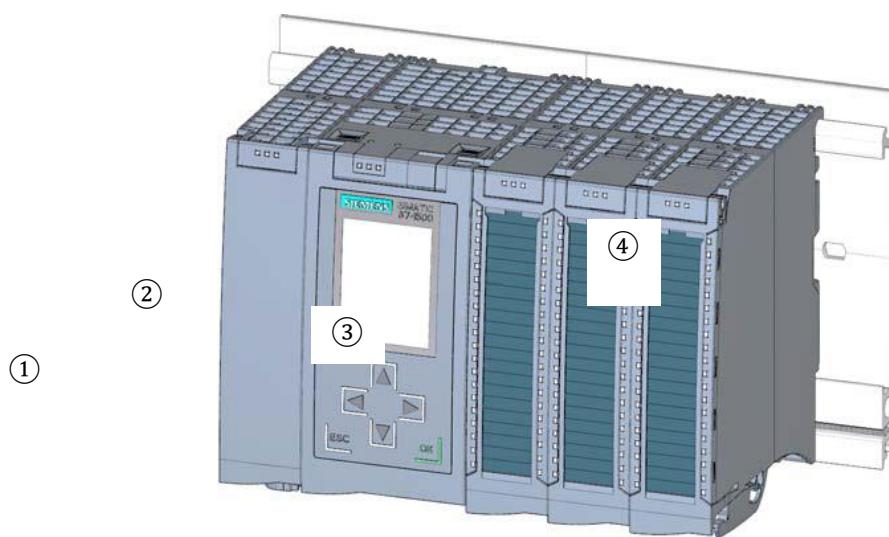
4.1 SIMATIC S7-1500 automation system

The SIMATIC S7-1500 automation system is a modular controller system for the middle to upper performance range. A comprehensive range of modules is available to optimally adapt the system to the automation task.

SIMATIC S7-1500 is the next generation of the SIMATIC S7-300 and S7-400 automation systems with the following new performance features.

- Increased system performance
- Integrated motion control functionality
- PROFINET IO IRT
- Integrated display for machine-level operation and diagnostics
- STEP 7 language innovations while maintaining proven functions

The S7-1500 controller consists of a power supply ①, a CPU with integrated display ② and input and output modules for digital and analog signals ③. The modules are mounted on a mounting rail with integrated DIN rail profile ④. If necessary, communication processors and function modules are also used for special tasks such as stepper motor control.



The programmable logic controller (PLC) uses the S7 program to monitor and control a machine or process. In doing so, the S7 program scans the IO modules via input addresses (%I) and addresses their output addresses (%Q).

The system is programmed with the STEP 7 Professional V1X software.

4.1.1 Range of modules

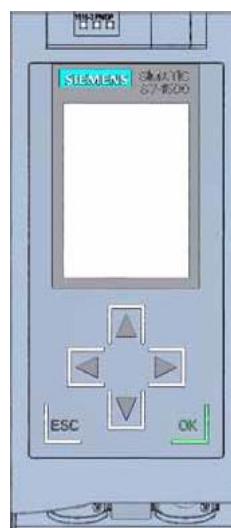
The SIMATIC S7-1500 is a modular automation system and offers the following range of modules:

Central processing units (CPUs) with integrated display

The CPUs have different performance capability and execute the user program. In addition, the other modules are supplied power via the backplane bus with the integrated system power supply.

Additional properties and functions of the CPU:

- Communication via Ethernet
- Communication via PROFIBUS/PROFINET
- HMI communication for HMI devices
- Web server
- Integrated technology functions (e.g. PID controller, motion control, etc.)
- System diagnostics
- Integrated security (e.g. know-how, copy, access, integrity protection)



System power supply modules (PS) (rated input voltages 24 V DC to 230 V AC/DC)

with connection to the backplane bus supply the configured modules with the internal supply voltage.

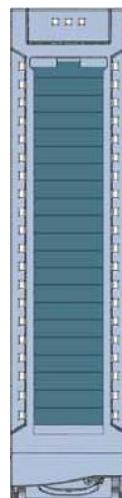


Load current supply modules (PM) (rated input voltages 120/230 V AC)

do not have a connection to the backplane bus of the S7-1500 automation system. The load current supply is used to supply 24 V DC to the system power supply of the CPU, the input and output circuits of IO modules and the sensors and actuators.

**IO modules**

for digital input (DI) / digital output (DO) / analog input (AI) / analog output (AQ)

**Technology modules (TM)**

as incremental encoders and pulse encoders with/without direction signal

**Communication modules (CM)**

for serial communication RS232 / RS422 / RS485, PROFIBUS and PROFINET



SIMATIC memory card

up to a maximum of 2 GB for storing program data and for easy replacement of CPUs during maintenance.



4.1.2 Example configuration

The following configuration of an S7-1500 automation system will be used for the program example in this curriculum.



- ① Load current supply module (PM) with 120/230 V AC, 50 Hz / 60 Hz, 190 W input and 24 V DC / 8 A output
- ② Central processing unit CPU 1516F-3 PN/DP with integrated PROFIBUS and PROFINET interfaces
- ③ IO module 32x digital input DI 32x24VDC HF
- ④ IO module 32x digital output DQ 32x24VDC/0.5A HF
- ⑤ IO module 8x analog input AI 8xU/I/RTD/TC ST

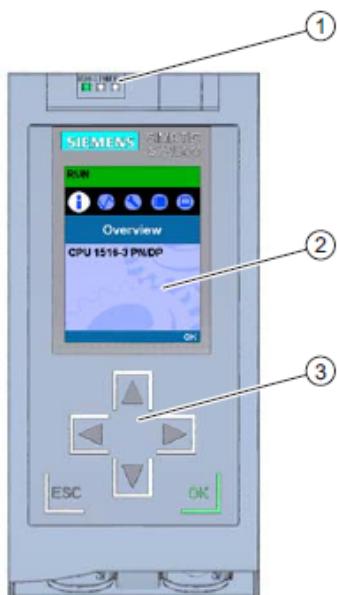
- ⑥ IO module 4x analog output AQ 4xU/I ST

4.2 Operator control and display elements of the CPU 1516F-3 PN/DP

The figure below shows the operator control and display elements of a CPU 1516F-3 PN/DP.

The arrangement and number of elements differ from this figure for other CPUs.

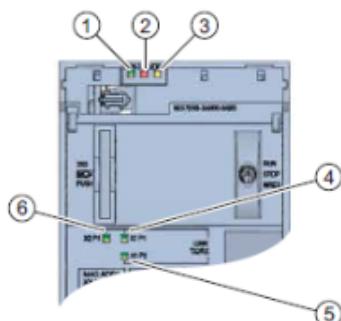
4.2.1 Front view of the CPU 1516F-3 PN/DP with integrated display



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display
- ③ Control keys

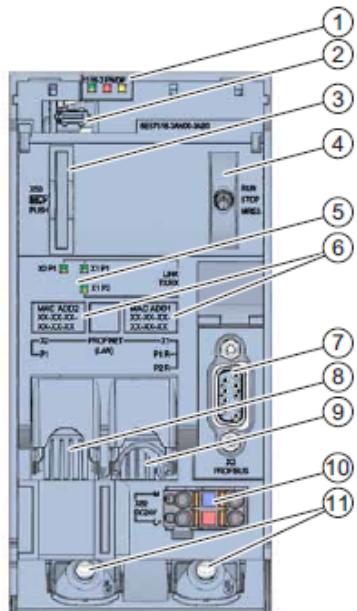
4.2.2 Status and error displays

The CPU comes with the following LED displays:



- ① RUN/STOP LED (yellow/green LED)
- ② ERROR LED (red LED)
- ③ MAINT LED (yellow LED)
- ④ LINK RX/TX LED for port X1 P1 (yellow/green LED)
- ⑤ LINK RX/TX LED for port X1 P2 (yellow/green LED)
- ⑥ LINK RX/TX LED for port X2 P1 (yellow/green LED)

4.2.3 Operator control and connection elements behind the front flap



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display connection
- ③ Slot for the SIMATIC memory card
- ④ Mode switch
- ⑤ LED displays for the 3 ports of the PROFINET interfaces X1 and X2
- ⑥ MAC addresses of the interfaces
- ⑦ PROFIBUS interface (X3)
- ⑧ PROFINET interface (X2) with 1 port
- ⑨ PROFINET interface (X1) with 2-port switch
- ⑩ Connection for supply voltage
- ⑪ Fastening screws

Note: The front flap with the display can be removed and inserted during operation.

4.2.4 SIMATIC memory card

A SIMATIC Micro Memory Card is used as the memory module for the CPUs. This is a preformatted memory card that is compatible with the Windows file system. It is available with various storage capacities and can be used for the following purposes:

- Transportable data storage medium
- Program card
- Firmware update card

The MMC **must** be inserted to operate the CPU as the CPUs have no integrated load memory. A commercially available SD card reader is needed to write/read the SIMATIC memory card with the programming device or PG/PC. This allows files to be copied directly to the SIMATIC memory card using Windows Explorer, for example.

Note: It is recommended that the SIMATIC memory card only be removed or inserted when the CPU is in the POWER OFF state.

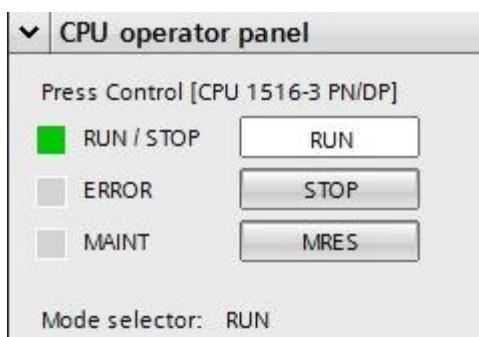
4.2.5 Mode switch

The mode switch allows you to set the operating mode of the CPU. The mode switch is designed as a toggle switch with 3 switch positions.

Position	Meaning	Explanation
RUN	RUN mode	The CPU processes the user program.
STOP	STOP mode	The CPU is not executing the user program.
MRES	Memory reset	Position for CPU memory reset.

You can also use the button on the CPU operator panel of the STEP 7 Professional V1X software in Online & Diagnostics to switch the operating mode (**STOP** or **RUN**).

The operator panel also contains an **MRES** button for performing a memory reset and displays the status LEDs of the CPU.



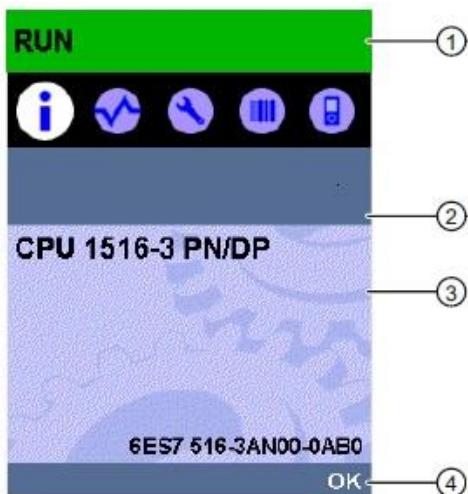
4.2.6 Display of the CPU

The S7-1500 CPU has a front flap with a display and control keys. Control data and status data can be displayed in various menus on the display and numerous settings can be configured. You use the control keys to navigate through the menus.

The display of the CPU offers the following functions:

- 6 different display languages can be selected.
- Diagnostic messages are displayed in plain text.
- The interface settings can be changed locally.
- Password assignment for display operation is possible through the TIA Portal.

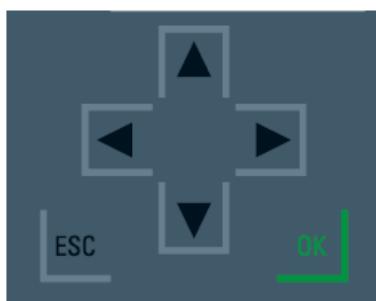
View of the display of an S7-1500:



- ① CPU status information
- ② Submenu name
- ③ Information display field
- ④ Navigation aid, e.g. OK/ESC or the page number

Control keys of the display

- Four arrow keys: "up", "down", "left", "right"
- An ESC key
- An OK key



Functions of the "OK" and "ESC" keys

- For menu commands in which an input can be made:
 - OK → valid access to the menu command, confirmation of input and exit from editing mode
 - ESC → restoration of original content (which means changes are not saved) and exit from editing mode
- For menu commands in which no input can be made:
 - OK → to next submenu command
 - ESC → back to previous menu command

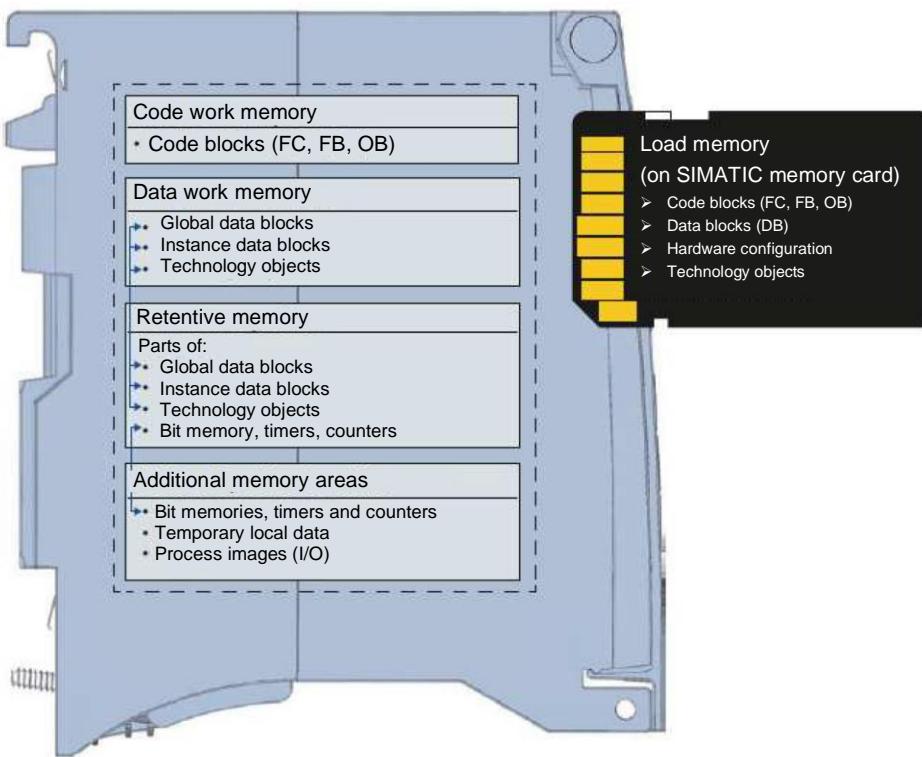
Available submenus of the display:

Main menu commands	Meaning	Explanation
	Overview	The "Overview" menu contains information about the properties of the CPU.
	Diagnostics	The "Diagnostics" menu contains information about diagnostic messages, the diagnostic description and the indication of interrupts. There is also information about the network properties of each interface of the CPU.
	Settings	In the "Settings" menu, the IP addresses of the CPU are assigned, the date, time, time zones, operating modes (RUN/STOP) and protection levels are set, the CPU memory is reset and its factory settings are restored and the status of firmware updates is displayed.
	Modules	The "Modules" menu contains information about the modules that are used in your configuration. The modules can be used as central or distributed modules. Distributed modules are connected to the CPU via PROFINET and/or PROFIBUS. You have the option here to set the IP addresses for a CPU.
	Display	In the "Display" menu, settings are made for all aspects of the display, such as the language setting, brightness setting and Energy-saving mode. (Energy-saving darkens the display. Standby mode switches off the display.)

4.3 Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card

The following figure shows the memory areas of the CPU and the load memory on the SIMATIC memory card.

In addition to the load memory, other data can be loaded onto the SIMATIC memory card using Windows Explorer. This includes recipes, data logs, project backups and additional documentation for the program.



Load memory

Load memory is non-volatile memory for code blocks, data blocks, technology objects and the hardware configuration. When these objects are downloaded to the CPU, they are first stored in the load memory. This memory is located on the SIMATIC memory card.

Work memory

Work memory is volatile memory that contains the code and data blocks. The work memory is integrated into the CPU and cannot be expanded. In S7-1500 CPUs, the work memory is divided into two areas:

- Code work memory:
 - The code work memory contains runtime-relevant parts of the program code.
- Data work memory:
 - The data work memory contains the runtime-relevant parts of the data blocks and technology objects.

At the operating mode transitions from POWER ON to startup and from STOP to startup, tags of global data blocks, instance data blocks and technology objects are initialized with their start values. Retentive tags retain their actual values that were saved in the retentive memory.

Retentive memory

Retentive memory is non-volatile memory for saving certain data in the event of power failure. The tags and operand areas that have been defined as retentive are saved in the retentive memory. This data is retained beyond power-off or power failure.

All other program tags are set to their start values at the operating mode transitions from POWER ON to startup and from STOP to startup.

The content of retentive memory is deleted by the following actions:

- Memory reset
- Reset to factory settings

Note: Certain tags of technology objects are also stored in the retentive memory. These tags are not deleted by a memory reset.

4.4 STEP 7 Professional V1X (TIA Portal V1X) programming software

STEP 7 Professional V1X (TIA Portal V1X) software is the programming tool for the following automation systems:

- SIMATIC S7-1500
- SIMATIC S7-1200
- SIMATIC S7-300
- SIMATIC S7-400
- SIMATIC WinAC

STEP 7 Professional V1X provides the following functions for plant automation:

- Configuration and parameter assignment of the hardware
- Specification of the communication
- Programming
- Testing, commissioning and servicing with operational/diagnostic functions
- Documentation
- Creation of visualizations for SIMATIC Basic Panels using the integrated WinCC Basic software
- Visualization solutions for PCs and other panels can also be created with other WinCC software packages

Support is provided for all functions through detailed online help.

4.4.1 Project

To implement a solution for an automation and visualization task, you create a project in the TIA Portal. A project in the TIA Portal contains the configuration data for the configuration and internetworking of devices as well as the programs and the configuration of the visualization.

4.4.2 Hardware configuration

The *hardware configuration* includes the configuration of the devices, consisting of the hardware of the automation system, the intelligent field devices and the hardware for visualization. The configuration of the networks specifies the communication between the various hardware components. The individual hardware components are *inserted in the hardware configuration* from catalogs.

The hardware of automation systems comprises controllers (CPUs), signal modules for input and output signals (SMs) and communication processors, and interface modules (CP, IM). Power supply and voltage supply modules (PS, PM) are also available to supply the modules.

The signal modules and intelligent field devices connect the input and output data of the process to be automated and visualized to the automation system.

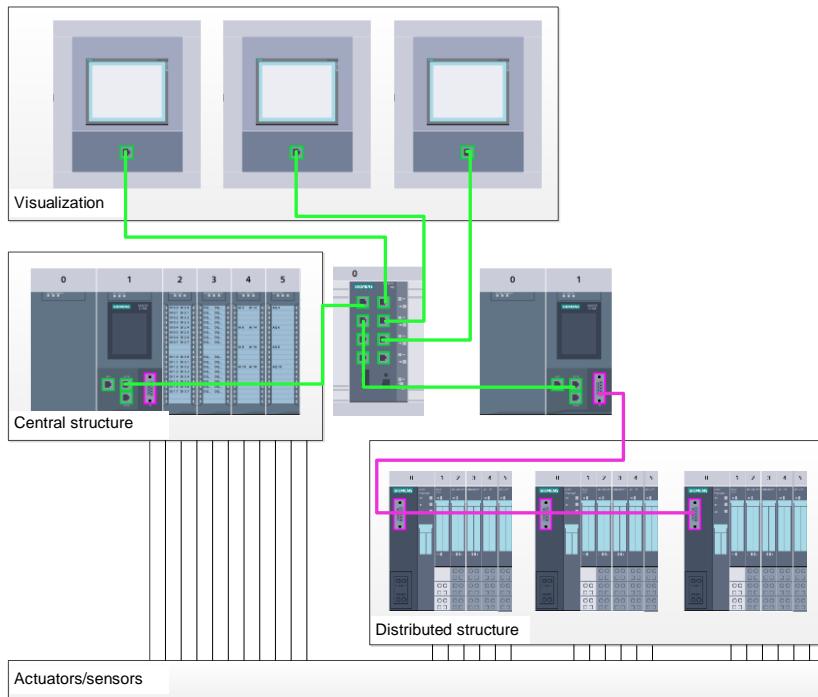


Figure 1: Example of hardware configuration with central and distributed structures

The hardware configuration enables the downloading of automation and visualization solutions to the automation system and access to the connected signal modules by the controller.

4.4.3 Central and distributed automation structure

Figure 1 shows an automation structure that contains both central and distributed structures.

In central structures, the input and output signals of the process are transmitted by way of conventional wiring to the signal modules, which are connected directly to the controller.

Conventional wiring refers to the connection of sensors and actuators using 2-wire or 4-wire cables.

The distributed structure is the predominant structure used today. Here, the sensors and actuators are wired conventionally only as far as the signal modules of the field devices. The signal transmission from the field devices to the controller is implemented using an industrial communication system.

Both classic fieldbuses such as PROFIBUS, Modbus and Foundation Fieldbus as well as Ethernet-based communication systems such as PROFINET can be used as the industrial communication system.

In addition, intelligent field devices in which stand-alone programs run can also be connected via the communication system. These programs can also be created with the TIA Portal.

4.4.4 Planning the hardware

Before you can configure the hardware, you must plan it (hardware planning). In general, you begin by selecting which controllers are needed and how many. Next you select the communication modules and signal modules. The selection of signal modules is based on the number and type of inputs and outputs needed. As the final step, a power supply that ensures that the necessary power is supplied must be selected for each controller or field device.

The functionality required and the ambient conditions are of vital importance for planning the hardware configuration. For example, the temperature range in the application area sometimes limits the devices available for selection. Fail-safe operation might be another requirement, for example.

The [TIA Selection Tool](#) (Select automation technology → TIA Selection Tool and follow the instructions) provides you support. Note: TIA Selection Tool requires Java.

Note for online research: If more than one manual is available, you should look for the description "Device Manual", "Product Manual" or simply "Manual" (as opposed to "Function Manual", "List Manual", "System Manual", etc.) in order to find the device specifications.

4.4.5 TIA Portal – Project view and portal view

The TIA Portal has two important views. When started, the TIA Portal displays the portal view by default. This view makes getting started easier, especially for beginning users.

The portal view provides a task-oriented view of the tools for working on the project. Here, you can quickly decide what you want to do and open the tool for the task at hand. If necessary, a change to the project view takes place automatically for the selected task.

Figure 2 shows the portal view. At the bottom left, there is an option to switch between this view and the project view.

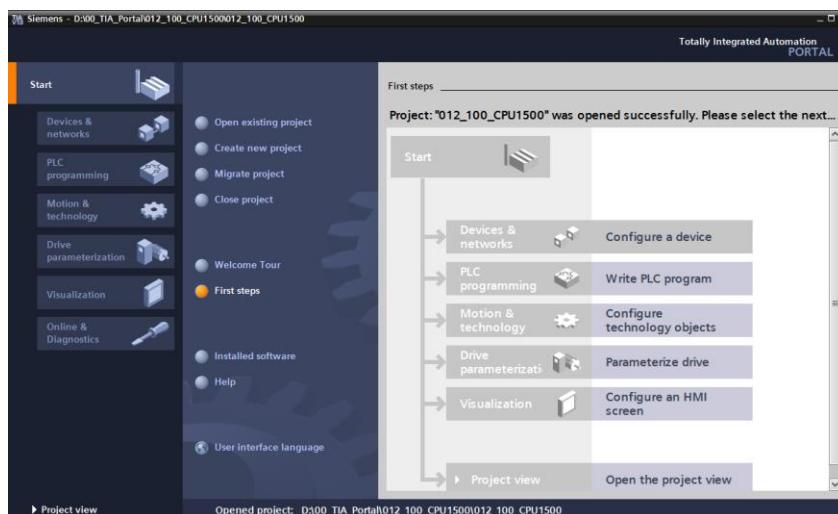


Figure 2: Portal view

The project view, as shown in Figure 3, is used for hardware configuration, programming, creation of the visualization and many other tasks.

By default, the project view displays the menu bar with the toolbars at the top, the project tree with all components of a project on the left and the so-called task cards with instructions and libraries, for example, on the right.

If an element (for example, the device configuration) is selected in the project tree, it is displayed in the center and can be worked on there.

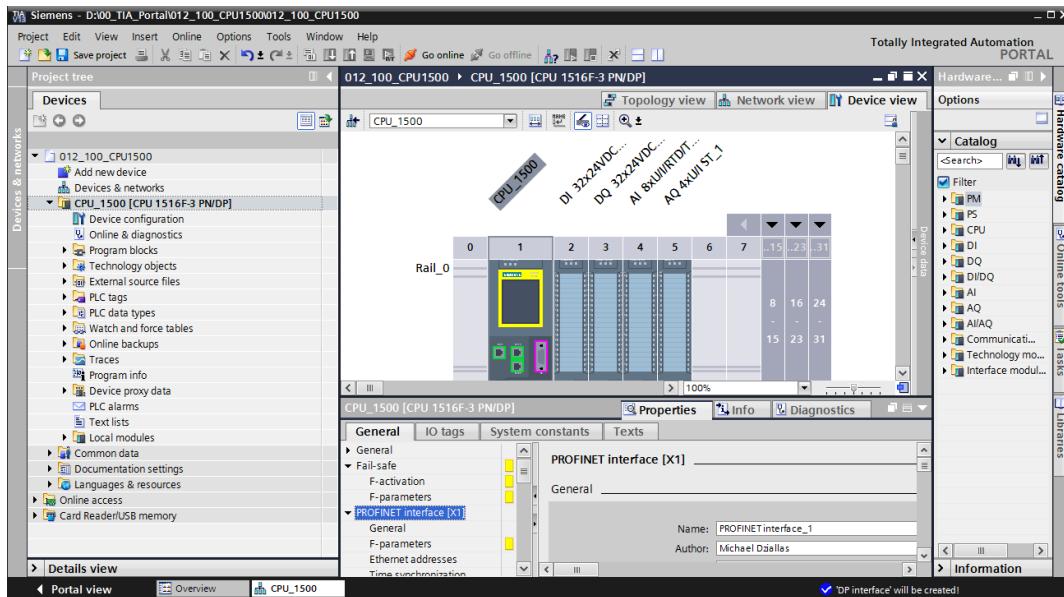
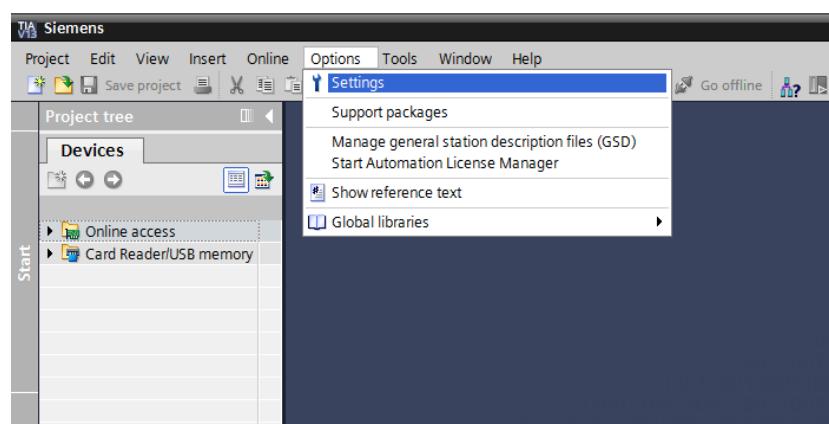


Figure 3: Project view

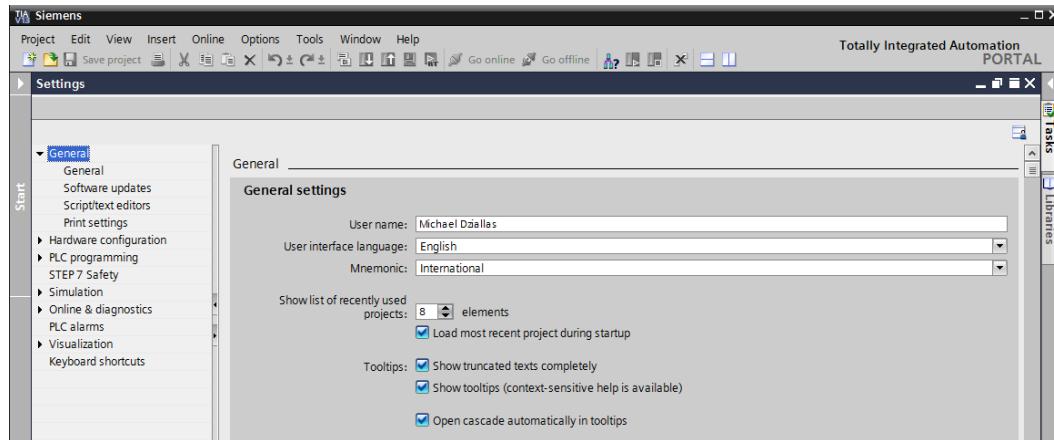
4.4.6 Basic settings for the TIA Portal

- Users can specify their own default settings for certain settings in the TIA Portal. A few important settings are shown here.
- In the project view, select the →"Options" menu and then → "Settings".



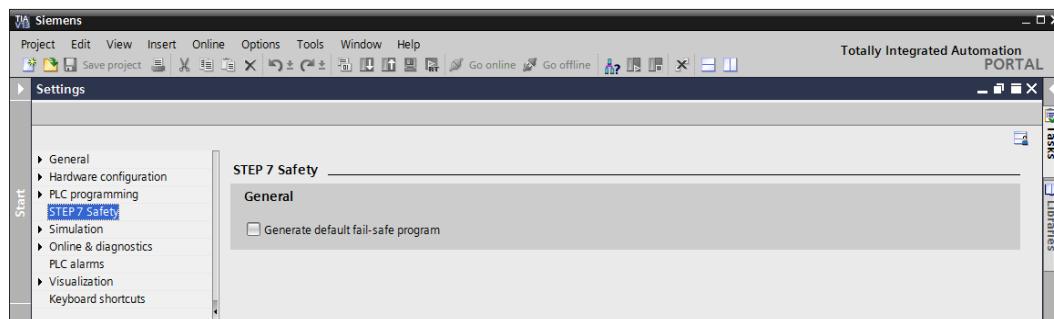
One basic setting is the selection of the user interface language and the language for the program display. In the curriculums to follow, "English" will be used for both settings.

- Under → "General" in "Settings", select "User interface language → English" and "Mnemonic → International".



Note: These settings can always be changed.

- When Safety CPUs are used (e.g. CPU 1516F-3 PN/DP) without the use of safety engineering, it is recommended that automatic creation of the safety program be deactivated before creating a project.
- In "Settings" under the → "STEP 7 Safety" item, deactivate → "Generate default fail-safe program".



4.4.7 Setting the IP address on the programming device

To program SIMATIC S7-1500 from the PC, the programming device or a laptop, you need a TCP/IP connection or an optional PROFIBUS connection.

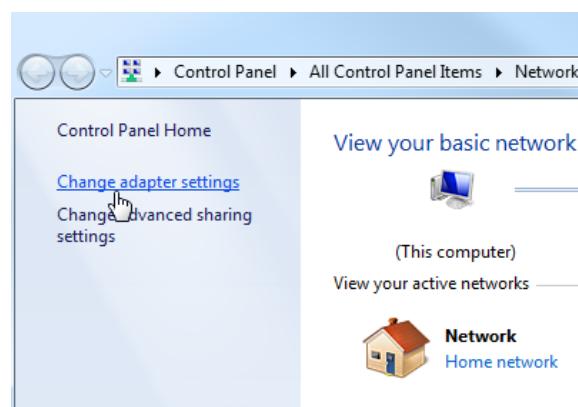
For the PC and SIMATIC S7-1500 to communicate with each other via TCP/IP, it is important that the IP addresses of both devices match.

First, we show you how to set the IP address of a computer with the Windows 7 operating system.

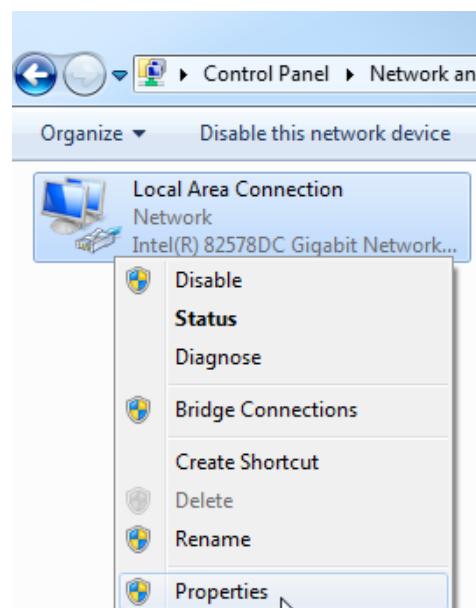
- Locate the network icon in the taskbar at the bottom and click → "Open Network and Sharing Center".



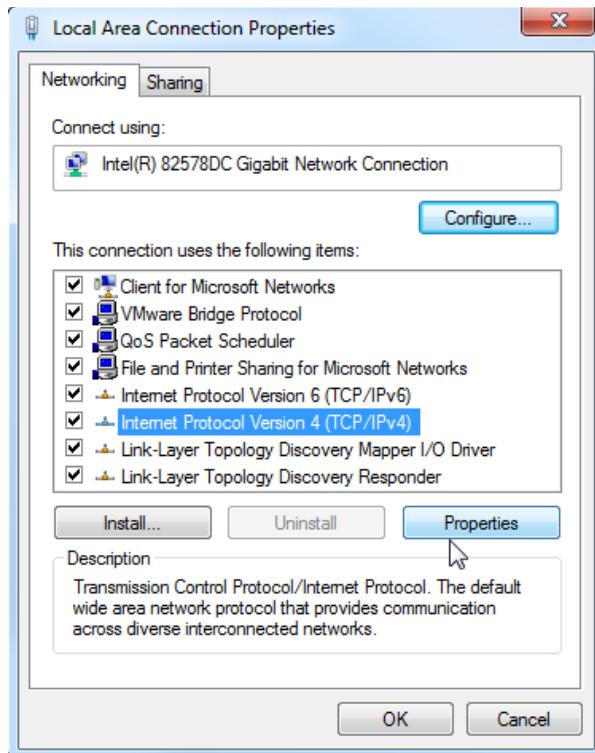
- In the open Network and Sharing Center window, click → "Change adapter settings".



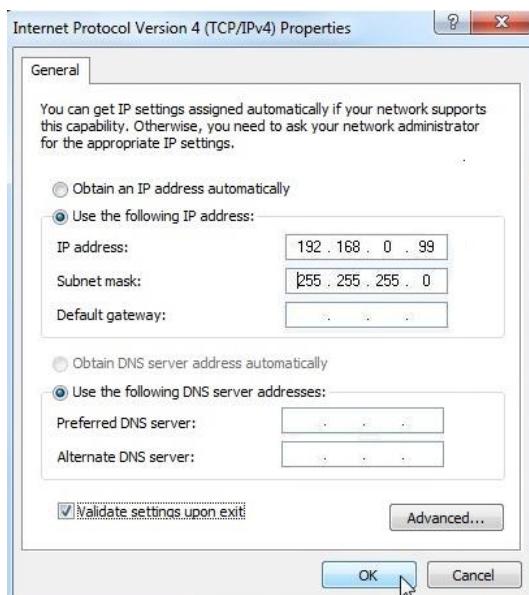
- Select the desired → "Local Area Connection" that you want to use to connect to the controller and click → "Properties".



- Next, select → "Properties" for → "Internet Protocol Version 4 (TCP/IP)".



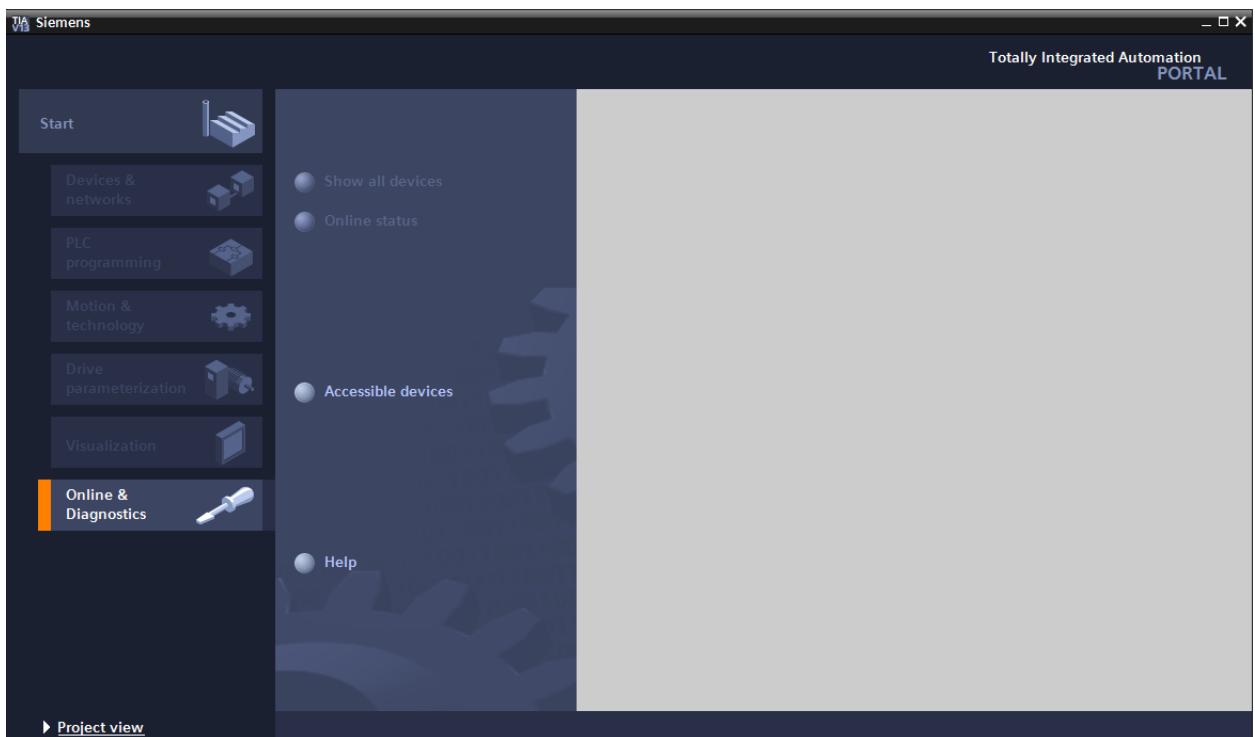
- You can use the following IP address, for example → IP address: 192.168.0.99 → Subnet mask 255.255.255.0 and accept the settings (→ "OK")



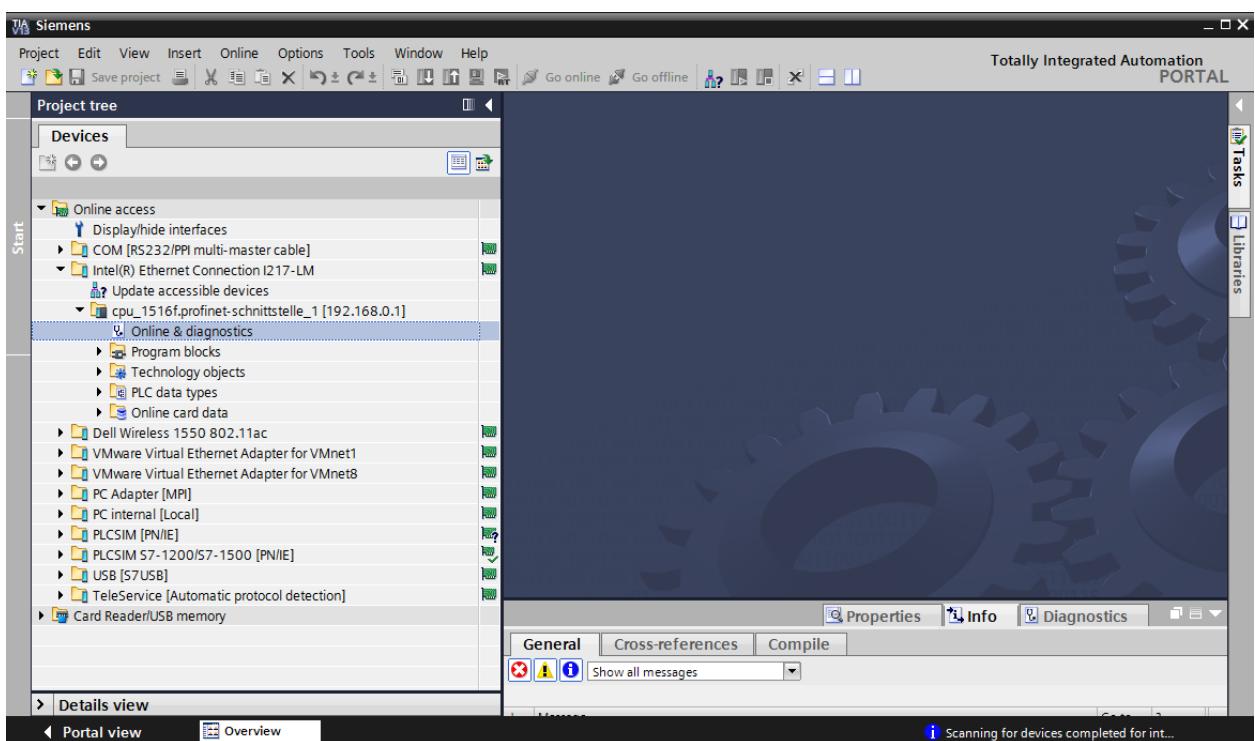
4.4.8 Setting the IP address in the CPU

The IP address of SIMATIC S7-1500 is set as follows.

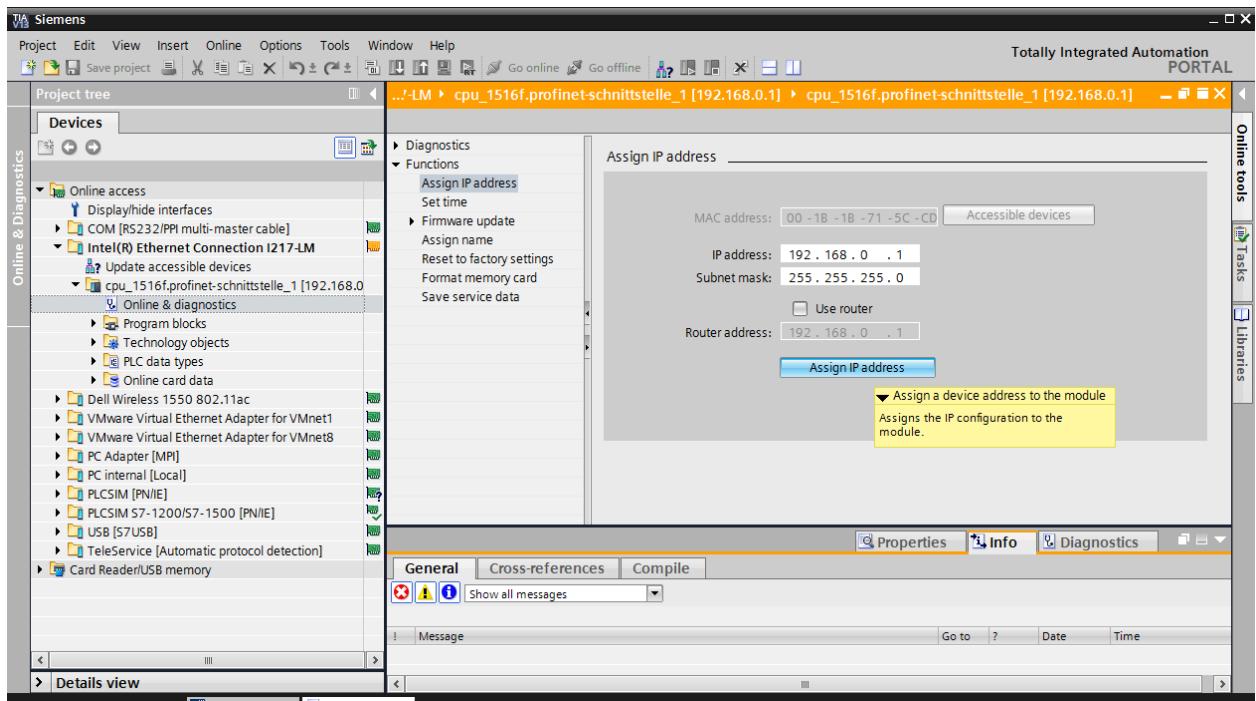
- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V1X)
- Select → "Online & Diagnostics" and open the → "project view".



- In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices" here, you will see the IP address (if previously set) or the MAC address (if IP address not yet assigned) of the connected SIMATIC S7-1500. Select → "Online & Diagnostics" here.

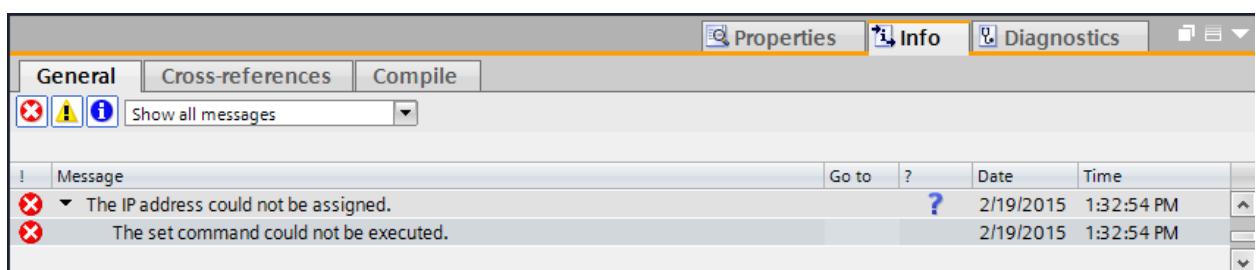


- Under → "Functions", you now find the → "Assign IP address" item. Enter the following IP address here (example): → IP address: 192.168.0.1 → Subnet mask 255.255.255.0.
- Next, click → "Assign IP address" and this new address will be assigned to your SIMATIC S7-1500.



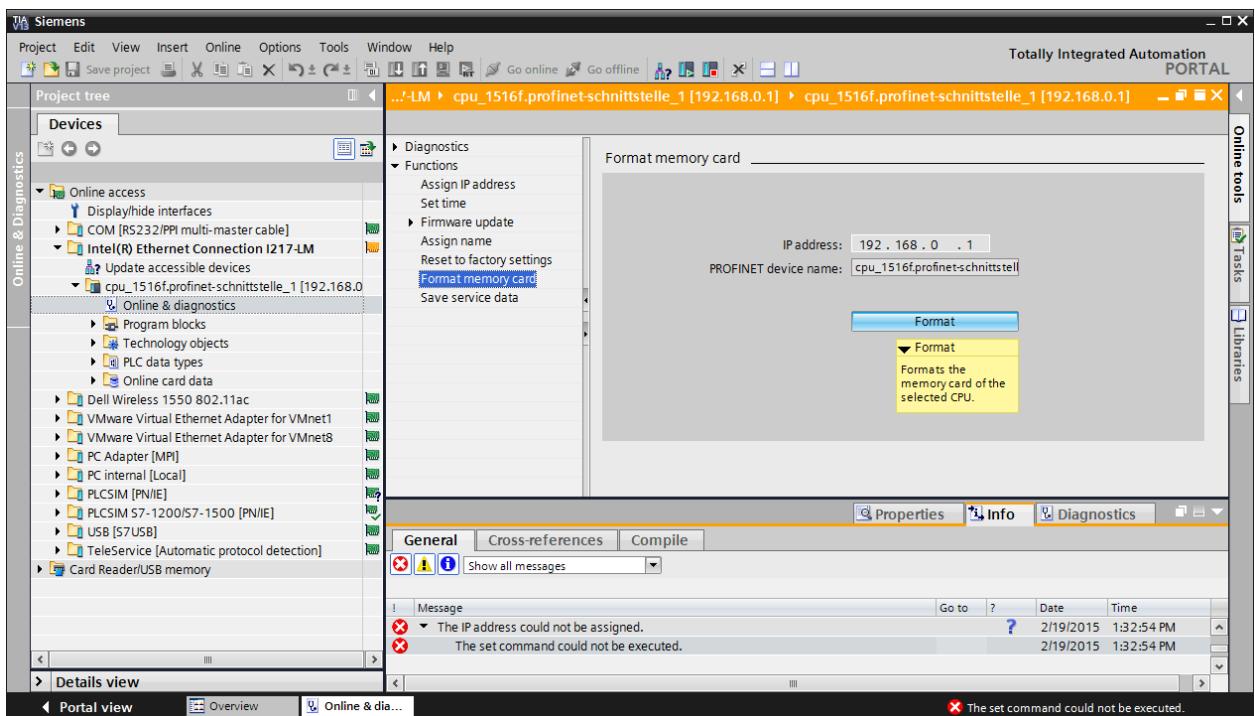
Note: The IP address of the SIMATIC S7-1500 can also be set via the display on the CPU, provided this is enabled in the hardware configuration.

- If the IP address was not successfully assigned, you will receive a message in the → "Info" window under → "General".

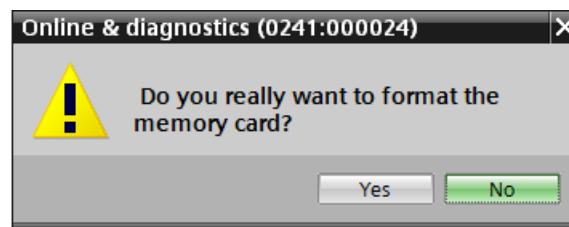


4.4.9 Formatting the memory card in the CPU

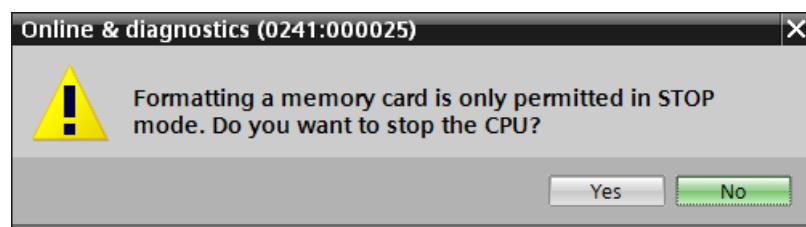
- If the IP address could not be assigned, the program data on the CPU must be deleted. This is accomplished in 2 steps: → "Format memory card" and → "Reset to factory settings".
- First, select the → "Format memory card" function and press the → "Format" button.



→ Confirm the prompt asking if you really want to format the memory card with → "Yes".

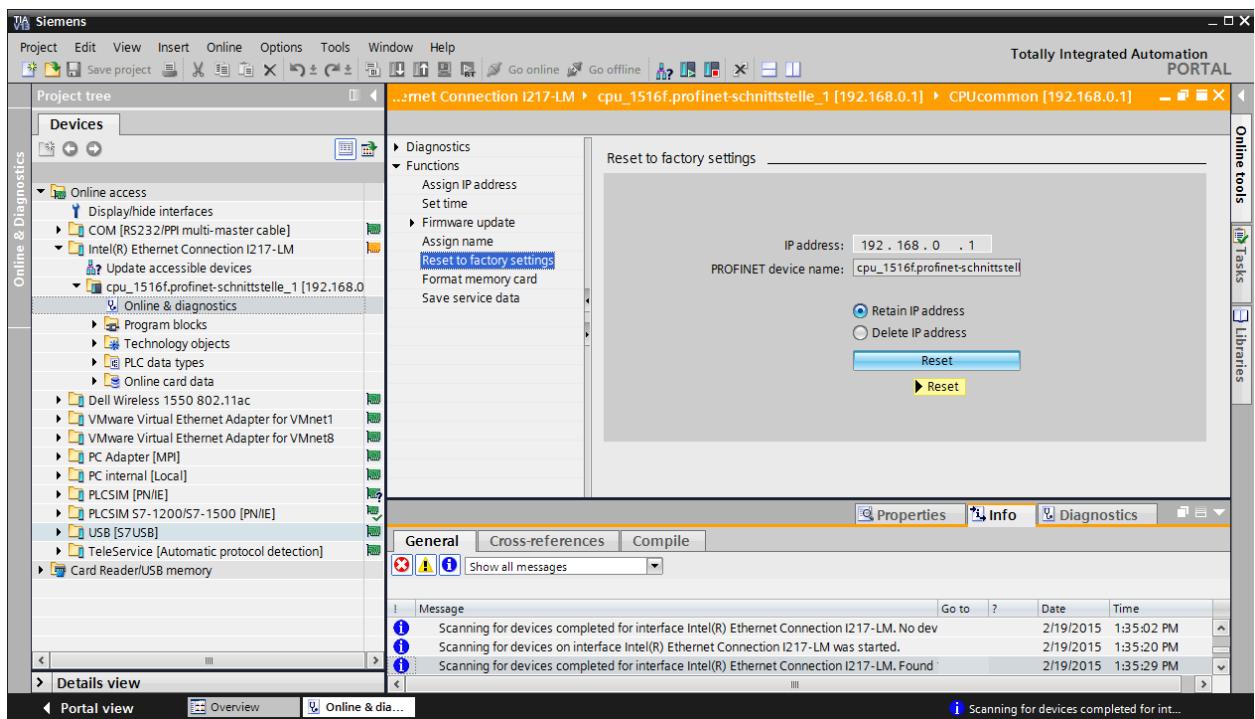


→ If necessary, stop the CPU. (→ "Yes")

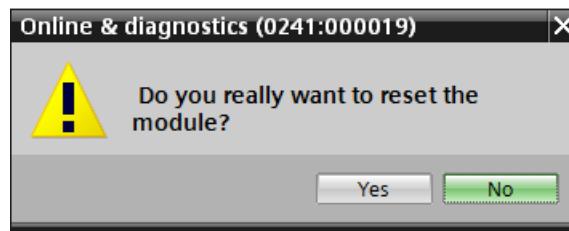


4.4.10 Resetting the CPU to factory settings

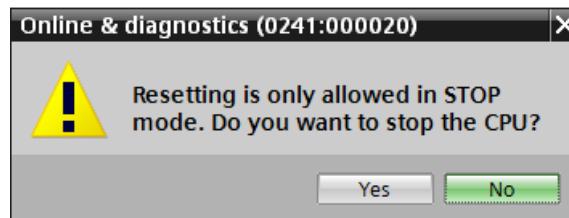
→ Before you can reset the CPU, you must wait until the formatting in the CPU has finished. Then you must select → "Update accessible devices" and → "Online & diagnostics" of your CPU again. To reset the controller, select the → "Reset to factory settings" function and click → "Reset".



→ Confirm the prompt asking if you really want to reset the module with → "Yes".



→ If necessary, stop the CPU. (→ "Yes")



5 Task

Create a project and add the modules of the existing hardware (here: **SIMATIC S7-1500F with CPU 1516F-3 PN/DP**) by using the automatic hardware detection of the **TIA Portal**. The following modules must be detected:

- SIMATIC S7-1500F, CPU 1516F-3 PN/DP, WORK MEMORY 1.5 MB PROGRAM, 5 MB DATA, 1. INTERFACE, PROFINET IRT WITH 2 PORT SWITCH, 2. INTERFACE, ETHERNET, 3. INTERFACE, PROFIBUS, 10 NS BITPERFORMANCE, SIMATIC MEMORY CARD REQUIRED (order number: 6ES7 516-3FN01-0AB0)
- 1X SIMATIC S7-1500, DIGITAL INPUT MODULE DI 32 X DC24V, 32 CHANNELS IN GROUPS OF 16 (order number: 6ES7521-1BL00-0AB0)
- 1X SIMATIC S7-1500, DIGITAL OUTPUT MODULE DQ 32 X DC24V / 0.5A; 32 CHANNELS (order number: 6ES7 522-1BL01-0AB0)
- 1X SIMATIC S7-1500, ANALOG INPUT MODULE AI 8 X U/I/RTD/TC, 16BIT RESOLUTION 8 CHANNELS IN GROUPS OF 8 (6ES7 531-7KF00-0AB0)
- 1X SIMATIC S7-1500, ANALOG OUTPUT MODULE AQ 4 X U/I ST, 16BIT RESOLUTION, 4 CHANNELS IN GROUPS OF 4 (order number: 6ES7 532-5HD00-0AB0)

You must add the following module yourself:

- 1X SIMATIC PM 190W 120/230VAC STABILIZED POWER SUPPLY Input: 120/230 VAC output: 24 V DC / 8 A (order number: 6EP1333-4BA00)

6 Planning

Because this is a new system, a new project must be created.

The hardware for this project is already specified by the existing hardware (here: SIMATIC S7-1516F PN/DP). Therefore, a selection does not have to be made. Instead, the listed modules of the Trainer Package are detected directly. The order numbers (see Task or Table 1) can be used for checking purposes.

Module	Order number	Slot	Address area
CPU 1516F-3 PN/DP	6ES7 516-3FN01-0AB0	1	
DI 32x24VDC HF	6ES7 521-1BL00-0AB0	2	0...3
DQ 32 X DC24V / 0.5A HF	6ES7 522-1BL01-0AB0	3	0...3
AI 8 X U/I/RTD/TC, 16BIT	6ES7 531-7KF00-0AB0	4	64...79
AQ 4 X U/I ST, 16BIT	6ES7 532-5HD00-0AB0	5	64...71

Table 1: Overview of the planned configuration

The address areas must now be configured.

The power module is not automatically detected and must be manually added.

Module	Order number	Slot	Address area
PM 190W 120/230VAC	6EP1333-4BA00	0	

Table 2: Module to be manually added

As the final step, the hardware configuration must be compiled and downloaded. Any errors present can be detected during compilation and incorrect modules can be detected when the controller is started (*only possible when hardware is present and structured identically*).

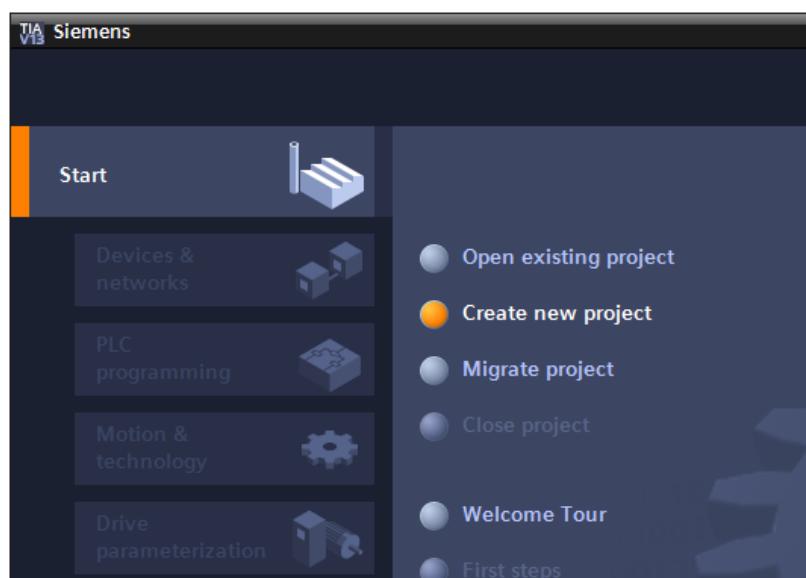
The tested project must be saved and archived.

7 Structured step-by-step instructions

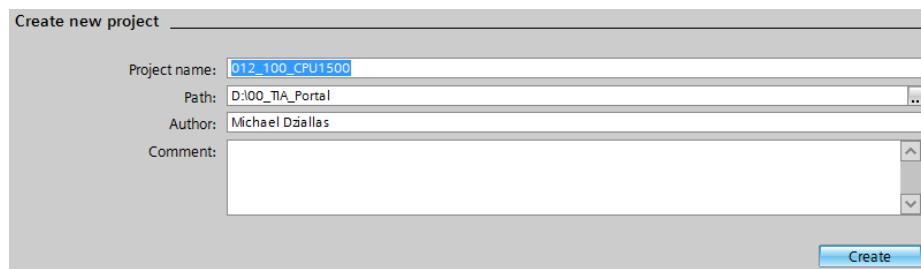
You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it is sufficient to focus on the numbered steps. Otherwise, simply follow the steps of the instructions illustrated below.

7.1 Create a new project

- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V1X)
- In the portal view under the "Start" menu, select the command → "Create new project".



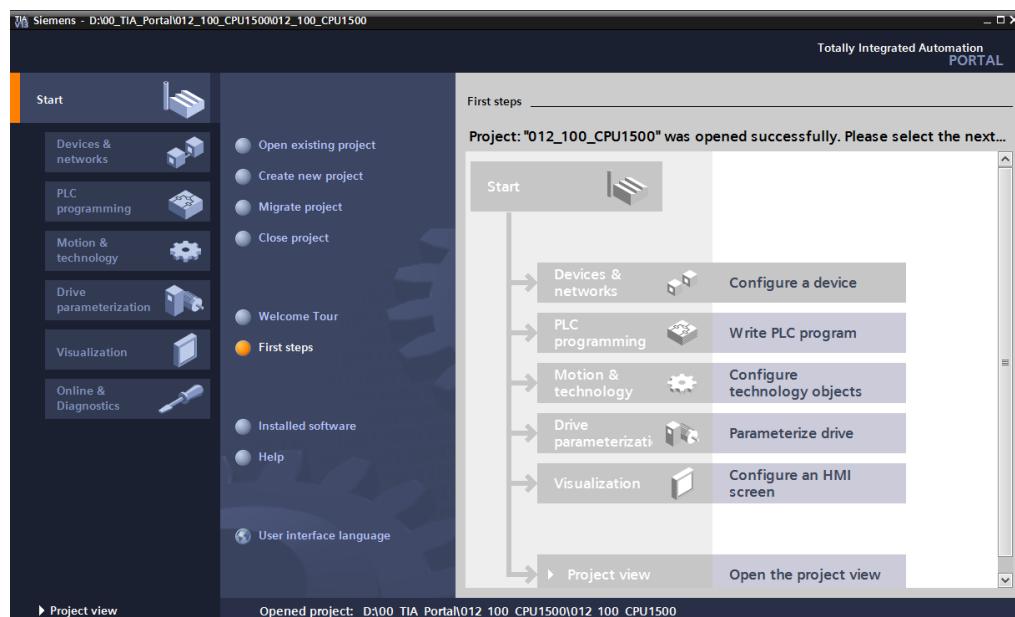
- Modify Project name, Path, Author and Comment as appropriate and click → "Create".



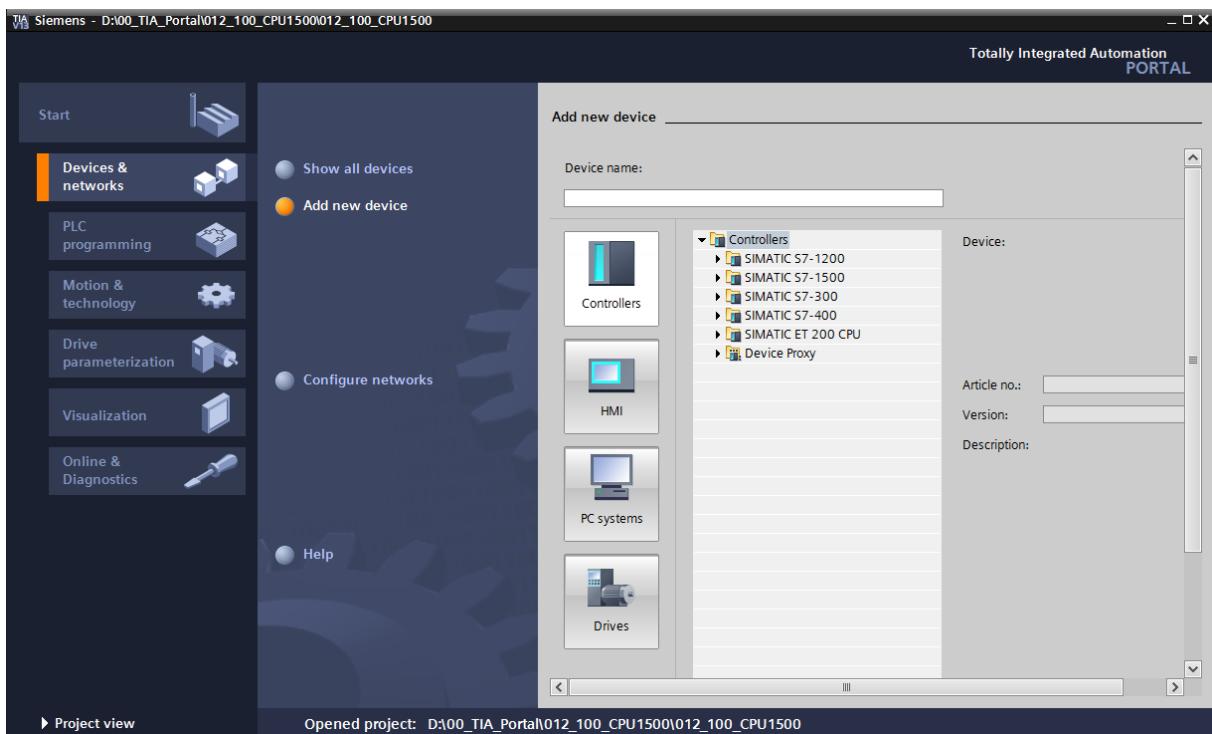
- The project will be created and opened and the menu "Start", "First steps" will open automatically.

7.2 Read the hardware of the SIMATIC S7-1500

- In the → "Start" portal, select → "First steps" → "Devices & Networks" → "Configure a device".

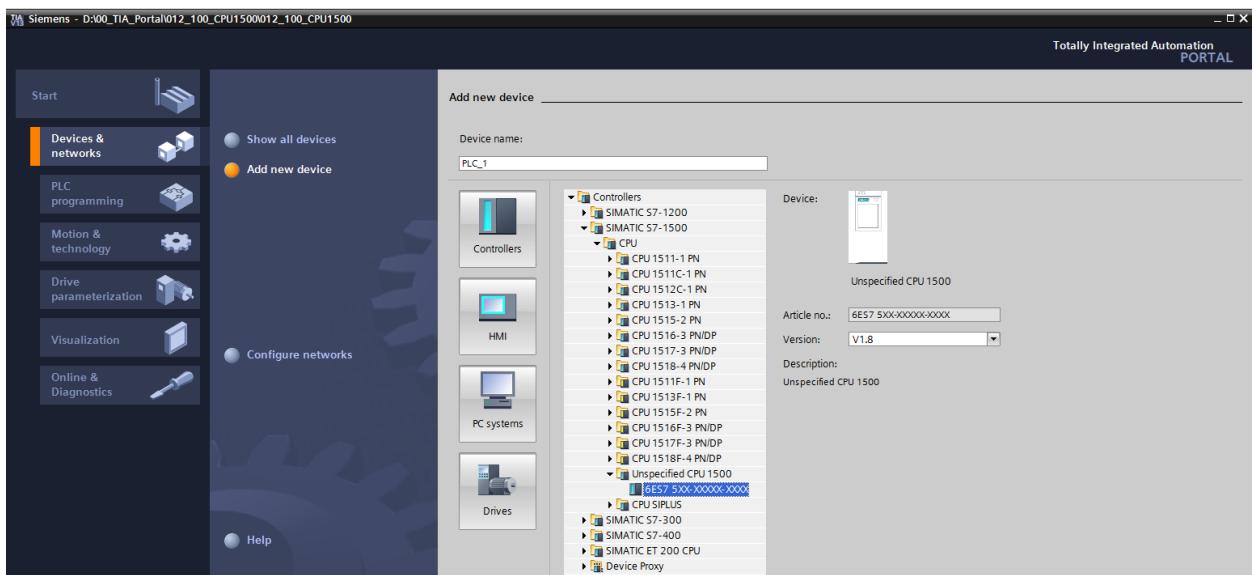


- The "Show all devices" menu opens in the "Devices & Networks" portal.
- Switch to the "Add new device" menu.



→ Create a new CPU. Use an unspecified model of the S7-1500 CPU with order number 6ES7 5XX-XXXXX-XXXX for this.

(Controllers → SIMATIC S7-1500 → CPU → Unspecified CPU 1500 → 6ES75XX-XXXXX-XXXX → V1.8)



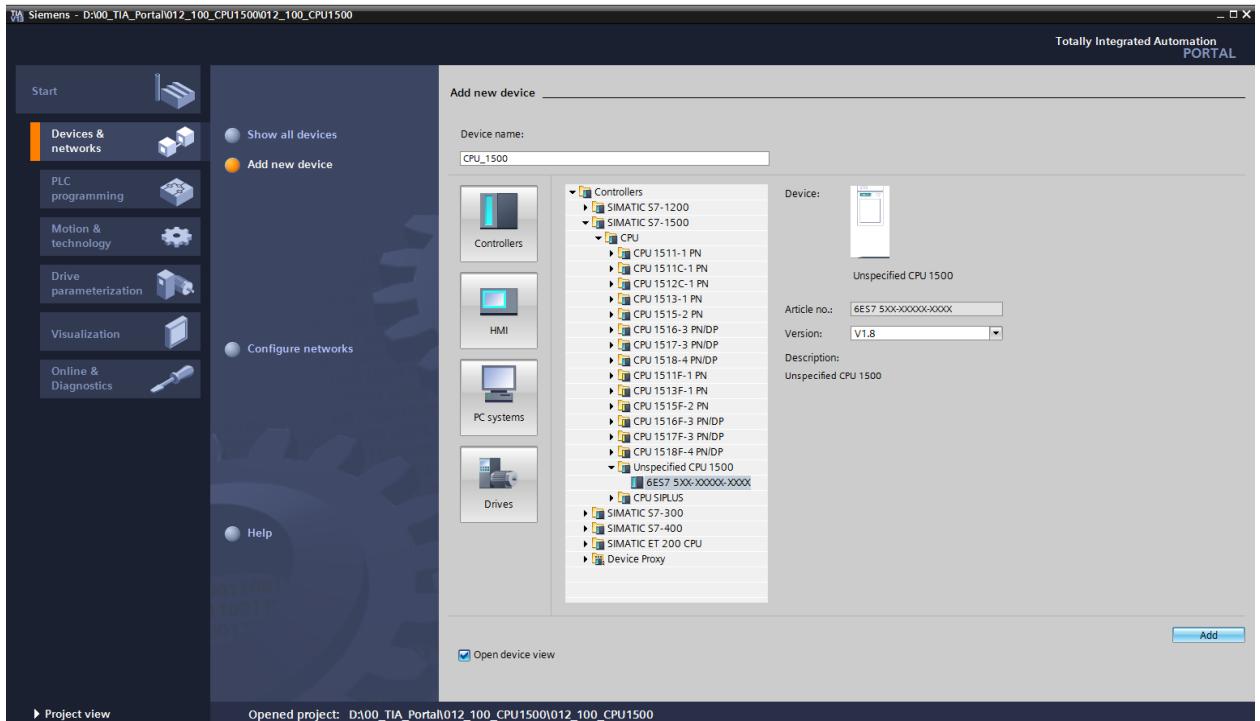
→ Assign a device name (Device name → "CPU_1500").



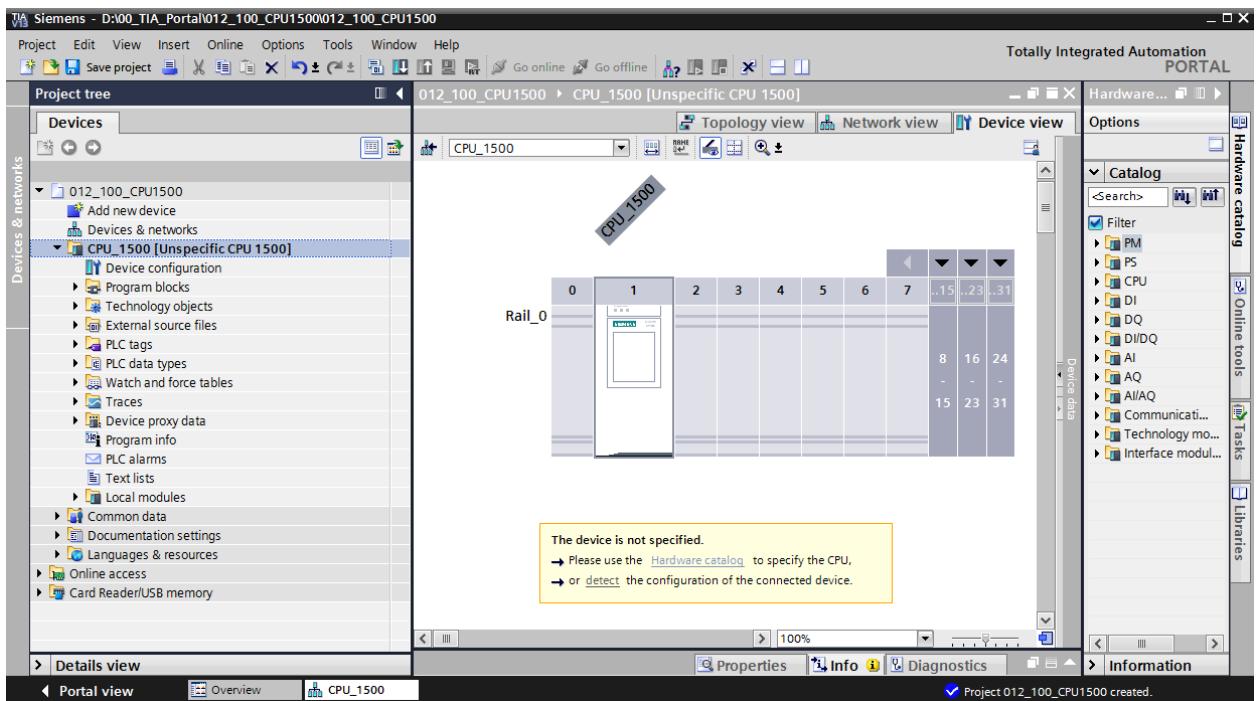
→ Select "Open device view".



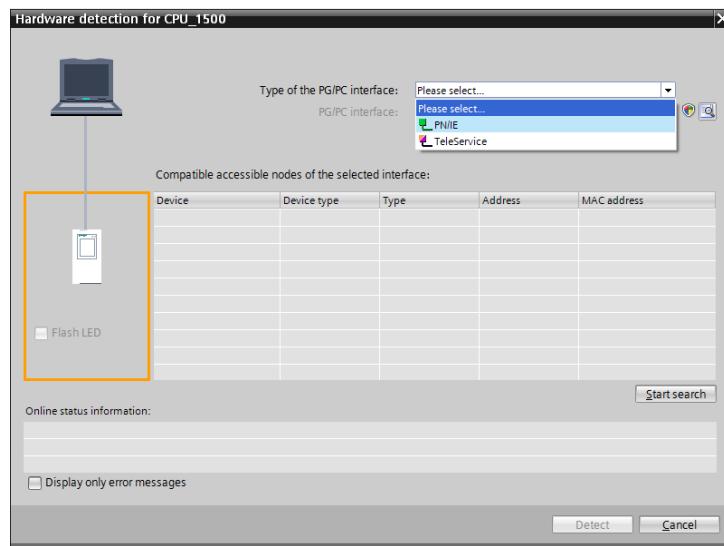
→ Click "Add".



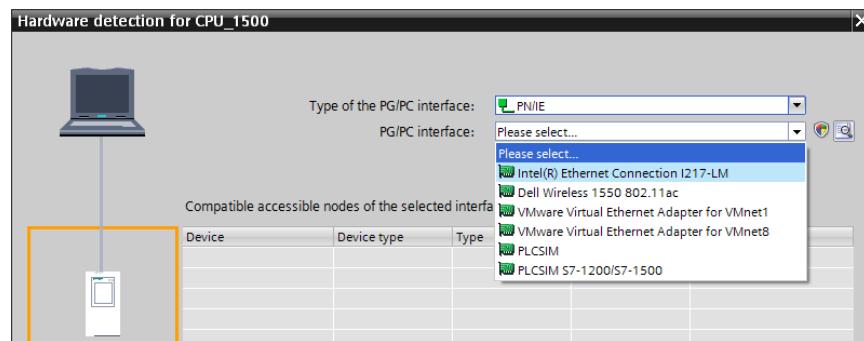
→ The TIA Portal now switches automatically to the project view and displays a notice there that this device is not specified. In order to have the hardware configuration automatically detected, start detection by clicking "detect" in the yellow information box (→ detect).



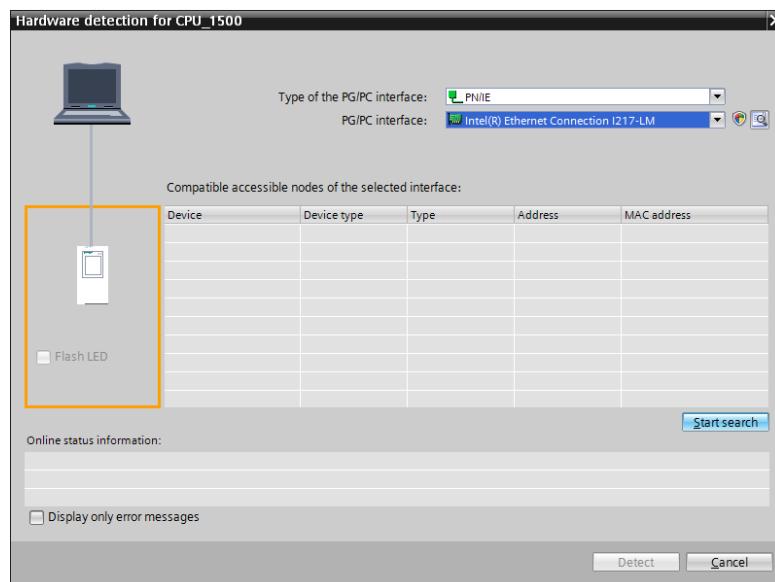
→ Select the type of your PG/PC interface. (→ Type of the PG/PC interface: PN/IE).



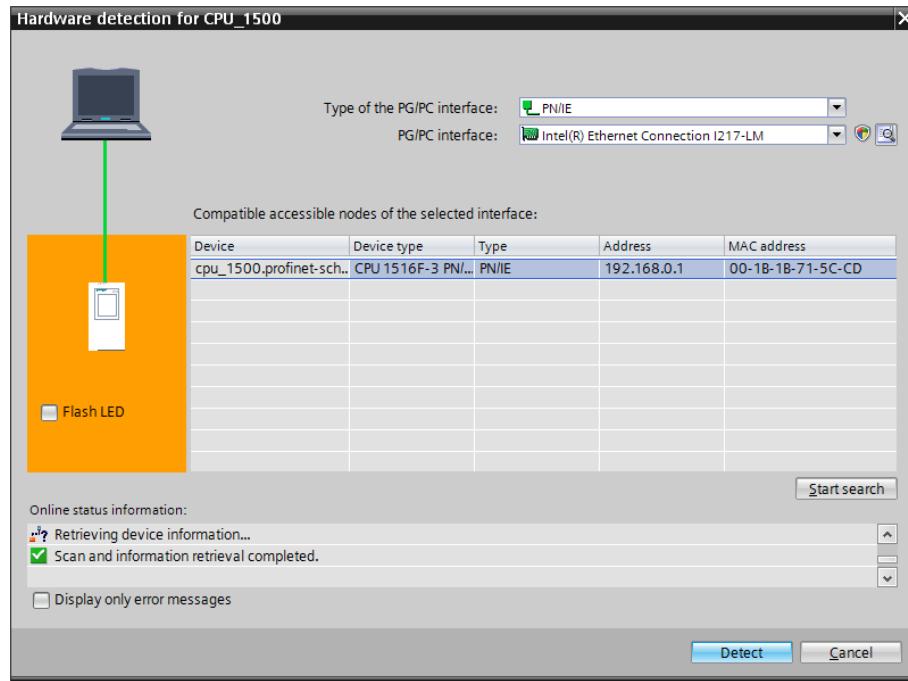
→ You can now select the network adapter you want to use to establish an Ethernet connection to the PLC. (→ PG/PC interface: Intel(R) Ethernet Connection I217-LM)



→ The search for devices in the network must be started by clicking the → **Start search** button.

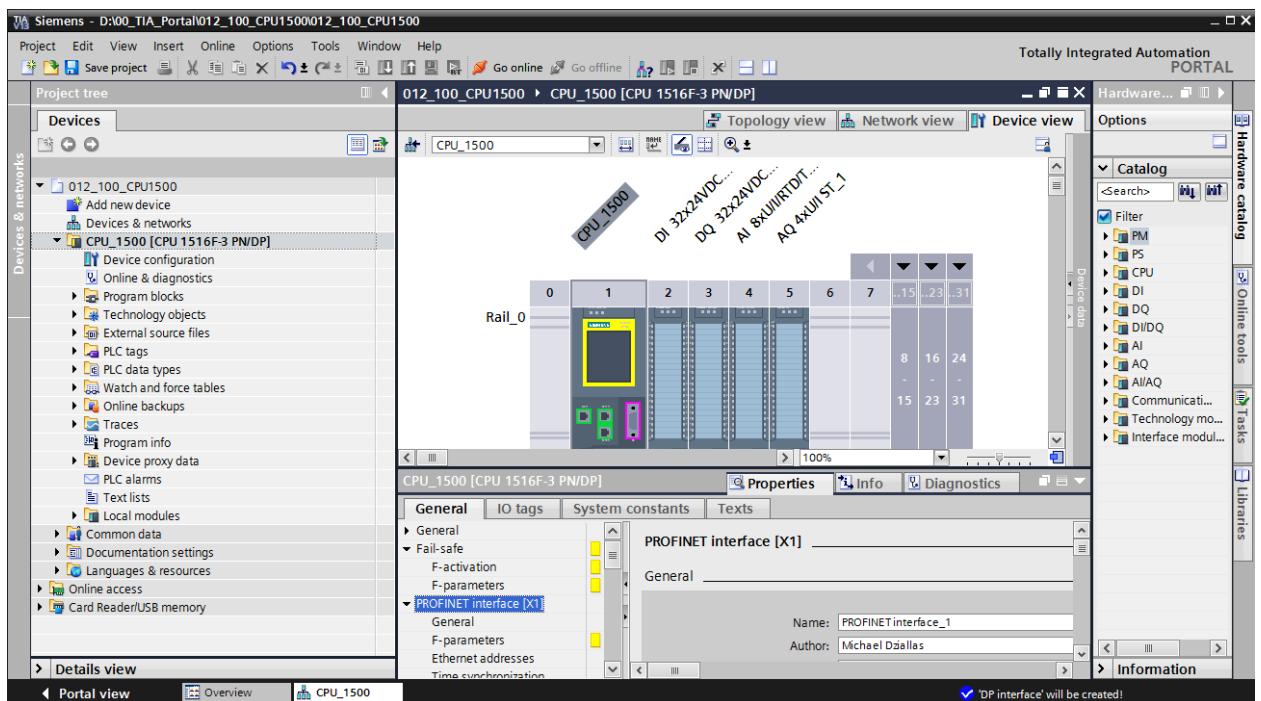


- All accessible nodes are found and listed. If you have selected the correct CPU, the corresponding CPU and all the connected modules will be detected when you click "Detect".



Note: If the list does not contain your CPU, ensure that you have selected the correct network adapter and established a connection between the laptop and CPU.

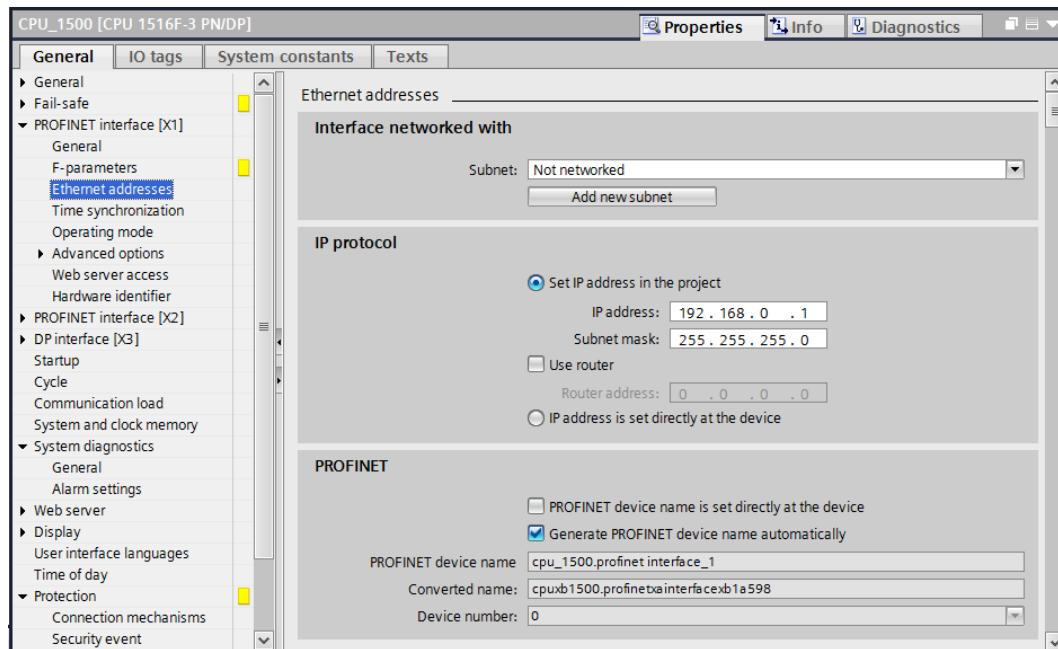
- The TIA Portal shows the complete device configuration of the selected CPU. Only the power module is lacking. This can be placed on slot 0 of the mounting rail later.



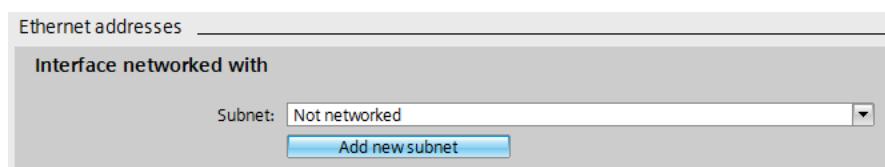
Note: You can now configure the CPU there according to your specifications. Possible settings include the PROFINET and PROFIBUS DP interfaces, startup characteristics, cycle, password protection, communication load and many others.

7.3 Configure the Ethernet interface of the CPU 1516F-3 PN/DP

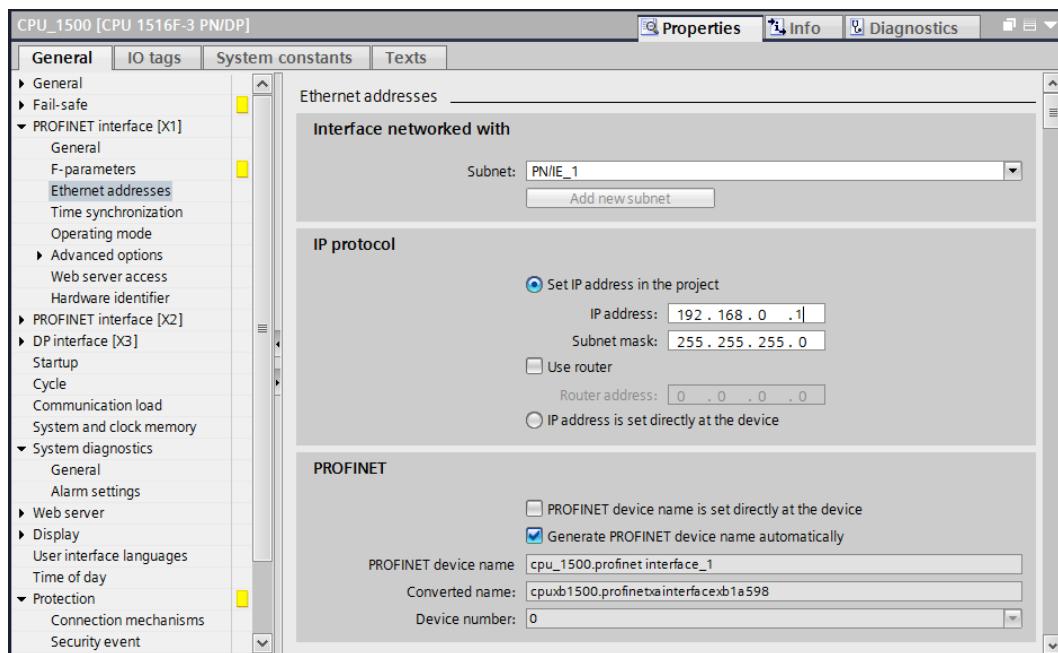
- Select the CPU with a double-click
- Under → "Properties", open the → "PROFINET-interface [X1]" menu and select the → "Ethernet addresses" entry there.



- Under "Interface connected with", only the "Not connected" entry is available.
- Add an Ethernet subnet with the → "Add new subnet" button.

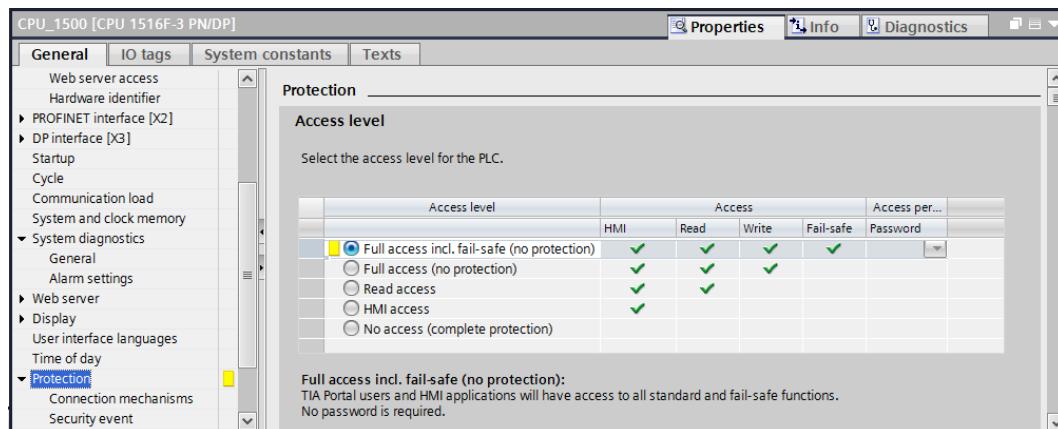


- Keep the pre-assigned "IP address" and "Subnet mask".



7.4 Configure the access level for the CPU 1516F-3 PN/DP

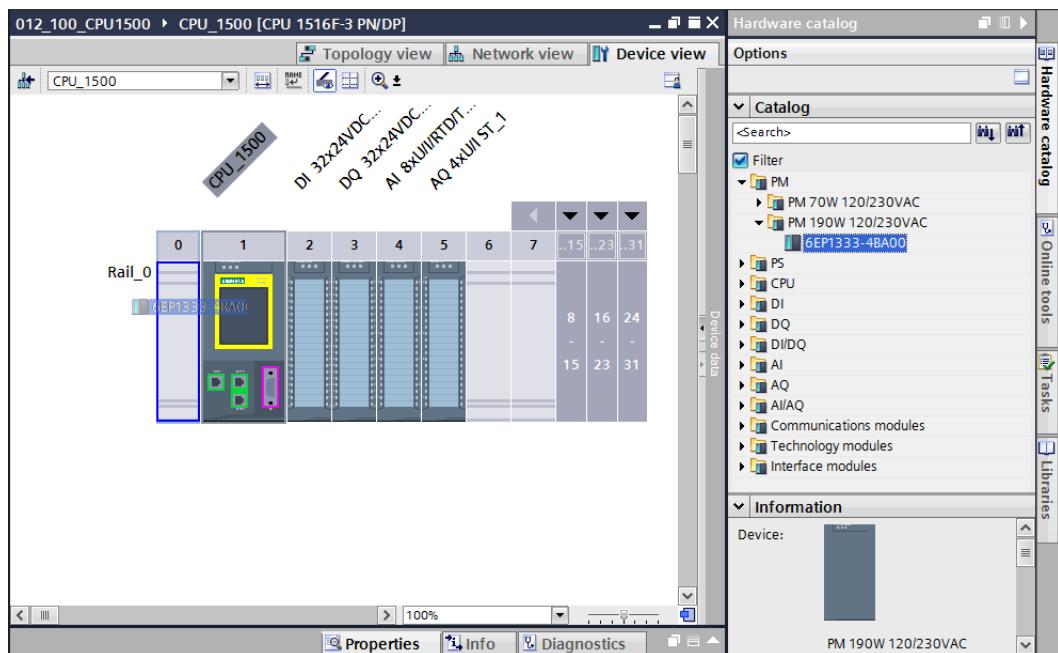
- Switch to the → "Protection" menu and select access level → "Full access incl. fail-safe (no protection)".



Note: The setting "Full access incl. fail-safe (no protection)" is recommended because a safety program is not created here and thus we also do not have to assign a password.

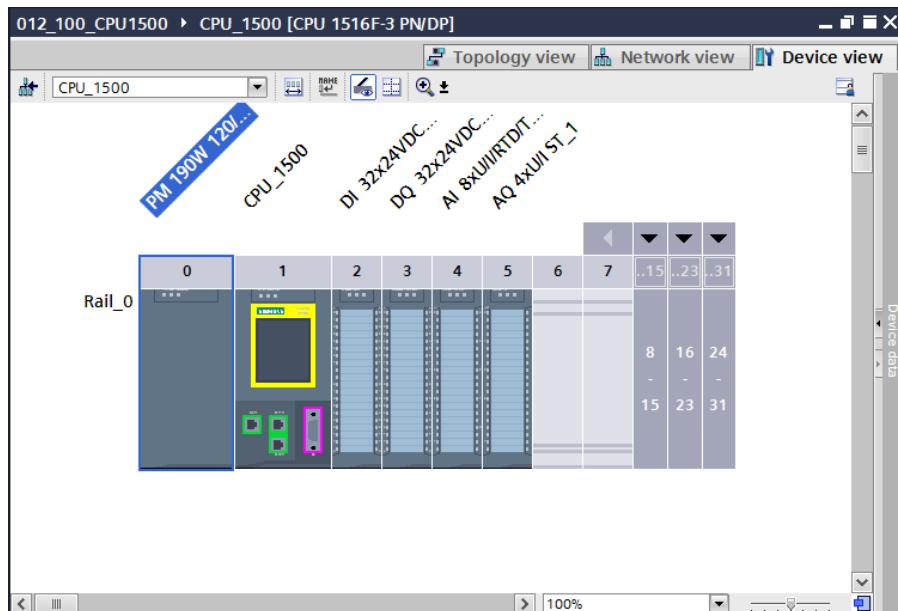
7.5 Insert power module PM 190W 120/230VAC

- Find the correct module in the hardware catalog and insert the power module into slot 0.
 (→ Hardware Catalog → PM → PM 190W 120/230VAC (order number 6EP1333-4BA00)
 → Slot 0)



Note: If a module as well as the power module is planned for one slot, it is not possible to place it at another position even in the device configuration.

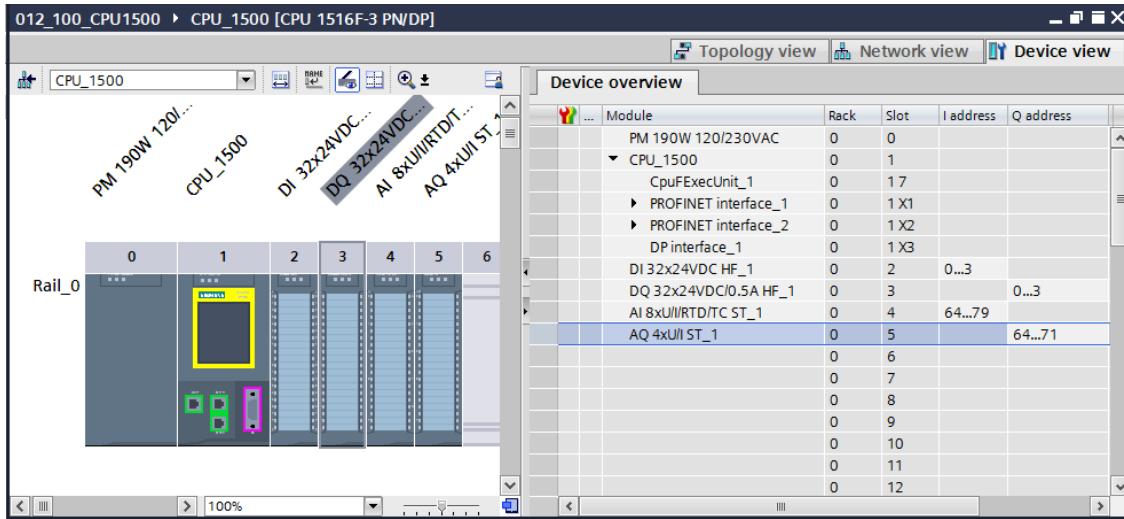
- Compare your device configuration with the following figure.



7.6 Configure the address areas of the digital input and output modules

- The next step is to check the address areas of the inputs and output cards and adapt them if necessary. DI/DO should have an address area of 0...3 and AI/AO should have an address area of 64...79 and 64...71, respectively. (→ Device overview → DI

32x24VDC HF_1 → I address: 0..3 → DQ 32x24VDC/0.5A HF_1 → Q address: 0...3 →
 AI 8xU/I/RTD/TC ST_1 → I address: 64...79 → AQ 4xU/I ST_1 → Q address: 64...71)

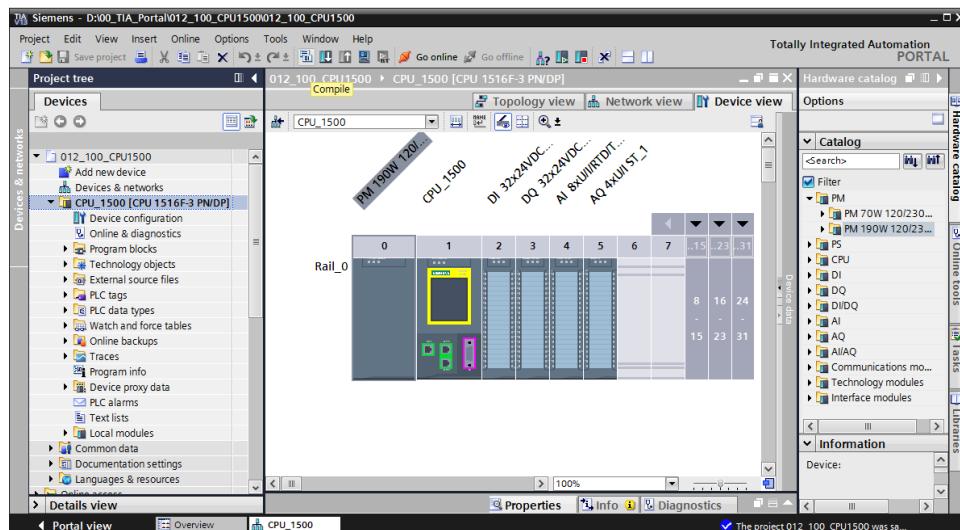


Note: To show and hide the Device overview, you must click the small arrow next to "Device data" on the right side of the hardware configuration.



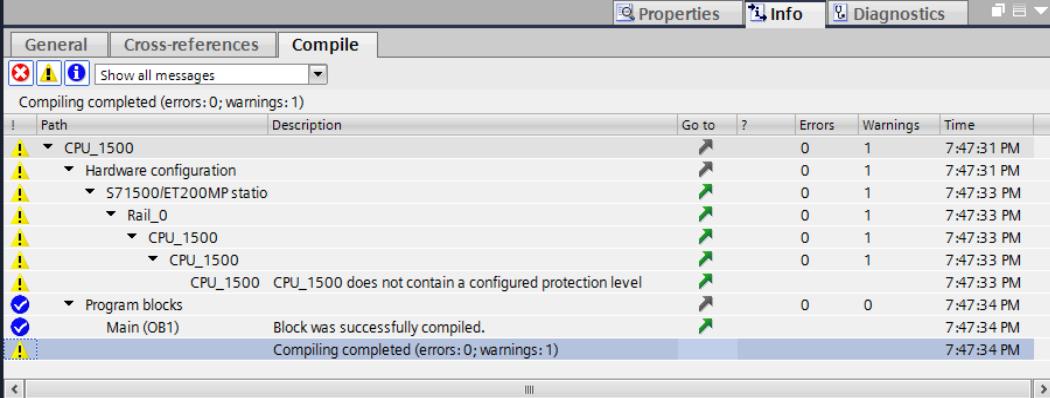
7.7 Save and compile the hardware configuration

- Before you compile the configuration, you should save your project by clicking the → button. To compile your CPU with the device configuration, first select the → "CPU_1500 [CPU1516F-3 PN/DP]" folder and click the → "Compile" icon.



Note: "Save project" should be used repeatedly when working on a project since this does not happen automatically. A prompt to save the project only occurs when the TIA Portal is closed.

- If the project was compiled without errors, you see the following screen.

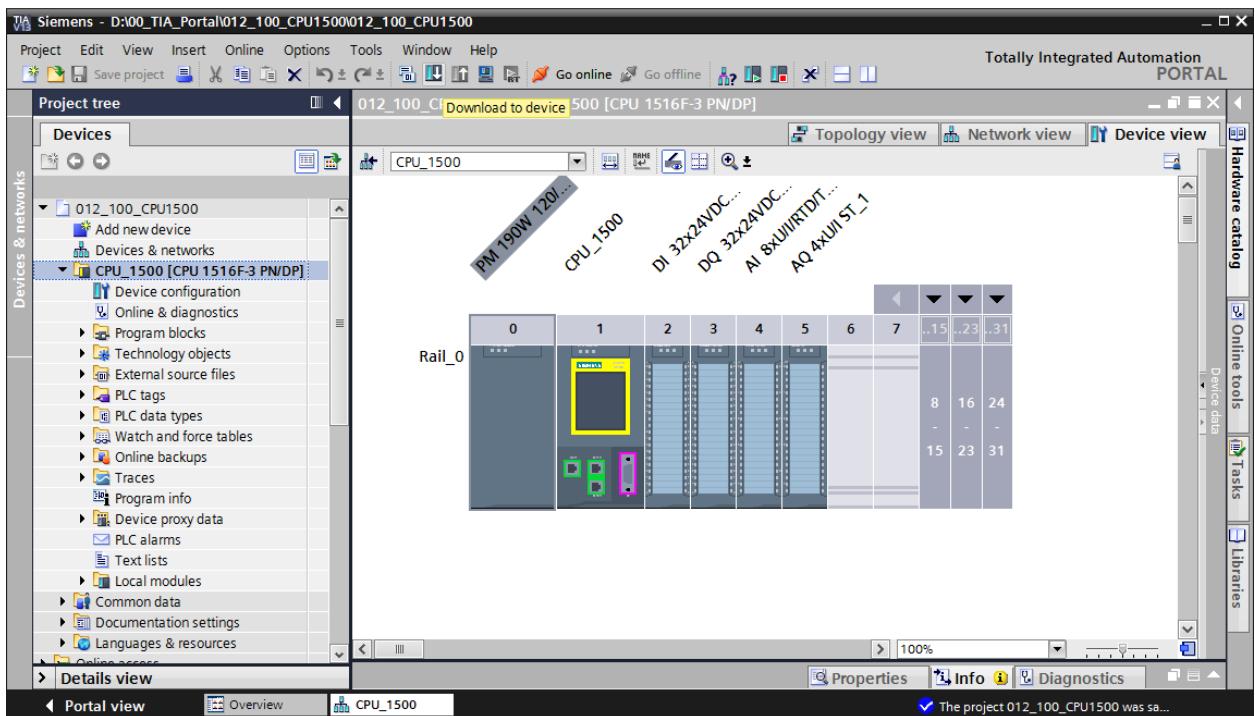


The screenshot shows the 'Compile' tab of the TIA Portal interface. The status bar at the top indicates 'Compiling completed (errors: 0; warnings: 1)'. The main area displays a table with columns for Path, Description, Go to, Errors, Warnings, and Time. The table lists the following items:

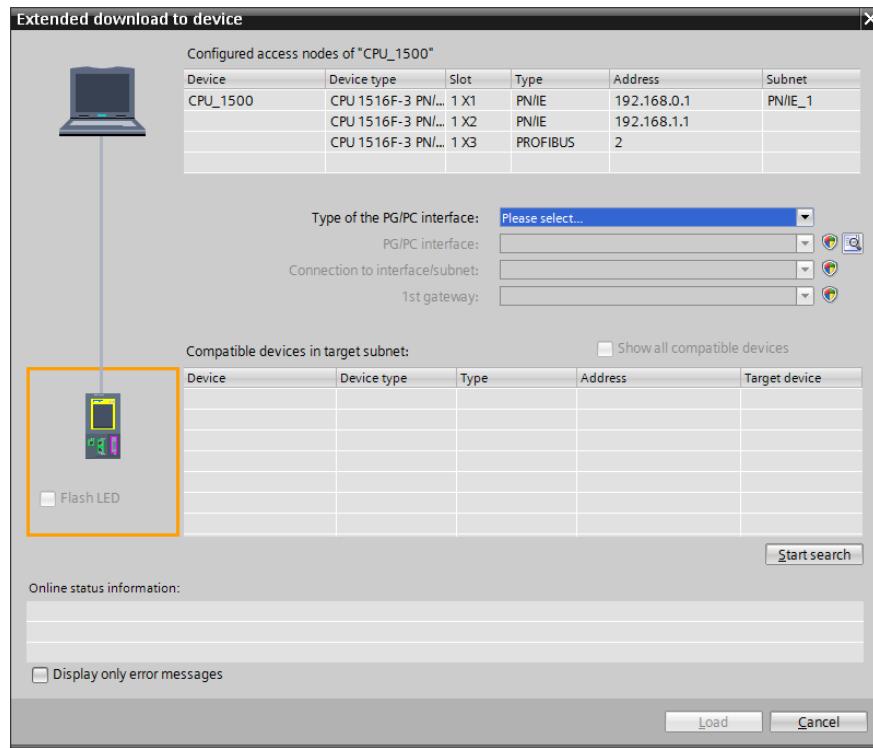
Path	Description	Go to	Errors	Warnings	Time
CPU_1500			0	1	7:47:31 PM
Hardware configuration			0	1	7:47:31 PM
S71500/ET200MP statio			0	1	7:47:33 PM
Rail_0			0	1	7:47:33 PM
CPU_1500	CPU_1500 CPU_1500 does not contain a configured protection level		0	1	7:47:33 PM
Program blocks			0	0	7:47:34 PM
Main (OB1)	Block was successfully compiled.		0	0	7:47:34 PM
	Compiling completed (errors: 0; warnings: 1)				7:47:34 PM

7.8 Download the hardware configuration to the device

- To download your entire CPU, select the → "CPU_1500 [CPU1516F-3 PN/DP]" folder and click the  → "Download to device" icon.

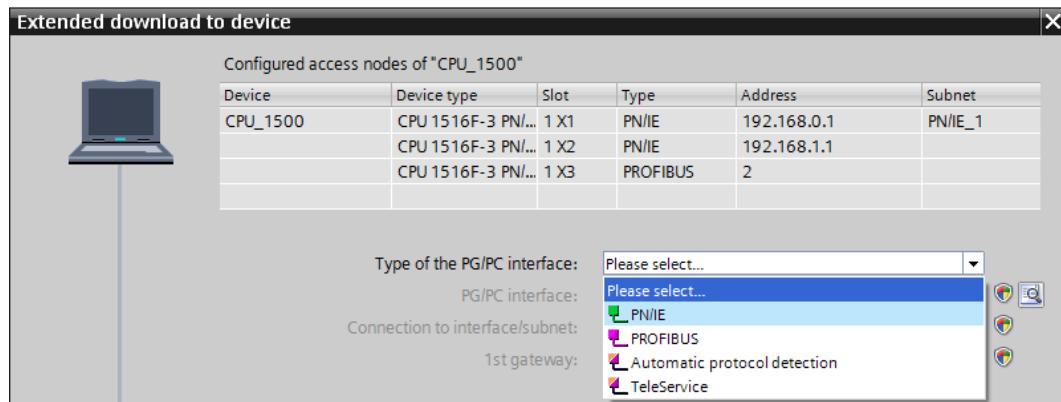


- The manager for configuring the connection properties (extended download) opens.

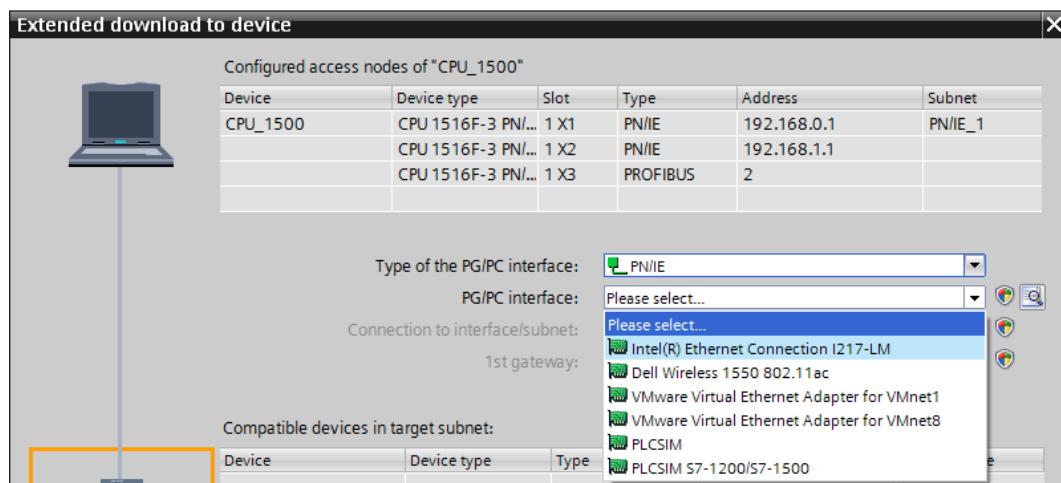


→ First, the interface must be correctly selected. This happens in three steps.

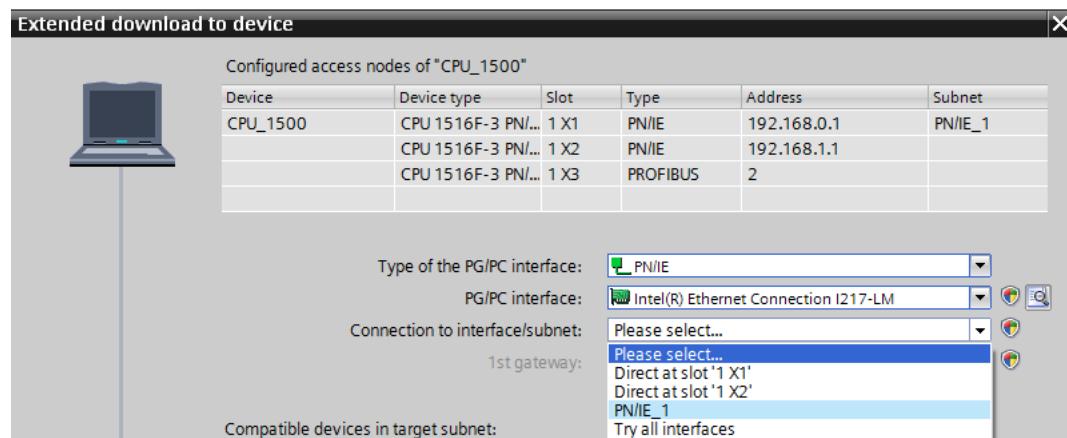
→ Type of the PG/PC interface → PN/IE



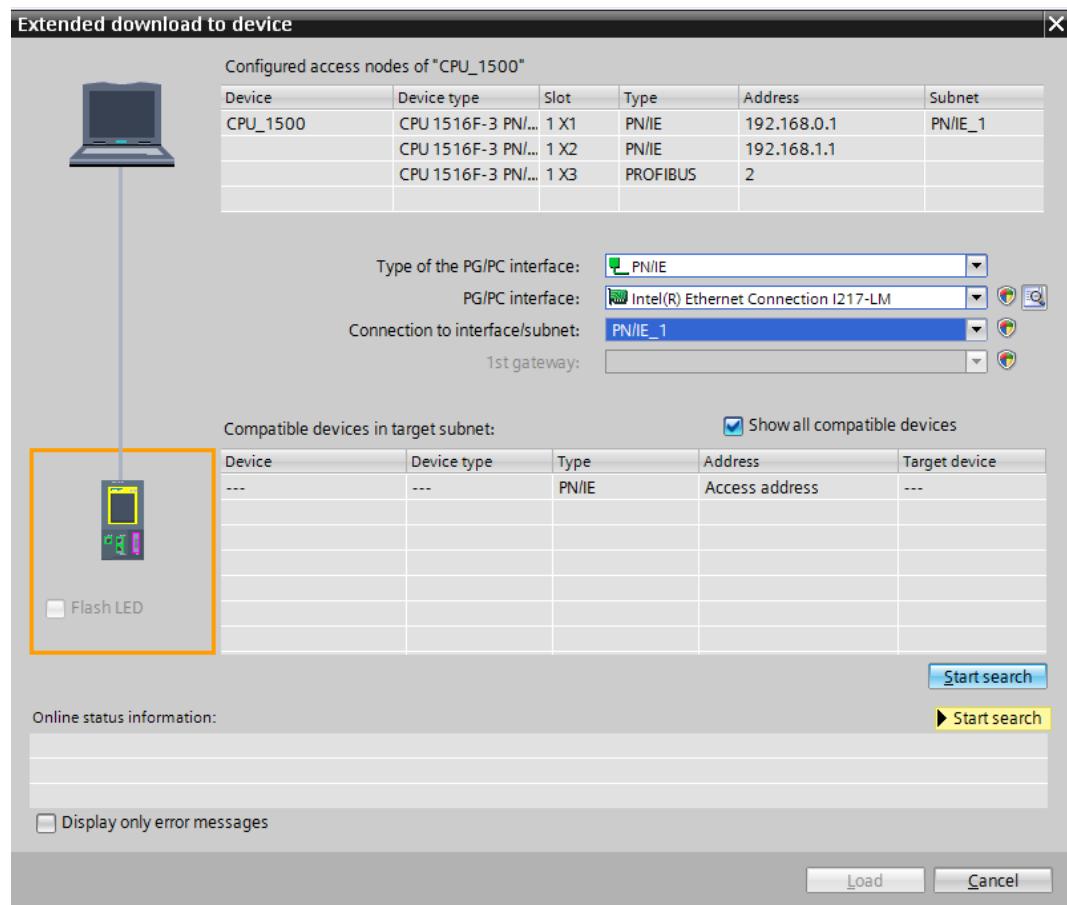
→ PG/PC interface → here: Intel(R) Ethernet Connection I217-LM



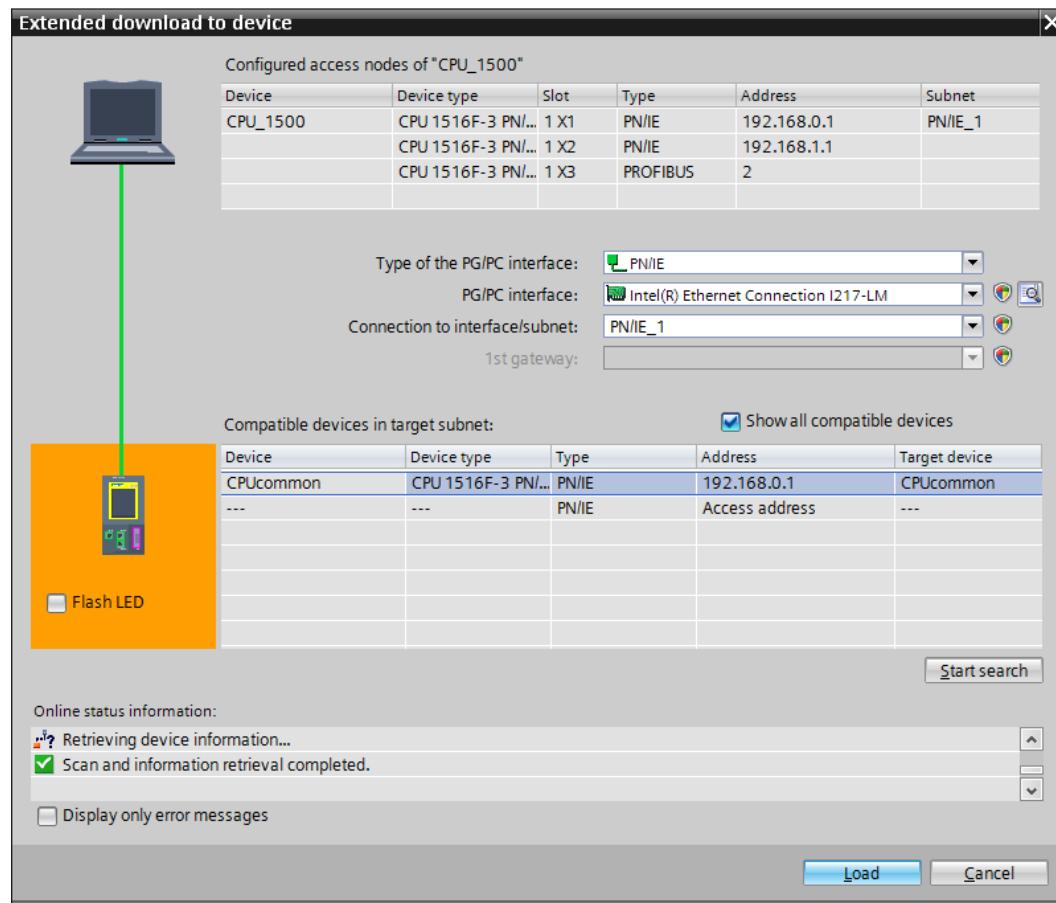
→ Connection to interface/subnet → "PN/IE_1"



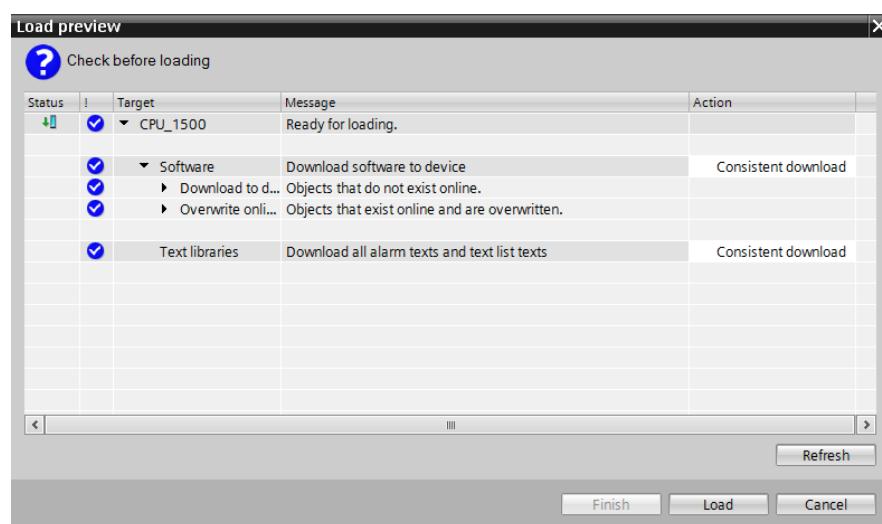
- The → "Show all compatible devices" check box must be selected. The search for devices in the network is started by clicking the → **Start search** button.



- If your CPU is shown in the "Compatible devices in target subnet" list, it must be selected. The download can then be started. (→ CPU 1516F-3 PN/DP → "Load")

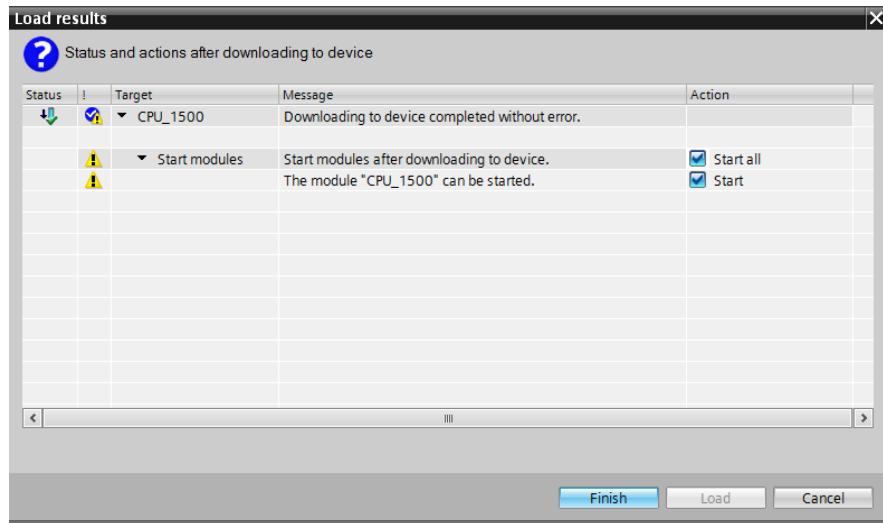


- You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → "Load".

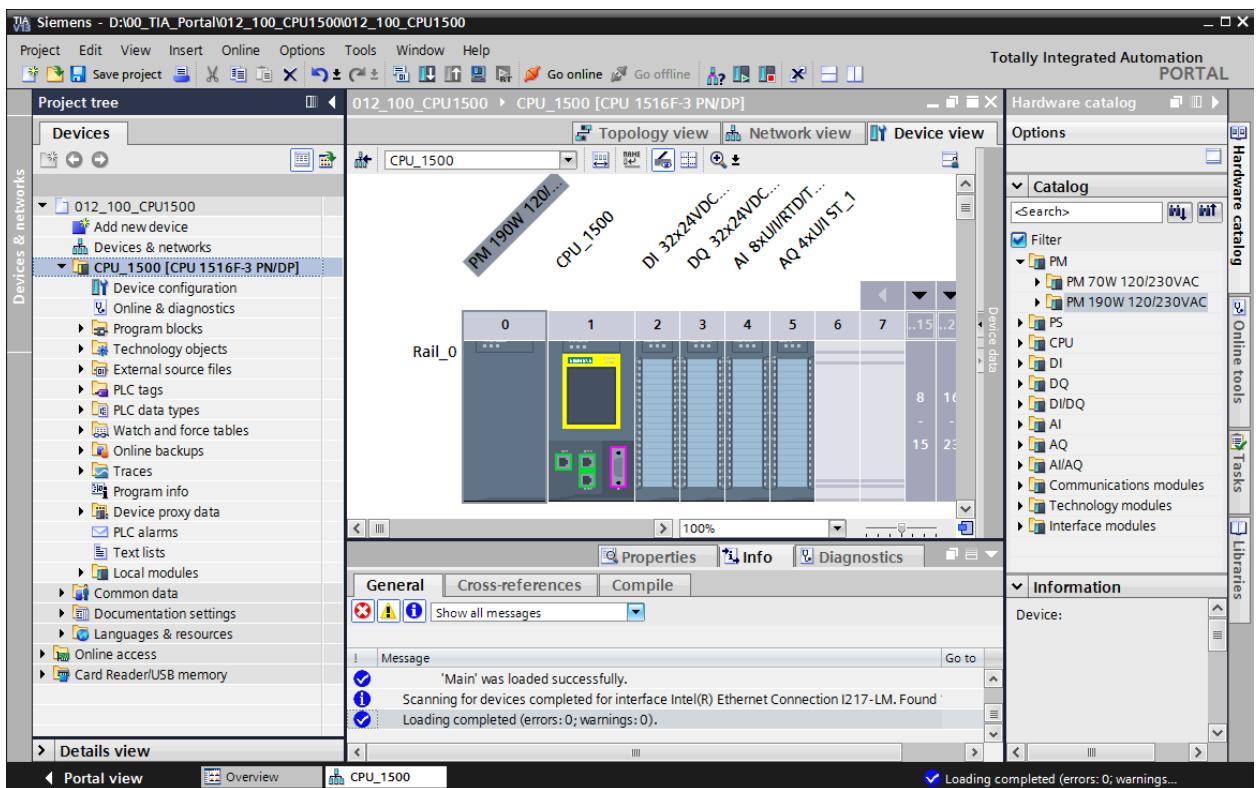


Note: The symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.

- The → "Start all" option will be selected next before the download operation can be completed with → "Finish".



- After a successful download, the project view will open again automatically. A loading report appears in the information field under "General". This can be helpful when troubleshooting an unsuccessful download.



7.9 Checklist

No.	Description	Completed
1	Project was created	
2	Slot 0: Power module with correct order number	
3	Slot 1: CPU with correct order number	
4	Slot 1: CPU with correct firmware version	
5	Slot 2: Digital input module with correct order number	
6	Slot 2: Digital input module with correct firmware version	
7	Address area of the digital input module is correct	
8	Slot 3: Digital output module with correct order number	
9	Slot 3: Digital output module with correct firmware version	
10	Slot 3: Address area of the digital output module is correct	
11	Slot 4: Analog input module with correct order number	
12	Slot 4: Analog input module with correct firmware version	
13	Slot 4: Address area of the analog input module is correct	
14	Slot 5: Analog output module with correct order number	
15	Slot 5: Analog output module with correct firmware version	
16	Slot 5: Address area of the analog output module is correct	
17	Hardware configuration was compiled without error message	
18	Hardware configuration was downloaded without error message	
19	Project was successfully archived	



SCE Training Curriculum

TIA Portal Module 003
Decentral Hardware configuration
with SIMATIC S7-1500 and
ET 200SP via PROFINET

Table of contents

1	Goal.....	80
2	Prerequisite.....	80
3	Required hardware and software.....	80
4	Theory	82
4.1	SIMATIC S7-1500 automation system	82
4.1.1	Range of modules.....	83
4.1.2	Example configuration.....	85
4.2	Operator control and display elements of the CPU 1516F-3 PN/DP.....	85
4.2.1	Front view of the CPU 1516F-3 PN/DP with integrated display	86
4.2.2	Status and error displays.....	86
4.2.3	Operator control and connection behind the front flap.....	87
4.2.4	SIMATIC Memory Card.....	87
4.2.5	Mode switch.....	88
4.2.6	Display der CPU	88
4.3	Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card	90
4.4	Configuration and operation of the SIMATIC ET 200SP	92
4.4.1	SIMATIC ET 200SP Distributed IOIO	92
4.4.2	Range of modules.....	93
4.4.3	Example configuration.....	95
4.5	STEP 7 Professional V1X (TIA Portal V1X) programming software	96
4.5.1	Project	96
4.5.2	Hardware configuration	96
4.5.3	Central and distributed automation structure	97
4.5.4	Planning the hardware	97
4.5.5	TIA Portal – Project view and portal view.....	98
4.5.6	Basic settings for the TIA Portal	99
4.5.7	Setting the IP address on the programming device.....	100
4.5.8	Setting the IP address in the CPU.....	102

4.5.9	Formatting the memory card in the CPU.....	104
4.5.10	Resetting the CPU to factory settings.....	105
4.5.11	Setting the IP address in the ET 200SP.....	106
4.5.12	Reading the firmware version of the ET 200SP	108
5	Task.....	109
6	Planning.....	110
7	Structured step-by-step instructions.....	111
7.1	Create a new project.....	111
7.2	Insert the CPU 1516F-3 PN/DP	112
7.3	Configure the Ethernet interface of the CPU 1516F-3 PN/DP	114
7.4	Configure the fail-safe operation of the CPU 1516F-3 PN/DP	116
7.5	Configure the access level for the CPU 1516F-3 PN/DP	116
7.6	Insert power module PM 190W 120/230VAC	117
7.7	Insert the ET 200SP interface module IM155-6PN HF.....	118
7.8	Configure the ET 200SP / IM 155-6PN HF.....	119
7.9	Insert the 2 digital input modules DI 8x24VDC HF.....	121
7.10	Insert the 2 digital output modules DQ 8x24VDC/0.5A HF	123
7.11	Replace components in the hardware configuration	123
7.12	Insert the server module	124
7.13	Configure the address areas DI/DQ: 0...1	125
7.14	Configuration of the potential groups of the Base Units	126
7.15	Save and compile the hardware configuration	128
7.16	Assign device name to interface module IM 155-6PN HF	129
7.17	Download the hardware configuration to the device	131
7.18	Checklist.....	136

DECENTRAL HARDWARE CONFIGURATION – SIMATIC S7-1516F PN/DP WITH ET 200SP VIA PROFINET

1 Goal

In this chapter, you will first learn how to ***create a project***. You are then shown how the ***hardware is configured***.

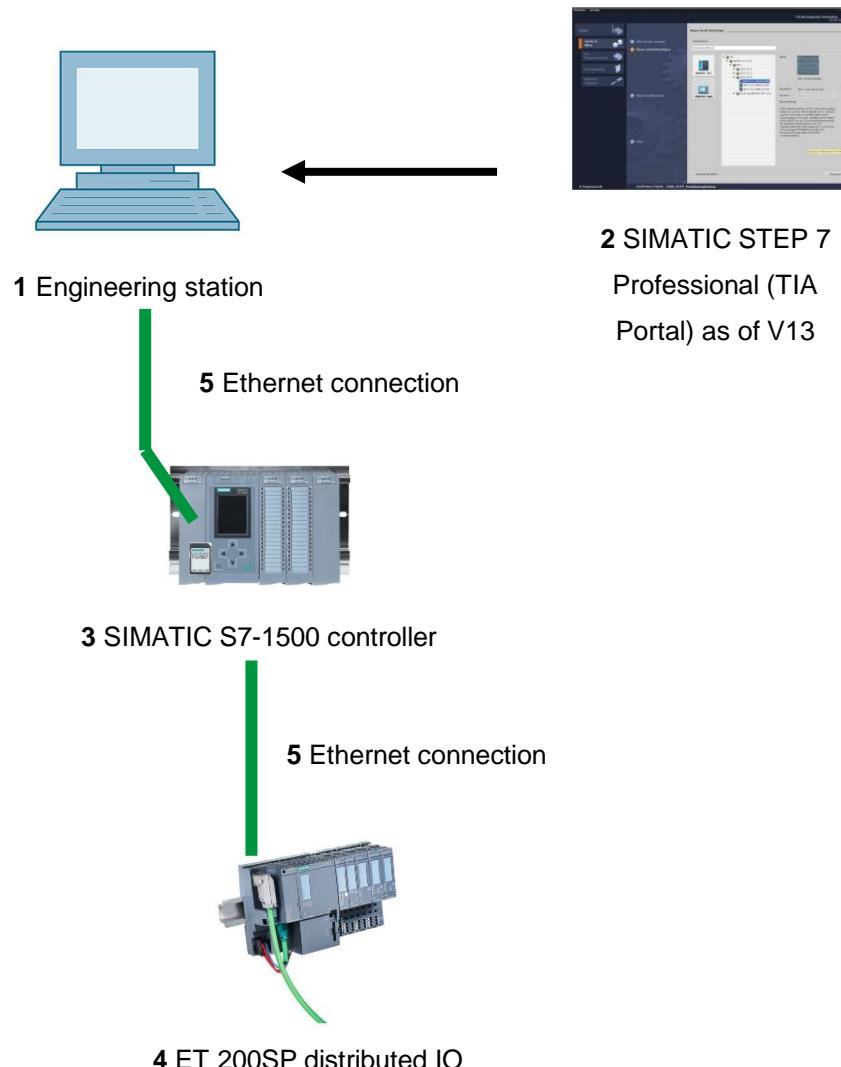
The SIMATIC S7 controllers listed in Chapter 3 can be used.

2 Prerequisite

You do not need any previous knowledge from other chapters to successfully complete this chapter.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 SIMATIC S7-1500 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card
- 4 ET 200SP distributed IO for PROFINET with 16DI/16DO and 2AI/1AO
Configuration example
 - Interface module IM155-6PN HF with Bus Adapter BA 2xRJ45
 - 2x IO module 8x digital input DI 8x24VDC HF
 - 2x IO module 8x digital output DQ 8x24VDC/0.5A HF
 - 2x IO module 2x analog input AI 2xU/I 2, 4-wire HS
 - IO module 2x analog output AQ 2xU/I HS
 - Server module
- 5 Ethernet connection between engineering station and controller and
between controller and ET 200SP distributed IO



4 Theory

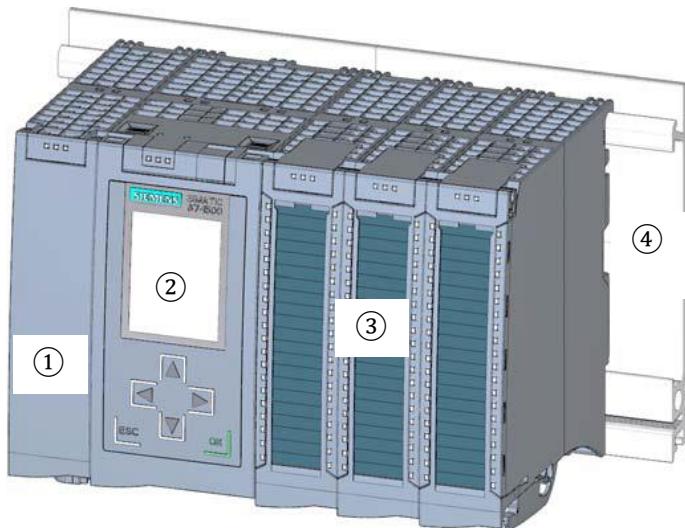
4.1 SIMATIC S7-1500 automation system

The SIMATIC S7-1500 automation system is a modular controller system for the middle to upper performance range. A comprehensive range of modules is available to optimally adapt the system to the automation task.

SIMATIC S7-1500 is the next generation of the SIMATIC S7-300 and S7-400 automation systems with the following new performance features.

- Increased system performance
- Integrated motion control functionality
- PROFINET IO IRT
- Integrated display for machine-level operation and diagnostics
- STEP 7 language innovations while maintaining proven functions

The S7-1500 controller consists of a power supply ①, a CPU with integrated display ② and input and output modules for digital and analog signals ③. The modules are mounted on a mounting rail with integrated DIN rail profile ④. If necessary, communication processors and function modules are also used for special tasks such as stepper motor control.



The programmable logic controller (PLC) uses the S7 program to monitor and control a machine or process. In doing so, the S7 program scans the IO modules via input addresses (%I) and addresses their output addresses (%Q).

The system is programmed with the STEP 7 Professional V1X software.

4.1.1 Range of modules

The SIMATIC S7-1500 is a modular automation system and offers the following range of modules:

Central processing units (CPUs) with integrated display

The CPUs have different performance capability and execute the user program. In addition, the other modules are supplied power via the backplane bus with the integrated system power supply.

Additional properties and functions of the CPU:

- Communication via Ethernet
- Communication via PROFIBUS/PROFINET
- HMI communication for HMI devices
- Web server
- Integrated technology functions (e.g. PID controller, motion control, etc.)
- System diagnostics
- Integrated security (e.g. know-how, copy, access, integrity protection)
- Integrated digital and analog inputs and outputs (Compact CPUs)



System power supply modules (PS) (rated input voltages 24 V DC to 230 V AC/DC)

with connection to the backplane bus supply the configured modules with the internal supply voltage.

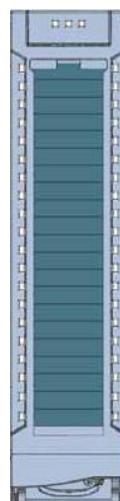


Load current supply modules (PM) (rated input voltages 120/230 V AC)

do not have a connection to the backplane bus of the S7-1500 automation system. The load current supply is used to supply 24 V DC to the system power supply of the CPU, the input and output circuits of IO modules and the sensors and actuators.

**IO modules**

for digital input (DI) / digital output (DQ) / analog input (AI) / analog output (AQ)

**Technology modules (TM)**

as incremental encoders and pulse encoders with/without direction signal.

**Communication modules (CM)**

for serial communication RS232 / RS422 / RS485, PROFIBUS and PROFINET.

SIMATIC memory card

up to a maximum of 32 GB for storing program data and for easy replacement of CPUs during maintenance.



4.1.2 Example configuration

The following configuration of an S7-1500 automation system will be used for the program example in this curriculum.



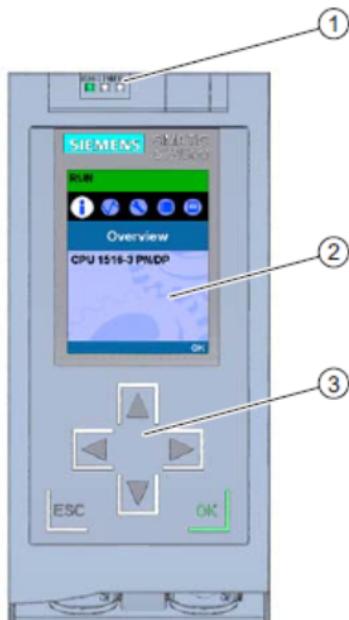
- ① Load current supply module (PM) with 120/230 V AC, 50 Hz / 60 Hz, 190 W input and 24 V DC / 8 A output
- ② Central processing unit CPU 1516F-3 PN/DP with integrated PROFIBUS and PROFINET interfaces

4.2 Operator control and display elements of the CPU 1516F-3 PN/DP

The figure below shows the operator control and display elements of a CPU 1516F-3 PN/DP.

The arrangement and number of elements differ from this figure for other CPUs.

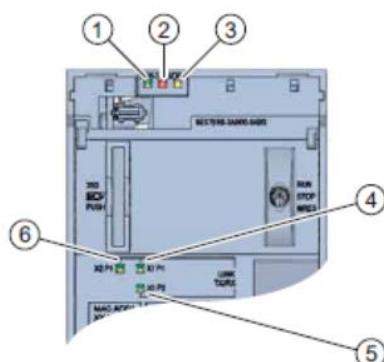
4.2.1 Front view of the CPU 1516F-3 PN/DP with integrated display



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display
- ③ Control keys

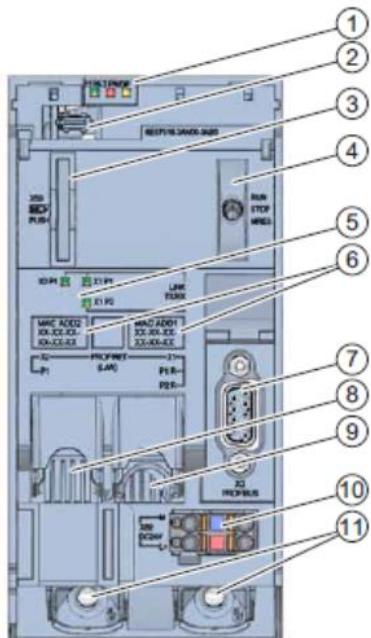
4.2.2 Status and error displays

The CPU comes with the following LED displays:



- ① RUN/STOP LED (yellow/green LED)
- ② ERROR LED (red LED)
- ③ MAINT LED (yellow LED)
- ④ LINK RX/TX LED for port X1 P1 (yellow/green LED)
- ⑤ LINK RX/TX LED for port X1 P2 (yellow/green LED)
- ⑥ LINK RX/TX LED for port X2 P1 (yellow/green LED)

4.2.3 Operator control and connection behind the front flap



- ① LED displays for the current operating mode and diagnostic status of the CPU
- ② Display connection
- ③ Slot for the SIMATIC memory card
- ④ Mode switch
- ⑤ LED displays for the 3 ports of the PROFINET interfaces X1 and X2
- ⑥ MAC addresses of the interfaces
- ⑦ PROFIBUS interface (X3)
- ⑧ PROFINET interface (X2) with 1 port
- ⑨ PROFINET interface (X1) with 2-port switch
- ⑩ Connection for supply voltage
- ⑪ Fastening screws

Note: The front flap with the display can be removed and inserted during operation.

Note: The PROFINET field devices (ET 200SP in this case) should be connected to the PROFINET interface(X1) with the 2 ports.

4.2.4 SIMATIC Memory Card

A SIMATIC Micro Memory Card is used as the memory module for the CPUs. This is a preformatted memory card that is compatible with the Windows file system. It is available with various storage capacities and can be used for the following purposes:

- Transportable data storage medium
- Program card
- Firmware update card

The MMC **must** be inserted to operate the CPU as the CPUs have no integrated load memory. A commercially available SD card reader is needed to write/read the SIMATIC memory card with the

programming device or PC. This allows files to be copied directly to the SIMATIC memory card using Windows Explorer, for example.

Note: It is recommended that the SIMATIC memory card only be removed or inserted when the CPU is in the POWER OFF state.

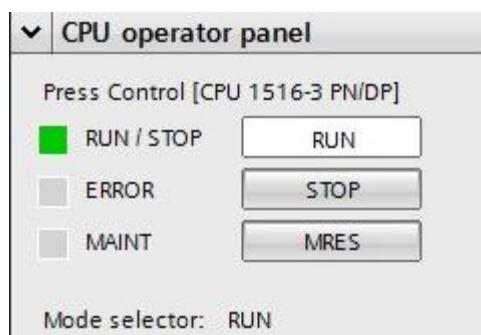
4.2.5 Mode switch

The mode switch allows you to set the operating mode of the CPU. The mode switch is designed as a toggle switch with 3 switch positions.

Position	Meaning	Explanation
RUN	RUN mode	The CPU processes the user program.
STOP	STOP mode	The CPU is not executing the user program.
MRES	Memory reset	Position for CPU memory reset.

You can also use the button on the CPU operator panel of the STEP 7 Professional V1X software in Online & Diagnostics to switch the operating mode (**STOP** or **RUN**).

The operator panel also contains an **MRES** button for performing a memory reset and displays the status LEDs of the CPU.



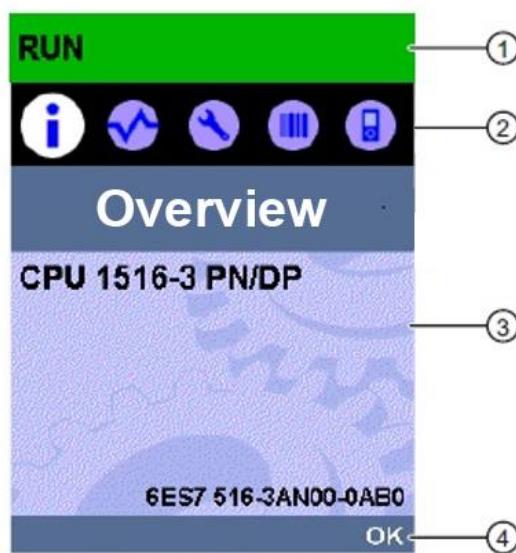
4.2.6 Display der CPU

The S7-1500 CPU has a front flap with a display and control keys. Control data and status data can be displayed in various menus on the display and numerous settings can be configured. You use the control keys to navigate through the menus.

The display of the CPU offers the following functions:

- 6 different display languages can be selected.
- Diagnostic messages are displayed in plain text.
- The interface settings can be changed locally.
- Password assignment for display operation is possible through the TIA Portal.

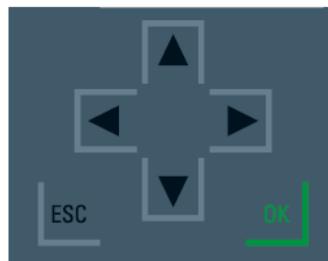
View of the display of an S7-1500:



- ① CPU status information
- ② Submenu name
- ③ Information display field
- ④ Navigation aid, e.g. OK/ESC or the page number

Control keys of the display

- Four arrow keys: "up", "down", "left", "right"
- An ESC key
- An OK key



Functions of the "OK" and "ESC" keys

- For menu commands in which an input can be made:
 - OK → valid access to the menu command, confirmation of input and exit from editing mode
 - ESC → restoration of original content (which means changes are not saved) and exit from editing mode
- For menu commands in which no input can be made:
 - OK → to next submenu command
 - ESC → back to previous menu command

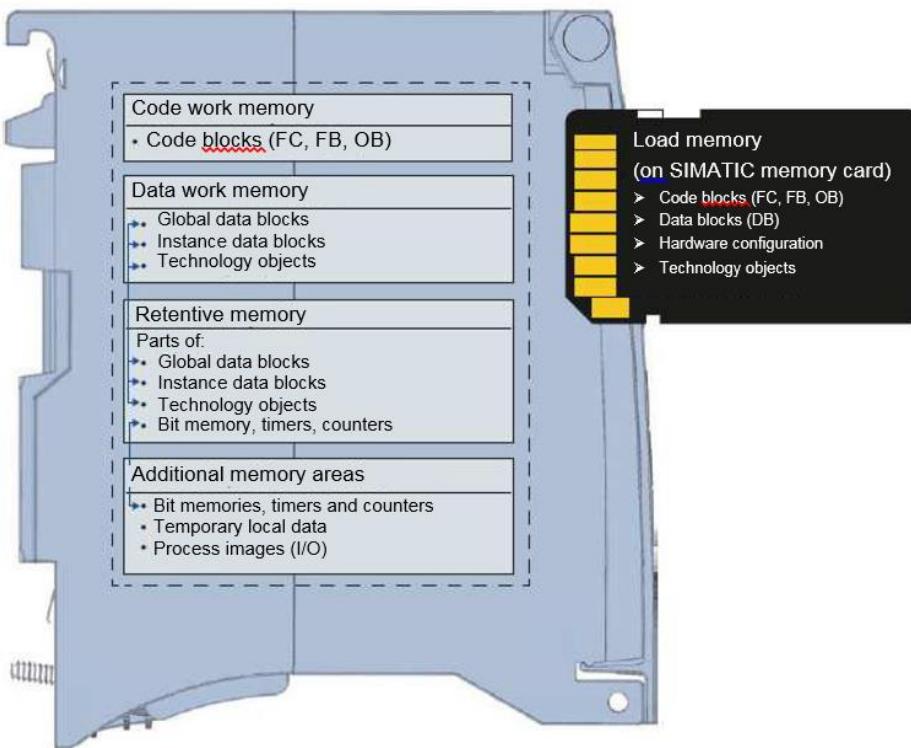
Available submenus of the display:

Main menu commands	Meaning	Explanation
	Overview	The "Overview" menu contains information about the properties of the CPU.
	Diagnostics	The "Diagnostics" menu contains information about diagnostic messages, the diagnostic description and the indication of interrupts. There is also information about the network properties of each interface of the CPU.
	Settings	In the "Settings" menu, the IP addresses of the CPU are assigned, the date, time, time zones, operating modes (RUN/STOP) and protection levels are set, the CPU memory is reset and its factory settings are restored and the status of firmware updates is displayed.
	Modules	The "Modules" menu contains information about the modules that are used in your configuration. The modules can be used as central or distributed modules. Distributed modules are connected to the CPU via PROFINET and/or PROFIBUS. You have the option here to set the IP addresses for a CPU.
	Display	In the "Display" menu, settings are made for all aspects of the display, such as the language setting, brightness setting and Energy-saving mode. (Energy-saving darkens the display. Standby mode switches off the display.)

4.3 Memory areas of the CPU 1516F-3 PN/DP and the SIMATIC memory card

The following figure shows the memory areas of the CPU and the load memory on the SIMATIC memory card.

In addition to the load memory, other data can be loaded onto the SIMATIC memory card using Windows Explorer. This includes recipes, data logs, project backups and additional documentation for the program.



Load memory

Load memory is non-volatile memory for code blocks, data blocks, technology objects and the hardware configuration. When these objects are downloaded to the CPU, they are first stored in the load memory. This memory is located on the SIMATIC memory card.

Work memory

Work memory is volatile memory that contains the code and data blocks. The work memory is integrated into the CPU and cannot be expanded. In S7-1500 CPUs, the work memory is divided into two areas:

- Code work memory:
 - The code work memory contains runtime-relevant parts of the program code.
- Data work memory:
 - The data work memory contains the runtime-relevant parts of the data blocks and technology objects.

At the operating mode transitions from POWER ON to startup and from STOP to startup, tags of global data blocks, instance data blocks and technology objects are initialized with their start values. Retentive tags retain their actual values that were saved in the retentive memory.

Retentive memory

Retentive memory is non-volatile memory for saving certain data in the event of power failure. The tags and operand areas that have been defined as retentive are saved in the retentive memory. This data is retained beyond power-off or power failure.

All other program tags are set to their start values at the operating mode transitions from POWER ON to startup and from STOP to startup.

The content of retentive memory is deleted by the following actions:

- Memory reset

- Reset to factory settings

Note: Certain tags of technology objects are also stored in the retentive memory. These tags are not deleted by a memory reset.

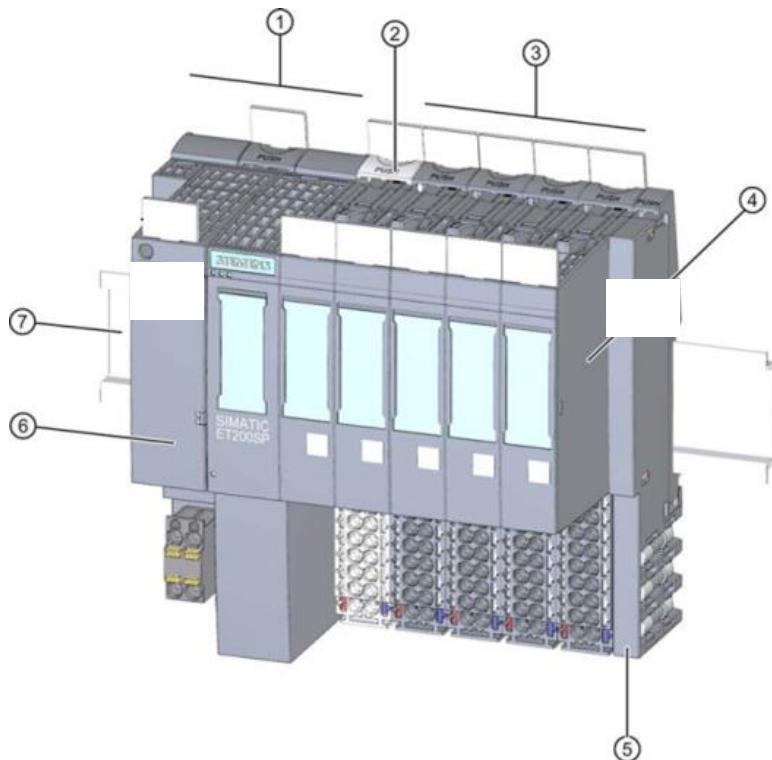
4.4 Configuration and operation of the SIMATIC ET 200SP

4.4.1 SIMATIC ET 200SP Distributed IOIO

SIMATIC ET 200SP Distributed IO is a modular distributed IO system for connecting process signals to a central automation system such as SIMATIC S7-1500. A comprehensive range of modules is available to optimally adapt the system to the automation task.

Distributed IO are often used when signals must be transmitted over a larger distance and the associated wiring overhead is too high. The signals can be collected locally at the remote location and connected to the central controller via a bus system. In the case of the ET 200SP system, devices can be connected via PROFINET or PROFIBUS.

The ET 200SP distributed IO is mounted on a standard mounting rail ⑦ and is composed of an interface module ① with bus adapter ⑥, up to 32/64 IO modules ④ inserted on BaseUnits ②, ③ and a terminating server module ⑤.



The distributed IO provide inputs and outputs locally for the process connection, which can be read and written by the central processing unit via a bus protocol. In doing so, the IO modules are scanned in the S7 program using input addresses (%I) and addressed using output addresses (%Q) in the usual way.

Because the distributed IO are an extension of the central controller, the system is also configured with the STEP 7 Professional V1X software.

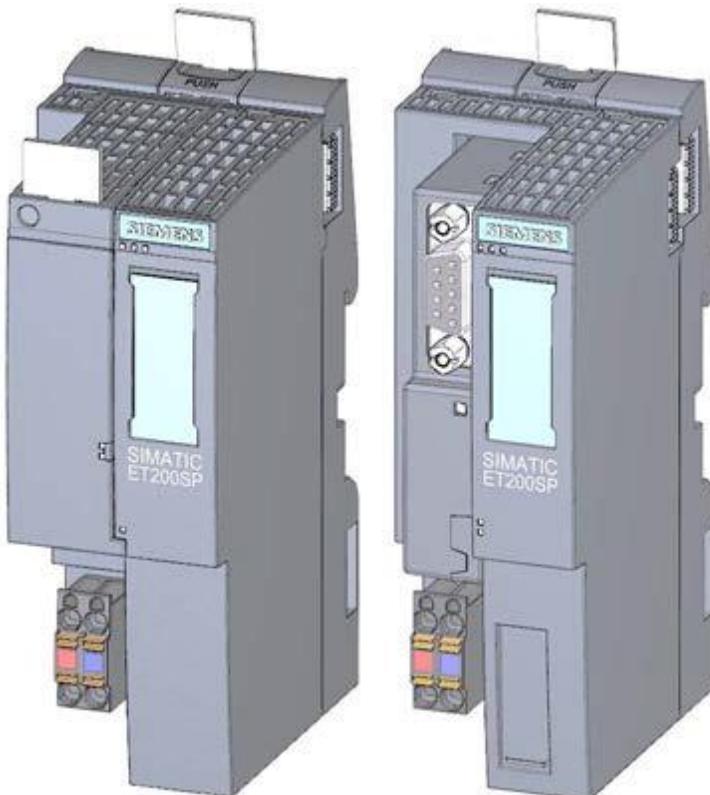
4.4.2 Range of modules

The SIMATIC ET 200SP is a modular distributed IO system and offers the following range of modules:

Interface modules with pluggable bus adapter

for connection of distributed IO to a central processing unit.

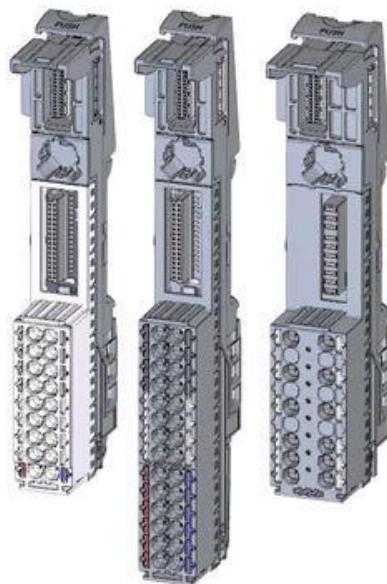
The bus adapter enables selection of any type of connection system. Interface modules have their own power supply, which is not connected via the backplane bus.



BaseUnits

as universal basic modules for electrical and mechanical connection of the IO modules.

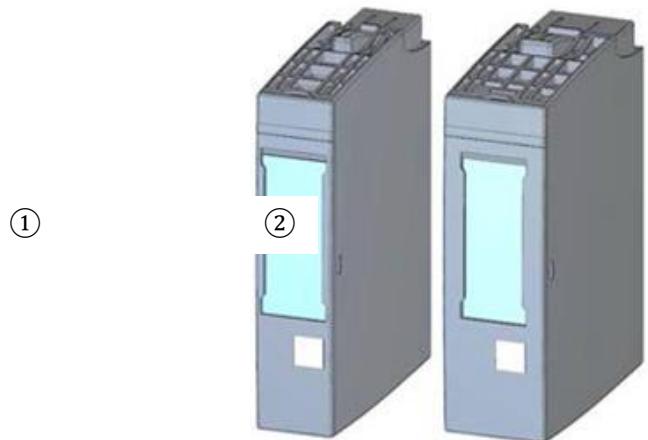
They are available in a light version BU..D that opens a new potential group for the voltage supply over the backplane bus and in a dark version BU..B that continues the potential group. At least one light BaseUnit BU..D must always be used in order to supply at least one potential group with voltage. The IO modules are inserted on the BaseUnits.



IO modules

for digital input (DI) / digital output (DO) / analog input (AI) / analog output (AO).

They are available in versions for 24 V DC ① and 400 V AC ②



Communication modules (CM)

for a point-to-point (PtP) connection ① or connection to the IO-Link ② and AS-i ③ communication systems.



Server module

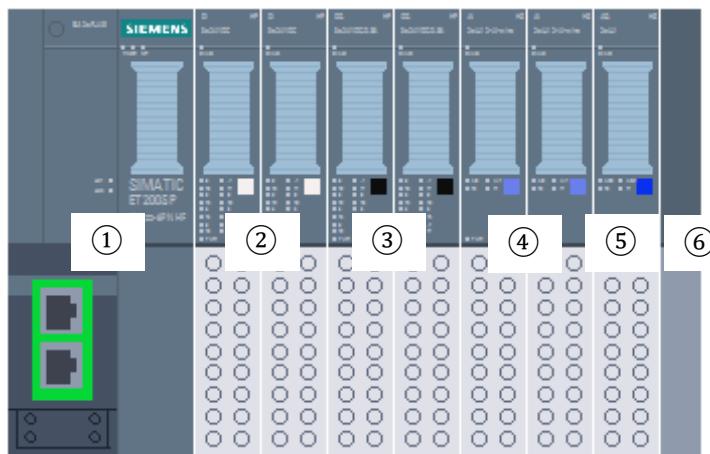
as a termination for the configuration of the ET 200SP system.

It can be used as a holder for 3 spare fuses. It serves as a terminating resistor for the backplane bus and is thus a mandatory component.



4.4.3 Example configuration

The following configuration of an ET 200SP distributed IO system will be used for the program example in this curriculum.



- ① Interface module IM155-6PN HF with Bus Adapter BA 2xRJ45
- ② IO module 8x digital input DI 8x24VDC HF (2x)
- ③ IO module 8x digital output DQ 8x24VDC/0.5A HF (2x)
- ④ IO module 2x analog input AI 2xU/I 2, 4-wire HS (2x)
- ⑤ IO module 2x analog output AQ 2xU/I HS (1x)
- ⑥ Server module

4.5 STEP 7 Professional V1X (TIA Portal V1X) programming software

STEP 7 Professional V1X (TIA Portal V1X) software is the programming tool for the following automation systems:

- SIMATIC S7-1500
- SIMATIC S7-1200
- SIMATIC S7-300
- SIMATIC S7-400
- SIMATIC WinAC

STEP 7 Professional V1X provides the following functions for plant automation:

- Configuration and parameter assignment of the hardware
- Specification of the communication
- Programming
- Testing, commissioning and servicing with operational/diagnostic functions
- Documentation
- Creation of visualizations for SIMATIC Basic Panels using the integrated WinCC Basic software
- Visualization solutions for PCs and other panels can also be created with other WinCC software packages

Support is provided for all functions through detailed online help.

4.5.1 Project

To implement a solution for an automation and visualization task, you create a project in the TIA Portal. A project in the TIA Portal contains the configuration data for the configuration and internetworking of devices as well as the programs and the configuration of the visualization.

4.5.2 Hardware configuration

The *hardware configuration* includes the configuration of the devices, consisting of the hardware of the automation system, the intelligent field devices and the hardware for visualization. The configuration of the networks specifies the communication between the various hardware components. The individual hardware components are *inserted in the hardware configuration* from catalogs.

The hardware of automation systems comprises controllers (CPUs), signal modules for input and output signals (SMs) and communication processors, and interface modules (CP, IM). Power supply and voltage supply modules (PS, PM) are also available to supply the modules.

The signal modules and intelligent field devices connect the input and output data of the process to be automated and visualized to the automation system.

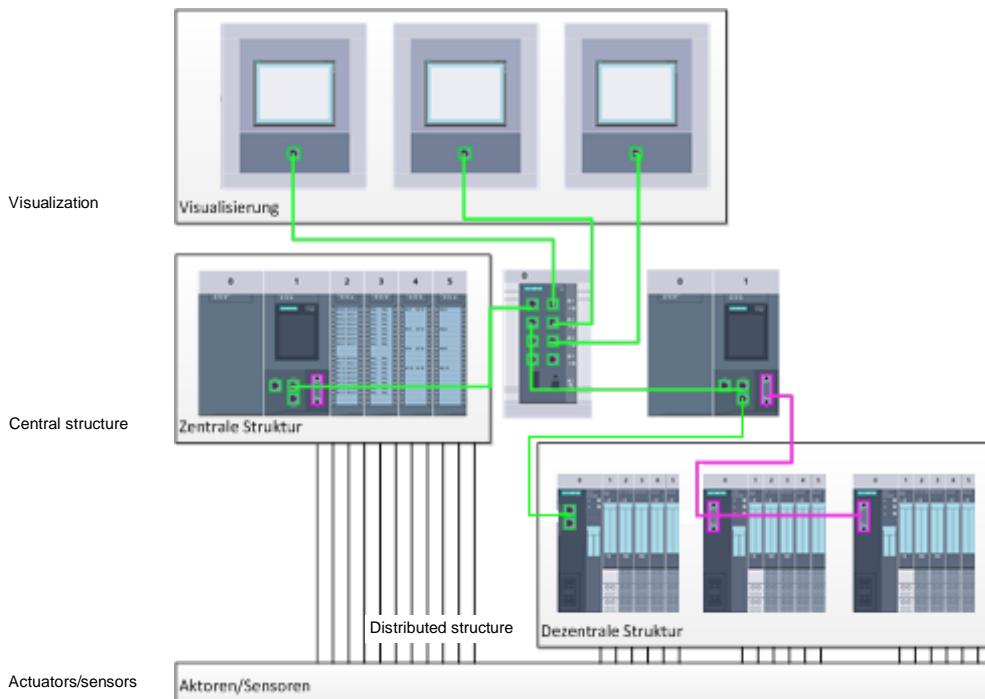


Figure 1: Example of hardware configuration with central and distributed structures

The hardware configuration enables the downloading of automation and visualization solutions to the automation system and access to the connected signal modules by the controller.

4.5.3 Central and distributed automation structure

Figure 1 shows an automation structure that contains both central and distributed structures.

In central structures, the input and output signals of the process are transmitted by way of conventional wiring to the signal modules, which are connected directly to the controller.

Conventional wiring refers to the connection of sensors and actuators using 2-wire or 4-wire cables.

The distributed structure is the predominant structure used today. Here, the sensors and actuators are wired conventionally only as far as the signal modules of the field devices. The signal transmission from the field devices to the controller is implemented using an industrial communication system.

Both classic fieldbuses such as PROFIBUS, Modbus and Foundation Fieldbus as well as Ethernet-based communication systems such as PROFINET can be used as the industrial communication system.

In addition, intelligent field devices in which stand-alone programs run can also be connected via the communication system. These programs can also be created with the TIA Portal.

4.5.4 Planning the hardware

Before you can configure the hardware, you must plan it (hardware planning). In general, you begin by selecting which controllers are needed and how many. Next you select the communication modules and signal modules. The selection of signal modules is based on the number and type of

inputs and outputs needed. As the final step, a power supply that ensures that the necessary power is supplied must be selected for each controller or field device.

The functionality required and the ambient conditions are of vital importance for planning the hardware configuration. For example, the temperature range in the application area sometimes limits the devices available for selection. Fail-safe operation might be another requirement, for example.

The [TIA Selection Tool](#) (Select automation technology → TIA Selection Tool and follow the instructions) provides you support. Note: TIA Selection Tool requires Java.

Note for online research: If more than one manual is available, you should look for the description "Device Manual", "Product Manual" or simply "Manual" (as opposed to "Function Manual", "List Manual", "System Manual", etc.) in order to find the device specifications.

4.5.5 TIA Portal – Project view and portal view

The TIA Portal has two important views. When started, the portal view appears by default. This view makes getting started easier, especially for beginning users.

The portal view provides a task-oriented view of the tools for working on the project. Here, you can quickly decide what you want to do and open the tool for the task at hand. If necessary, a change to the project view takes place automatically for the selected task.

Figure 2 shows the portal view. At the bottom left, there is an option to switch between this view and the project view.

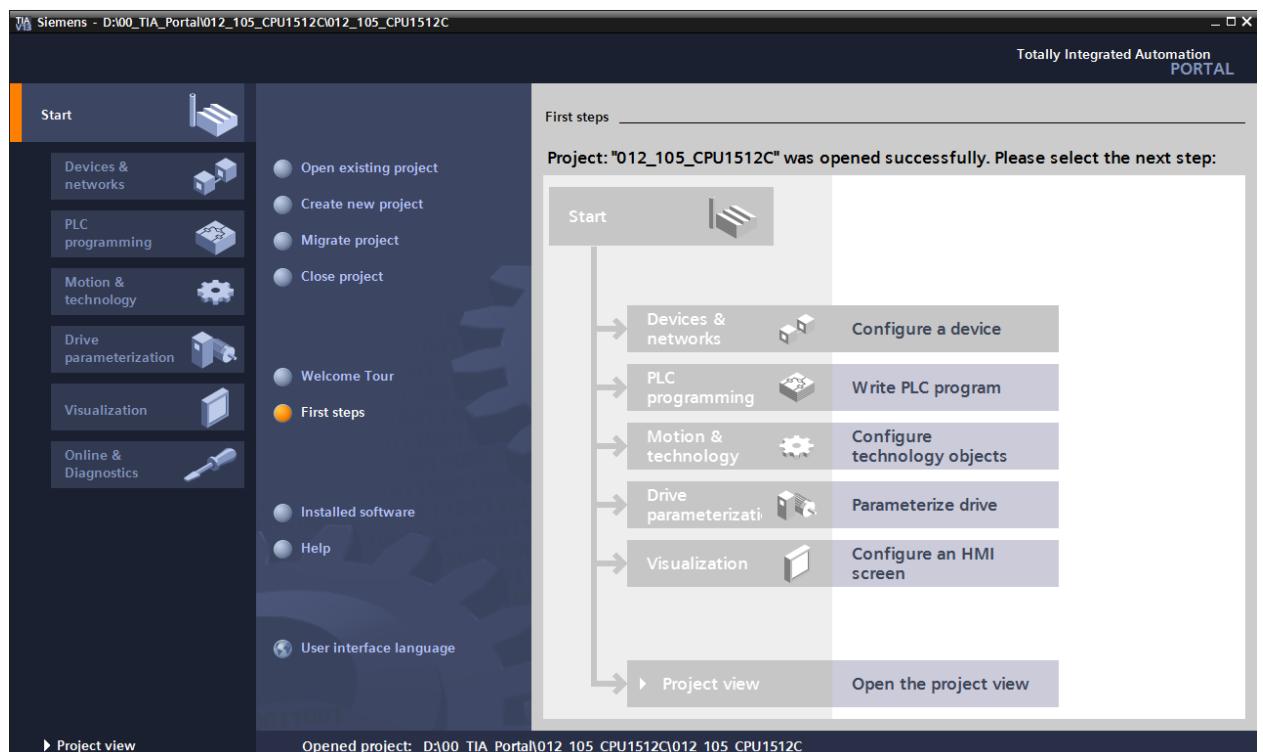


Figure 2: Portal view

The project view, as shown in Figure 3 is used for hardware configuration, programming, creation of the visualization and many other tasks.

By default, the project view displays the menu bar with the toolbars at the top, the project tree with all components of a project on the left and the so-called task cards with instructions and libraries, for example, on the right.

If an element (for example, the device configuration) is selected in the project tree, it is displayed in the center and can be worked on there.

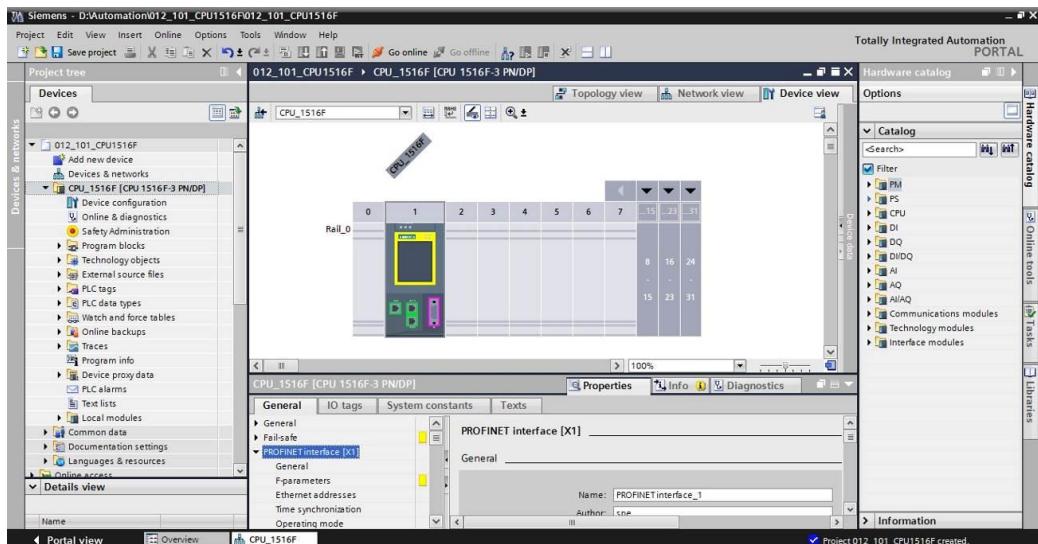
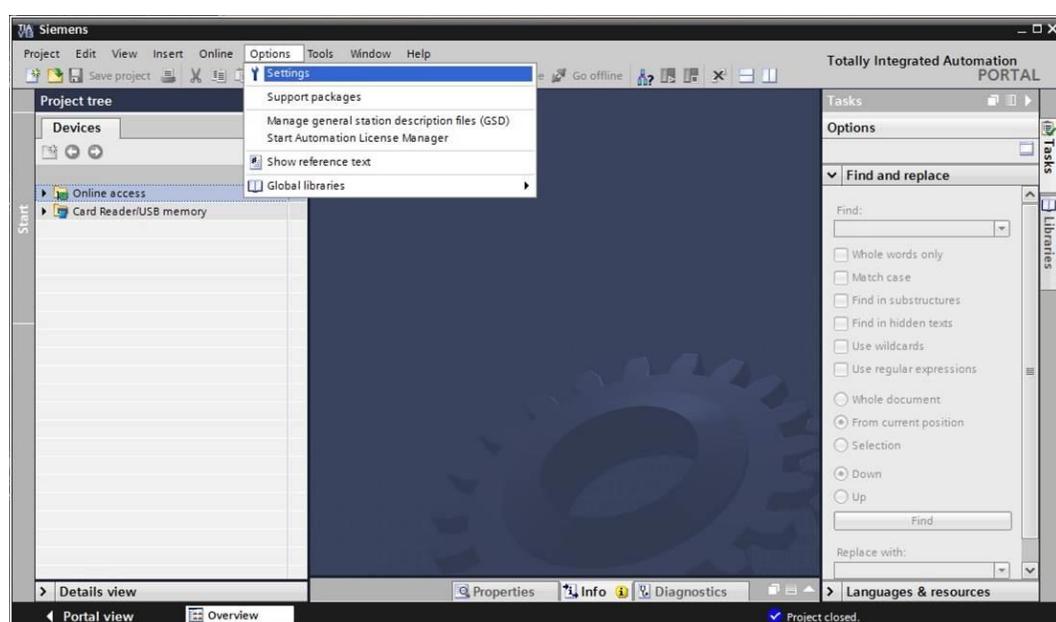


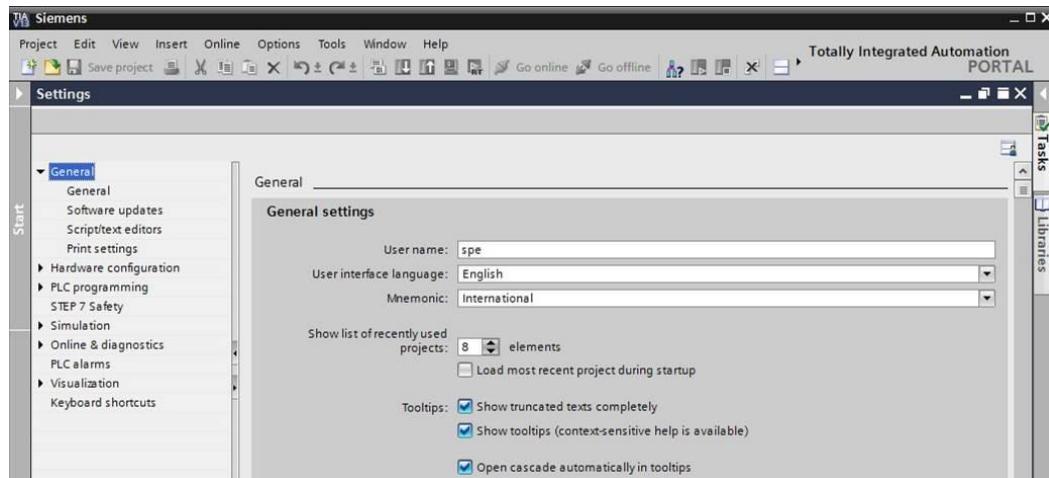
Figure 3: Project view

4.5.6 Basic settings for the TIA Portal

- Users can specify their own default settings for certain settings in the TIA Portal. A few important settings are shown here.
- In the project view, select the →"Options" menu and then → "Settings".

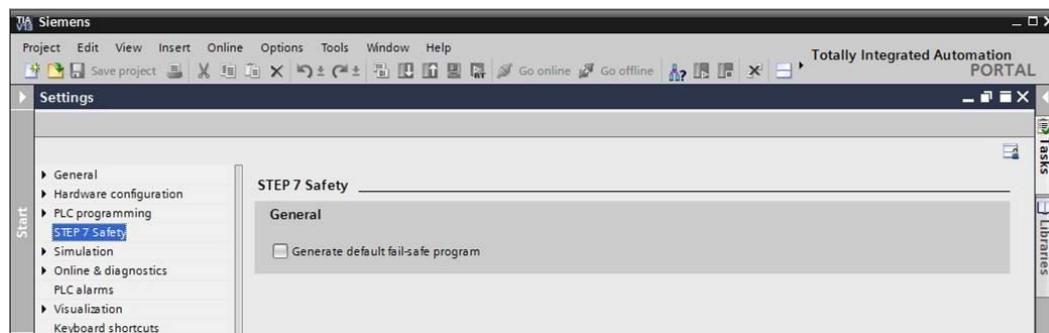


- One basic setting is the selection of the user interface language and the language for the program display. In the curriculums to follow, "English" will be used for both settings.
- Under → "General" in "Settings", select "User interface language → English" and "Mnemonic → International".



Note: These settings can always be changed.

- When Safety CPUs are used (e.g. CPU 1516F-3 PN/DP) without the use of safety engineering, it is recommended that automatic creation of the safety program be deactivated before creating a project.
- In "Settings" under the → "STEP 7 Safety" item, deactivate → "Generate default fail-safe program".



4.5.7 Setting the IP address on the programming device

To program SIMATIC S7-1500 from the PC, the programming device or a laptop, you need a TCP/IP connection or an optional PROFIBUS connection.

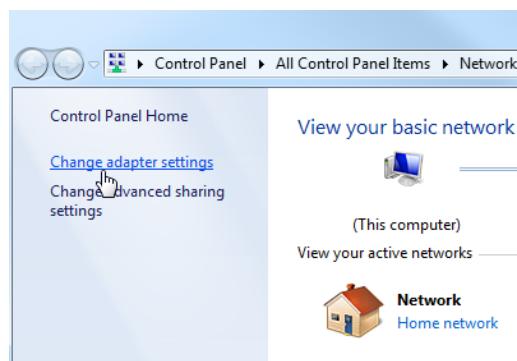
For the PC and SIMATIC S7-1500 to communicate with each other via TCP/IP, it is important that the IP addresses of both devices match.

First, we show you how to set the IP address of a computer with the Windows 7 operating system.

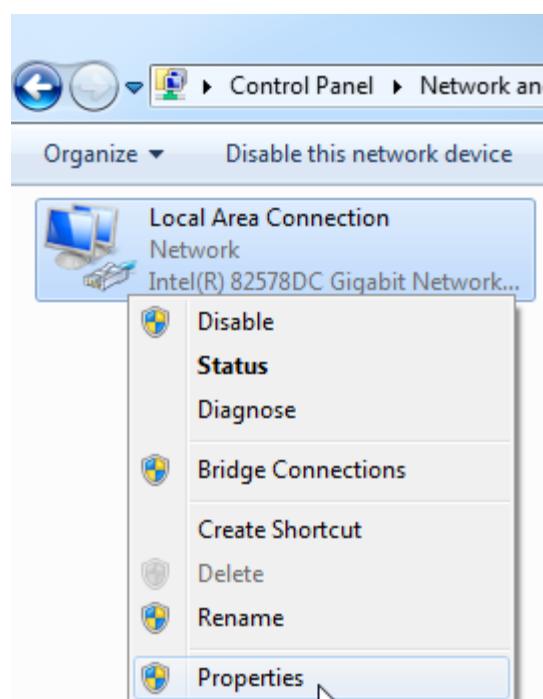
- Locate the network icon in the taskbar at the bottom and click → "Open Network and Sharing Center".



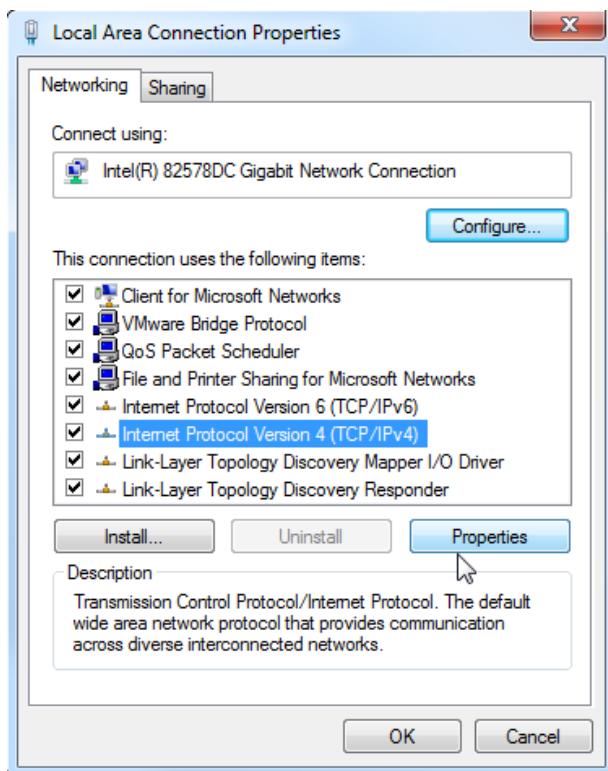
→ In the open Network and Sharing Center window, click → "Change adapter settings".



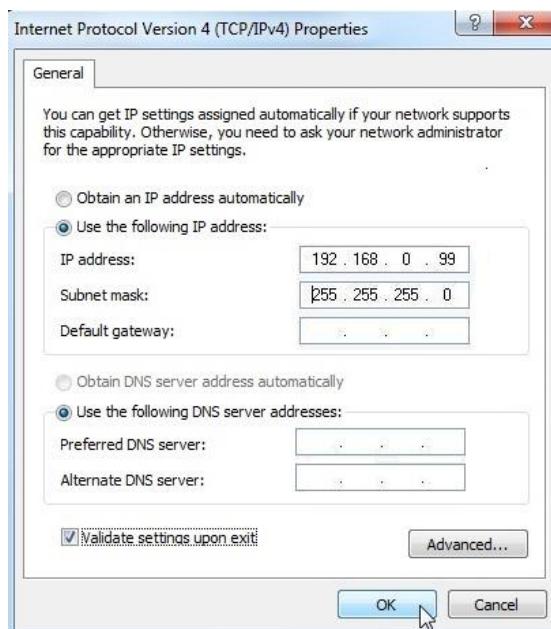
→ Select the desired → "Local Area Connection" that you want to use to connect to the controller and click → "Properties".



→ Next, select → "Properties" for → "Internet Protocol Version 4 (TCP/IP)".



→ You can use the following IP address, for example → IP address: 192.168.0.99 → Subnet mask 255.255.255.0 and accept the settings (→ "OK")

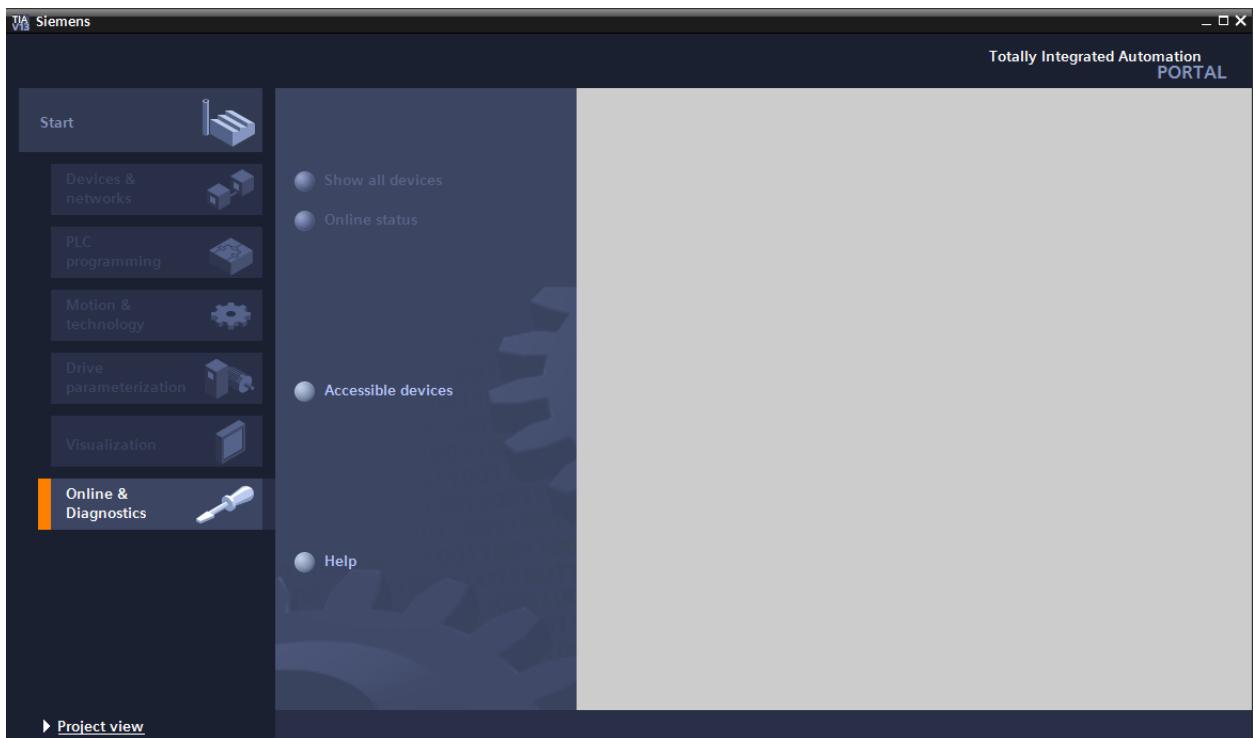


4.5.8 Setting the IP address in the CPU

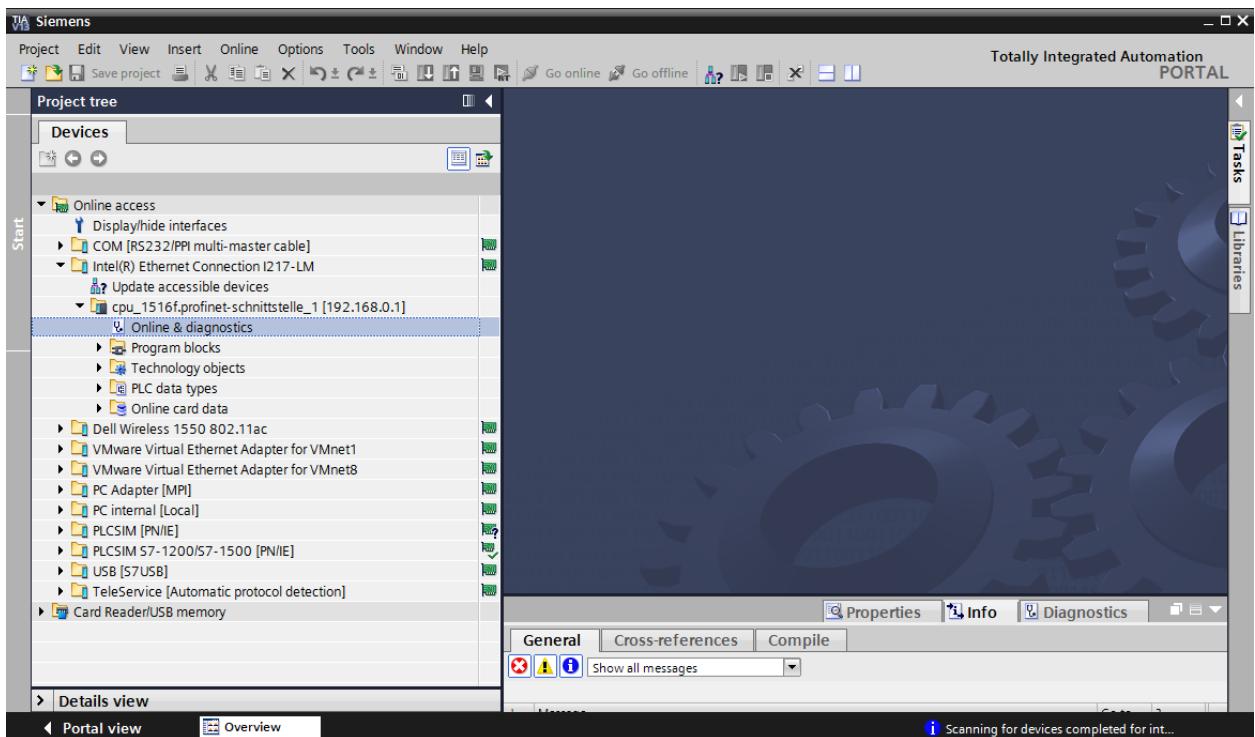
The IP address of SIMATIC S7-1500 is set as follows.

→ Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V1X)

→ Select → "Online & Diagnostics" and open the → "project view".

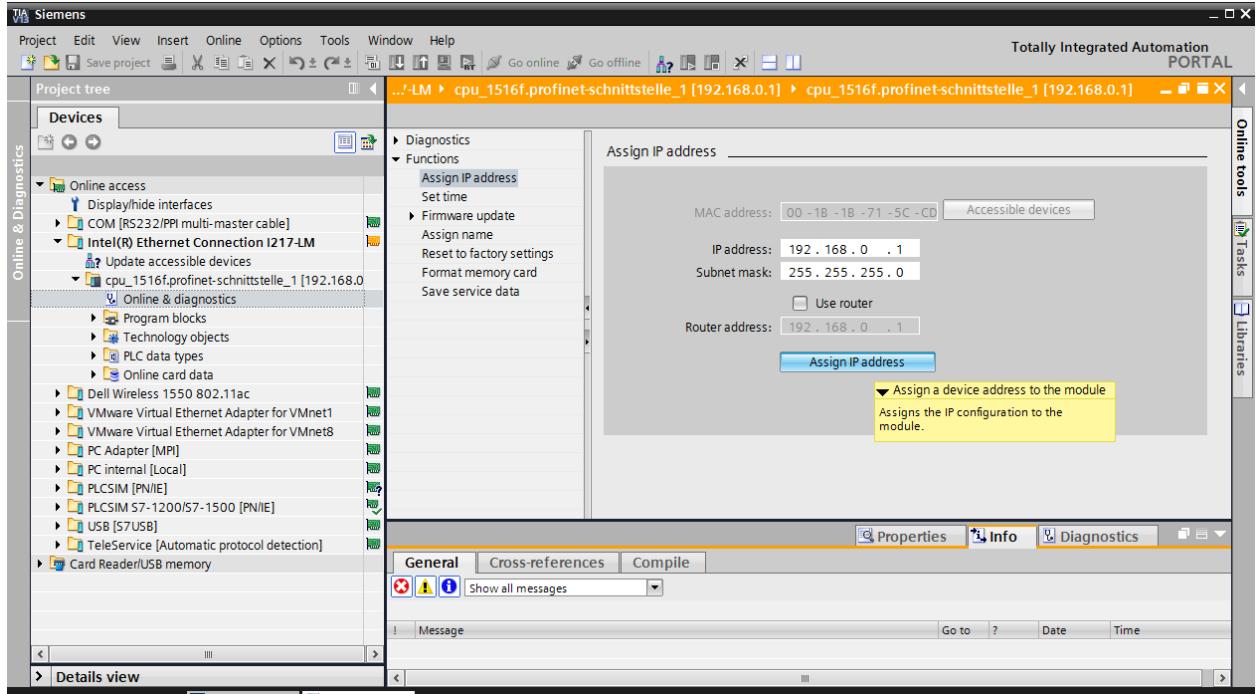


→ In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices" here, you will see the IP address (if previously set) or the MAC address (if IP address not yet assigned) of the connected SIMATIC S7-1500. Select → "Online & Diagnostics" here.



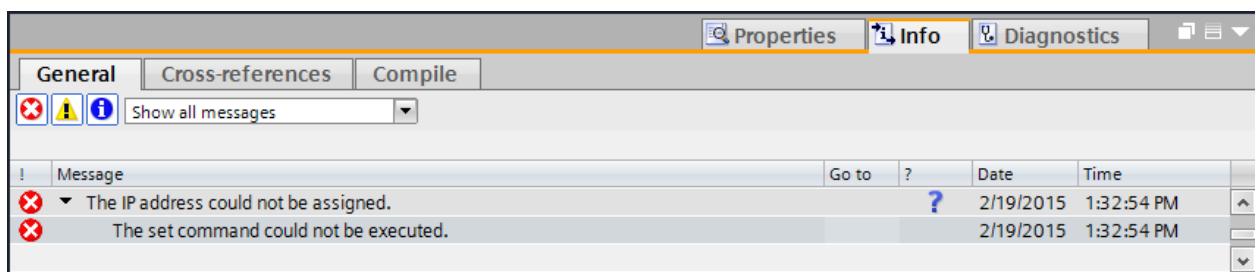
- Under → "Functions", you now find the → "Assign IP address" item. Enter the following IP address here (example): → IP address: 192.168.0.1 → Subnet mask 255.255.255.0.

Next, click → "Assign IP address" and this new address will be assigned to your SIMATIC S7-1500.



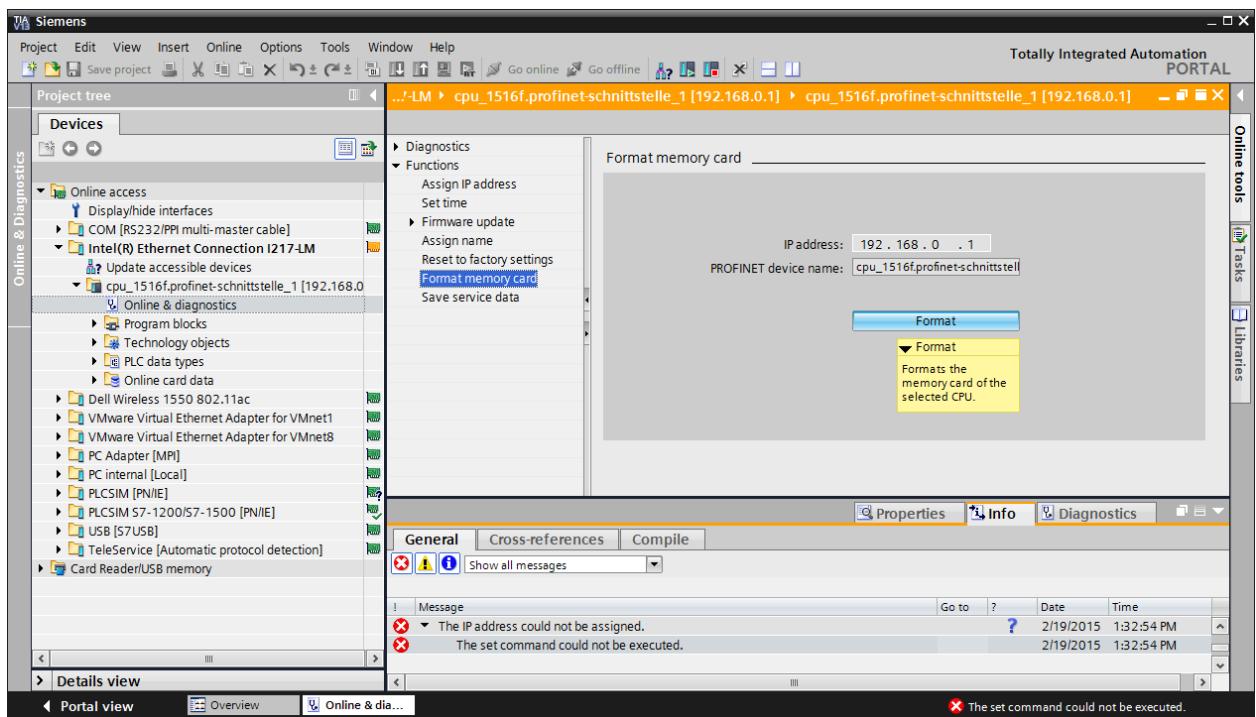
Note: The IP address of the SIMATIC S7-1500 can also be set via the display on the CPU, provided this is enabled in the hardware configuration.

- If the IP address was not successfully assigned, you will receive a message in the → "Info" window under → "General".

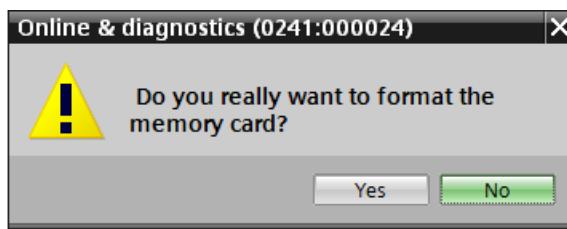


4.5.9 Formatting the memory card in the CPU

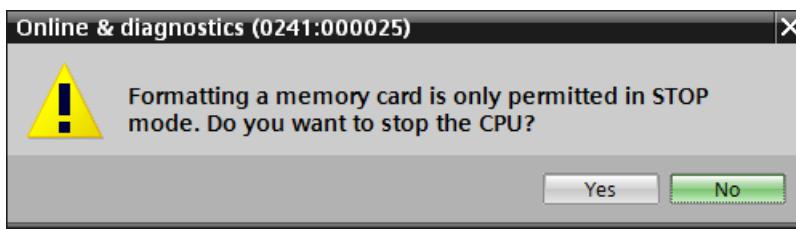
- If the IP address could not be assigned, the program data on the CPU must be deleted. This is accomplished in 2 steps: → "Format memory card" and → "Reset to factory settings".
- First, select the → "Format memory card" function and press the → "Format" button.



→ Confirm the prompt asking if you really want to format the memory card with → "Yes".

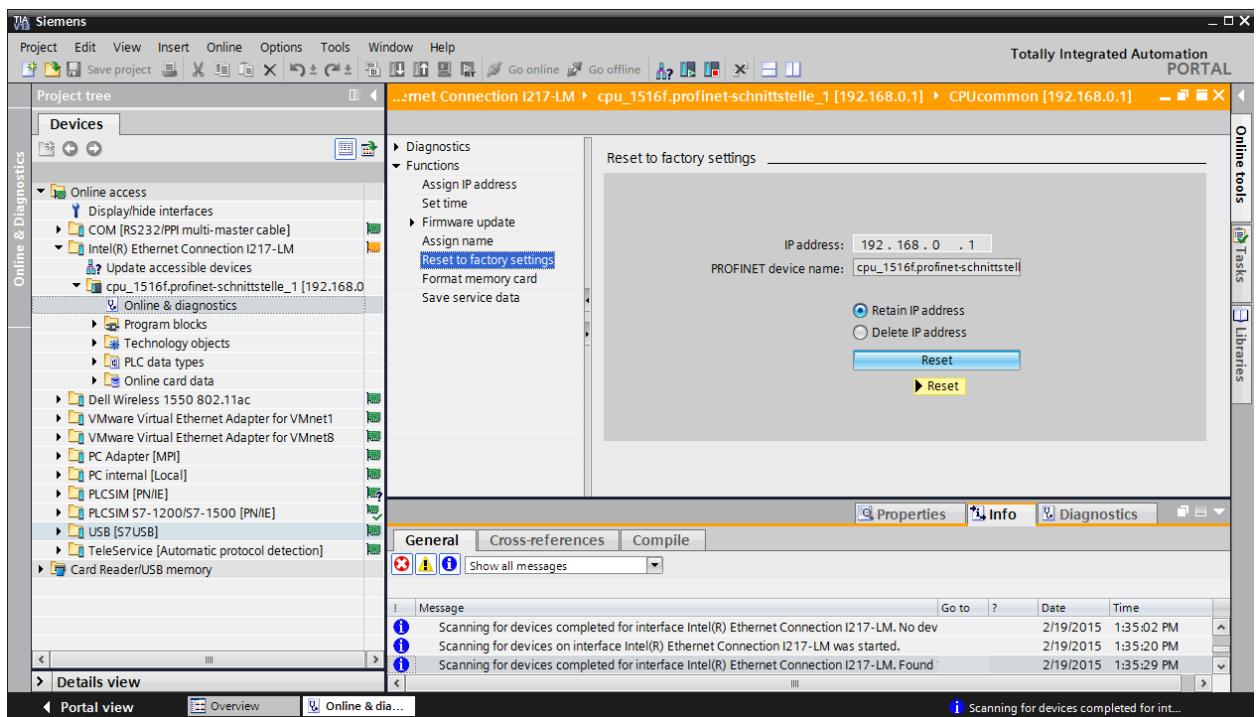


→ If necessary, stop the CPU. (→ "Yes")

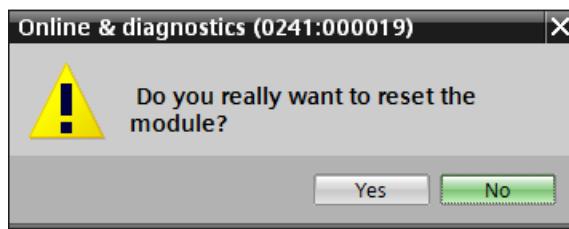


4.5.10 Resetting the CPU to factory settings

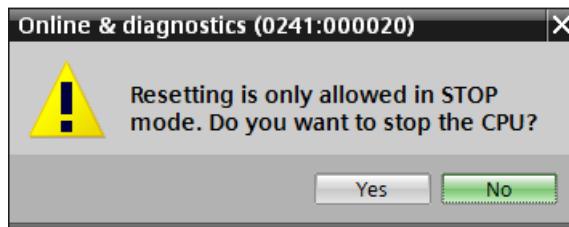
→ Before you can reset the CPU, you must wait until the formatting in the CPU has finished. Then you must select → "Update accessible devices" and → "Online & diagnostics" of your CPU again. To reset the controller, select the → "Reset to factory settings" function and click → "Reset".



- Confirm the prompt asking if you really want to reset the module with → "Yes".



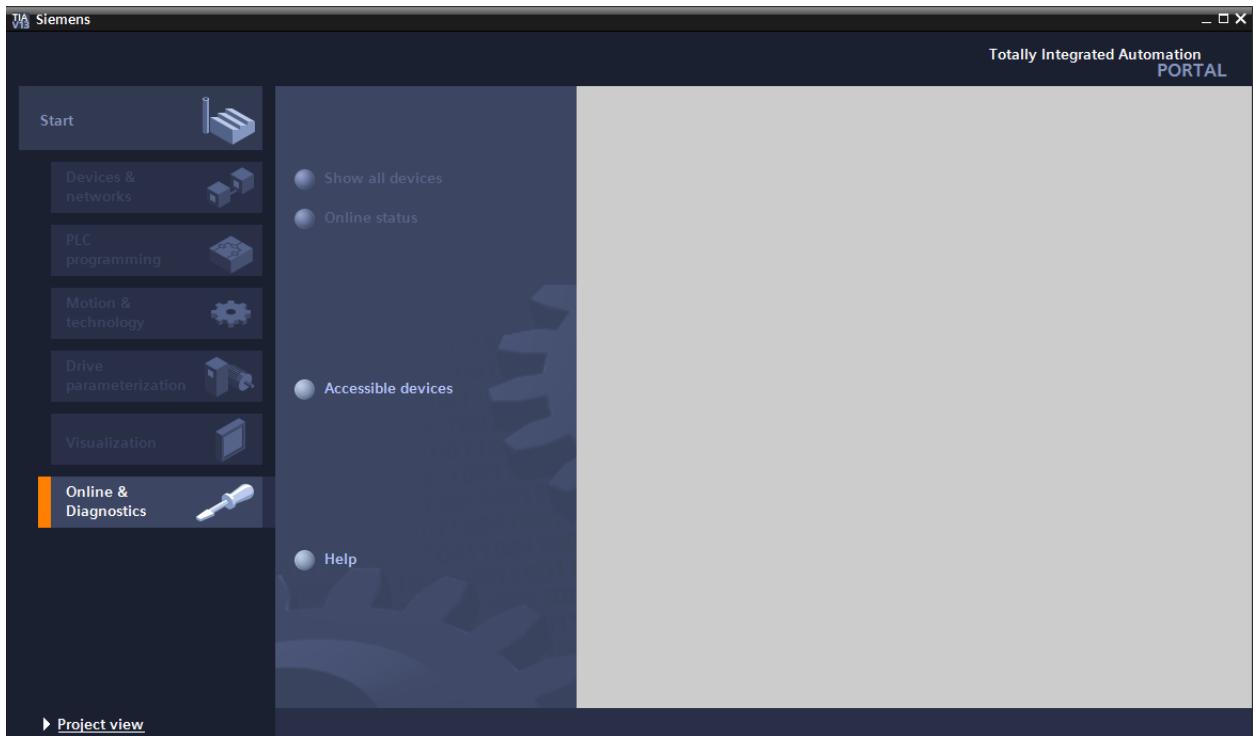
- If necessary, stop the CPU. (→ "Yes")



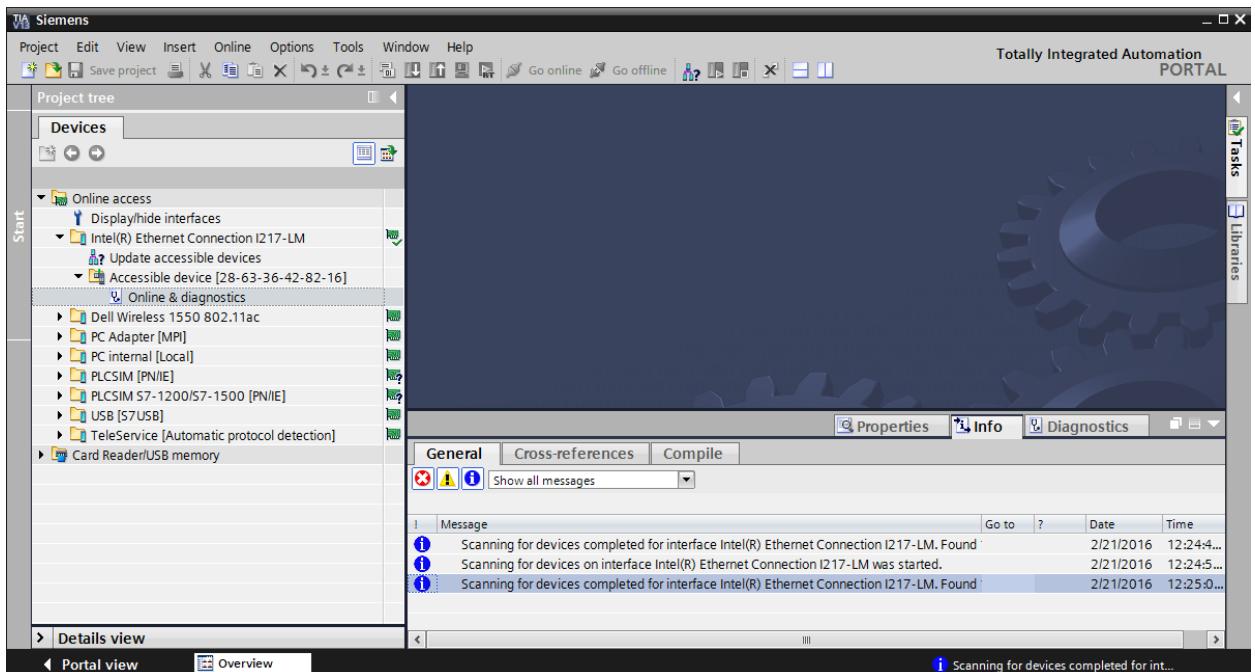
4.5.11 Setting the IP address in the ET 200SP

The IP address of the ET 200SP is set as follows.

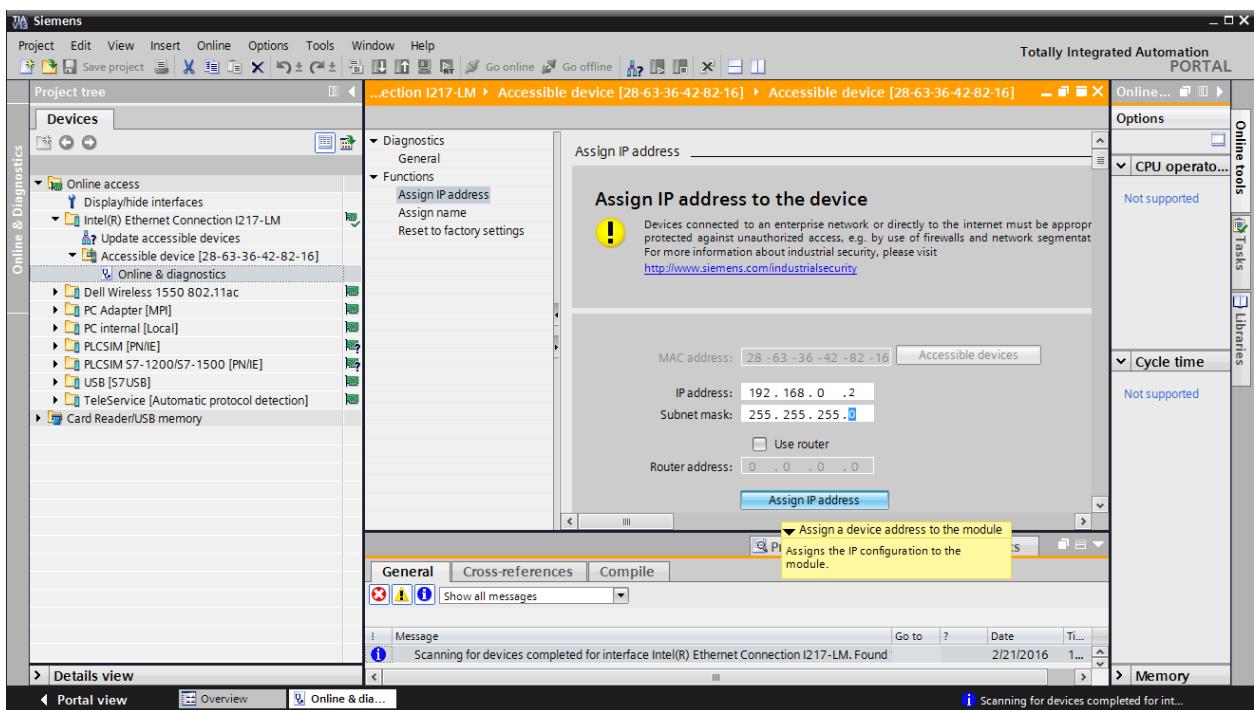
- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V1X)
- Select → "Online & Diagnostics" and open the →"Project view".



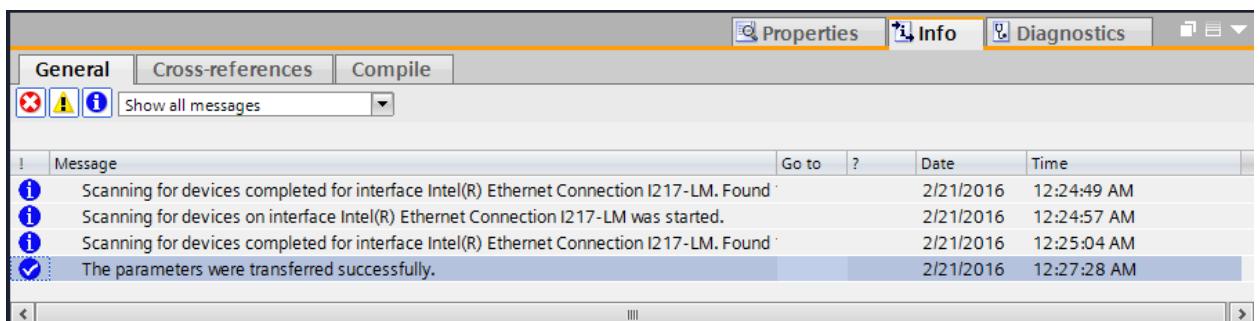
- In the project tree under → "Online access", select the network adapter that was set previously. If you click → "Update accessible devices" here, you will see the IP address (if previously set) or the MAC address (if the IP address is not yet assigned) of the connected ET 200SP. Select → "Online & Diagnostics" here.



- Under → "Functions", you now find the → "Assign IP address" item. Enter the following IP address here (example): → IP address: 192.168.0.2 → Subnet mask 255.255.255.0. Next, click → "Assign IP address" and this new address will be assigned to your ET 200SP.



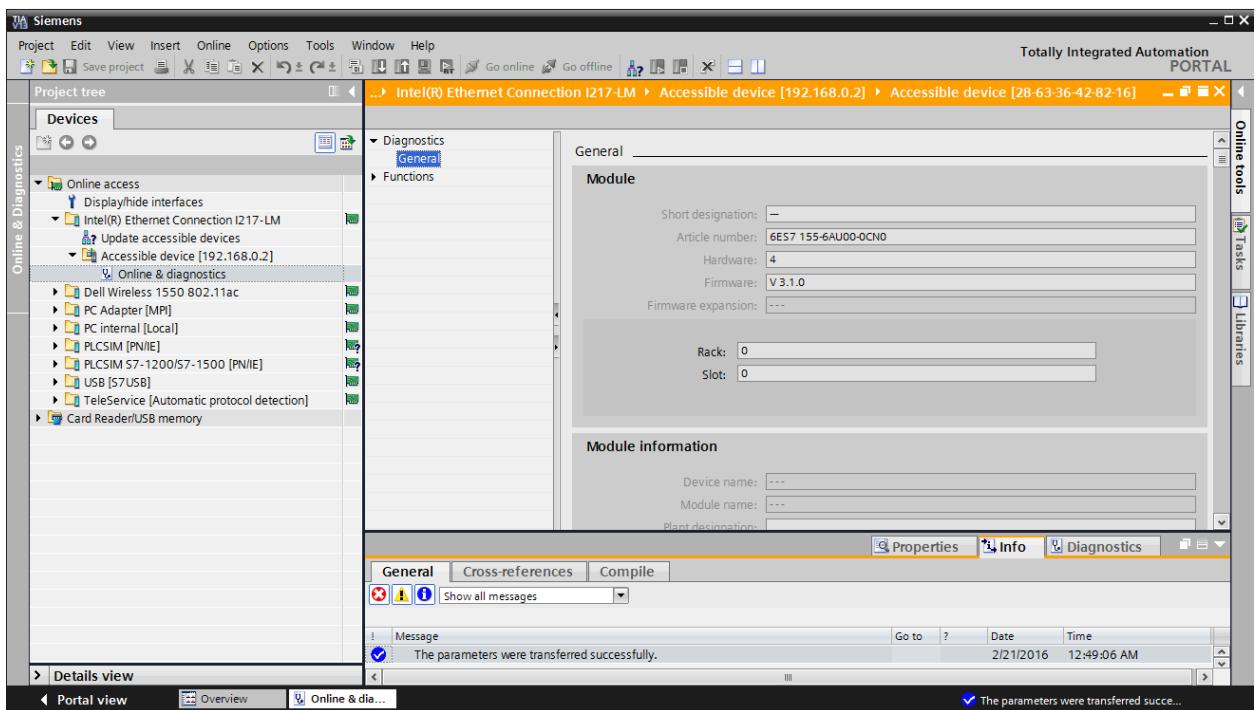
- A feedback message for the assignment of the IP address is provided in the → "Info" window under → "General".



Note: If a communication connection already exists between the ET 200SP (as device) and a higher-level controller (as controller), the IP address cannot be changed.

4.5.12 Reading the firmware version of the ET 200SP

- Before you can read the firmware version of the ET 200SP, you must select → "Update accessible devices" and → "Online & diagnostics" of your ET 200SP again. You can read the Short description, Article number, Hardware version and Firmware version in menu item → "Diagnostics" → "General".



5 Task

Create a project and configure the following modules of your hardware, which correspond to one part of SIMATIC CPU 1516F PN/DP Safety and SIMATIC ET 200SP Digital.

- SIMATIC S7-1500F, CPU 1516F-3 PN/DP, WORK MEMORY 1.5 MB PROGRAM, 5 MB DATA, 1. INTERFACE, PROFINET IRT WITH 2 PORT SWITCH, 2. INTERFACE, ETHERNET, 3. INTERFACE, PROFIBUS, 10 NS BITPERFORMANCE, SIMATIC MEMORY CARD REQUIRED (order number: 6ES7 516-3FN01-0AB0)
- 1X SIMATIC PM 1507 24 V/8 A STABILIZED POWER SUPPLY INPUT: 120/230 V AC OUTPUT: 24 V DC / 8 A (order number: 6EP1333-4BA00)
- 1X INTERFACE MODULE IM155-6PN HF (order number: 6ES7 155-6AU00-OCN0)
- 1X BUS ADAPTER BA 2XRJ45 (order number: 6ES7 193-6AR00-0AA0)
- 2X DI 8X24VDC/0.5A HF (order number: 6ES7 131-6BF00-0CA0)
- 2X DQ 8X24VDC/0.5A HF (order number: 6ES7 132-6BF00-0CA0)
- Server module (order number: 6ES7 193-6PA00-0AA0)

6 Planning

Because this is a new system, a new project must be created.

The hardware for this project is already specified by the existing hardware. Therefore, a selection does not have to be made. Instead, the listed modules of the trainer packages only have to be inserted in the project and connected. The order numbers (see Table 1 and Table 2) can be used to check that the correct modules are inserted.

Module	Order number	Slot	Address area
PM 190W 120/230VAC	6EP1333-4BA00	0	
CPU 1516F-3 PN/DP	6ES7516-3FN01-0AB0	1	

Table 1: Modules of the S7-1500

Module	Order number	Slot	Address area
IM155-6PN HF	6ES7155-6AU00-0CN0	0	
DI 8x24VDC HF	6ES7131-6BF00-0CA0	1	0
DI 8x24VDC HF	6ES7131-6BF00-0CA0	2	1
DQ 8x24VDC/0.5A HF	6ES7132-6BF00-0CA0	3	0
DQ 8x24VDC/0.5A HF	6ES7132-6BF00-0CA0	4	1
Server module	6ES7193-6PA00-0AA0	5	

Table 2: Modules of the ET 200SP

The Base Units are relevant for additional handling of the ET 200SP modules. These determine whether the potential is taken from the left terminal (dark Base Unit) or whether a new voltage supply must be connected and thus a new potential group is created (light Base Unit). The applicable rule here is that a new potential must always be provided on slot 1.

All Base Units included in the trainer packages are type BU15-P16+A0+2D (6ES7193-6BP00-0DA0). As a result, the light Base Unit version is set by default.

As the final step, the hardware configuration is saved, compiled, downloaded and started.

Any errors present can be detected during compilation and incorrect modules can be detected when the controller is started (*only possible when hardware is present and structured identically*).

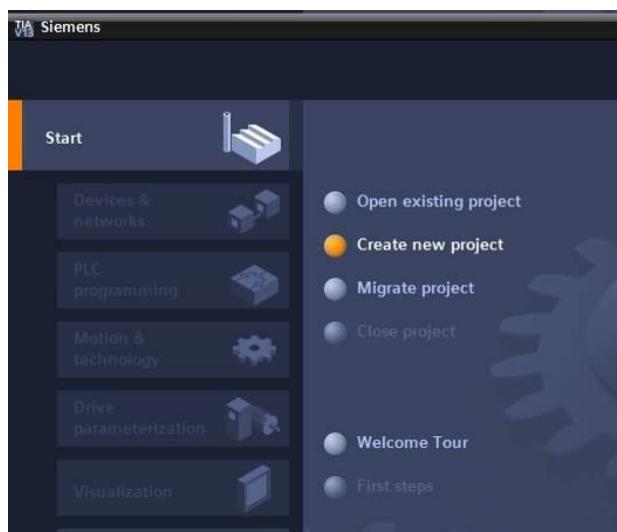
The result is archived to back up the working version.

7 Structured step-by-step instructions

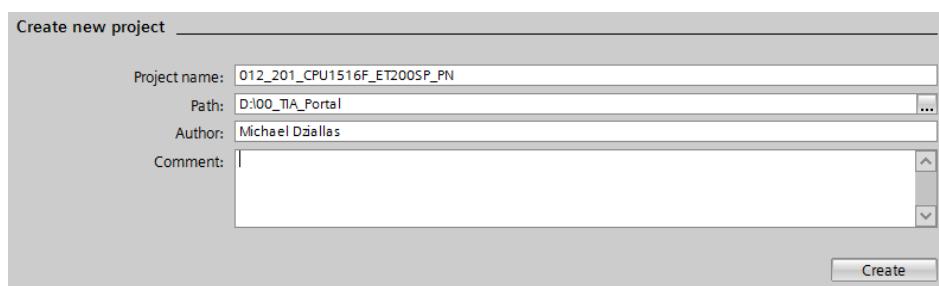
You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it is sufficient to focus on the numbered steps. Otherwise, simply follow the steps of the instructions illustrated below.

7.1 Create a new project

- Select the Totally Integrated Automation Portal for this, which is opened here with a double-click. (→ TIA Portal V1X)
- In the portal view under the "Start" menu, select → "Create new project".



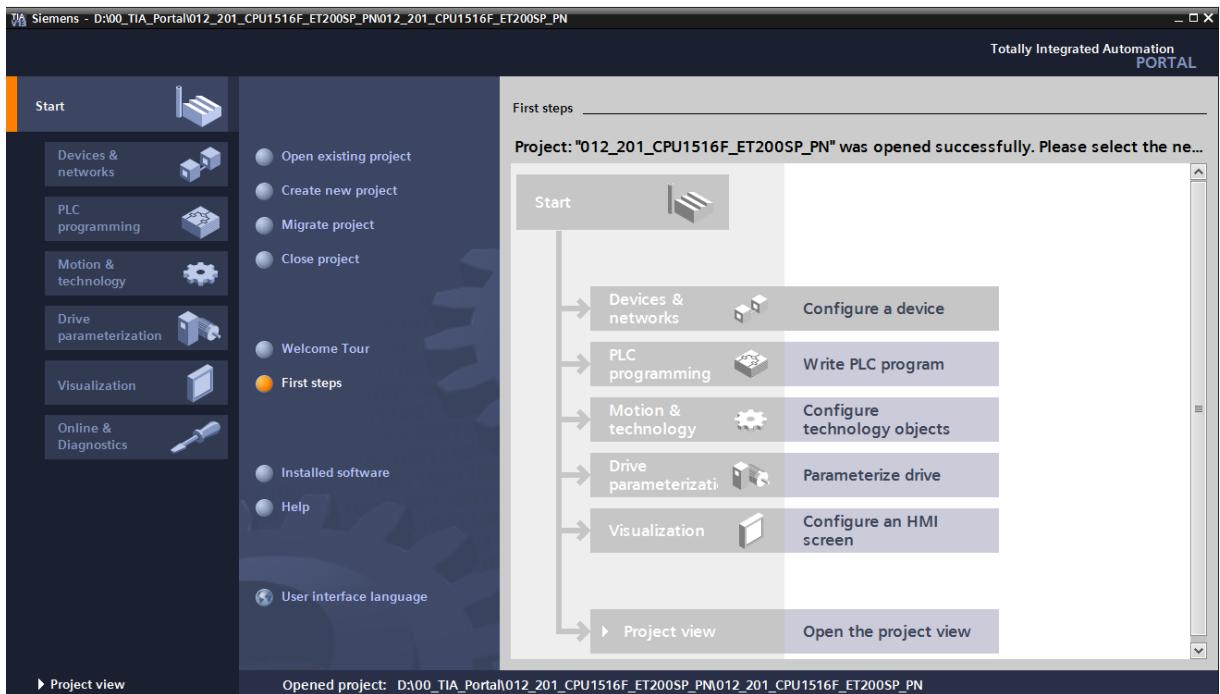
- Modify Project name, Path, Author and Comment as appropriate and click → "Create".



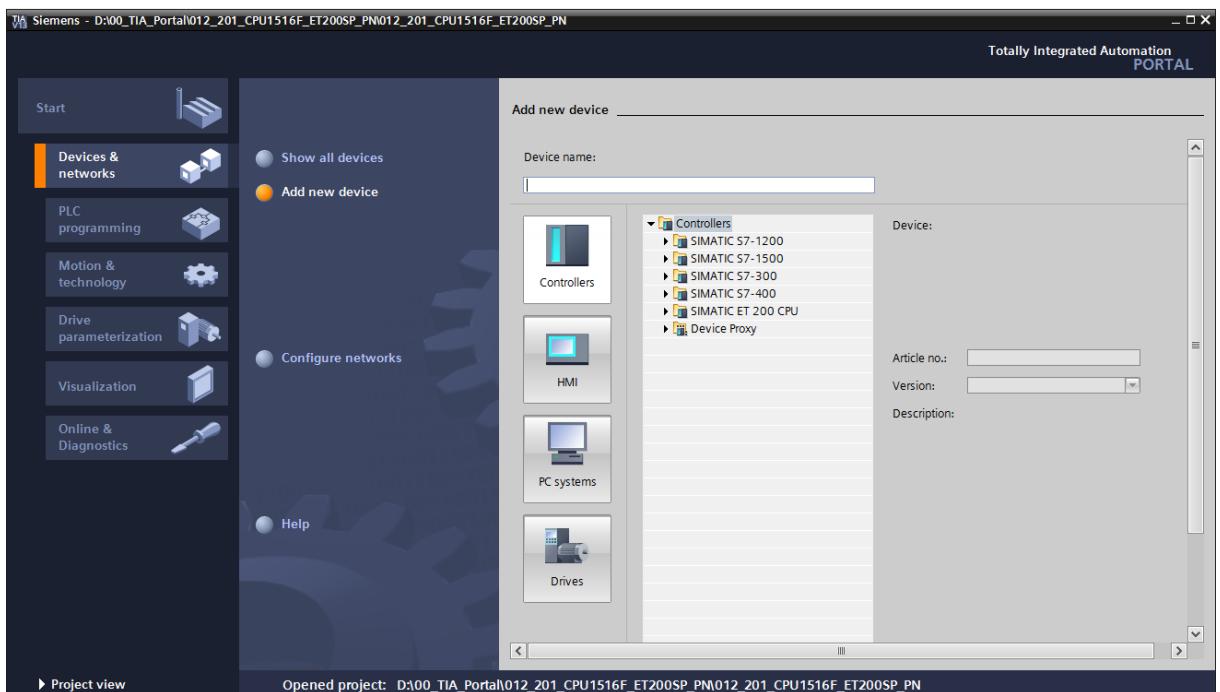
- The project is created and opened and the menu "Start", "First steps" opens automatically.

7.2 Insert the CPU 1516F-3 PN/DP

- In the "Start" → portal, select "First steps" → "Devices & Networks" → "Configure a device".

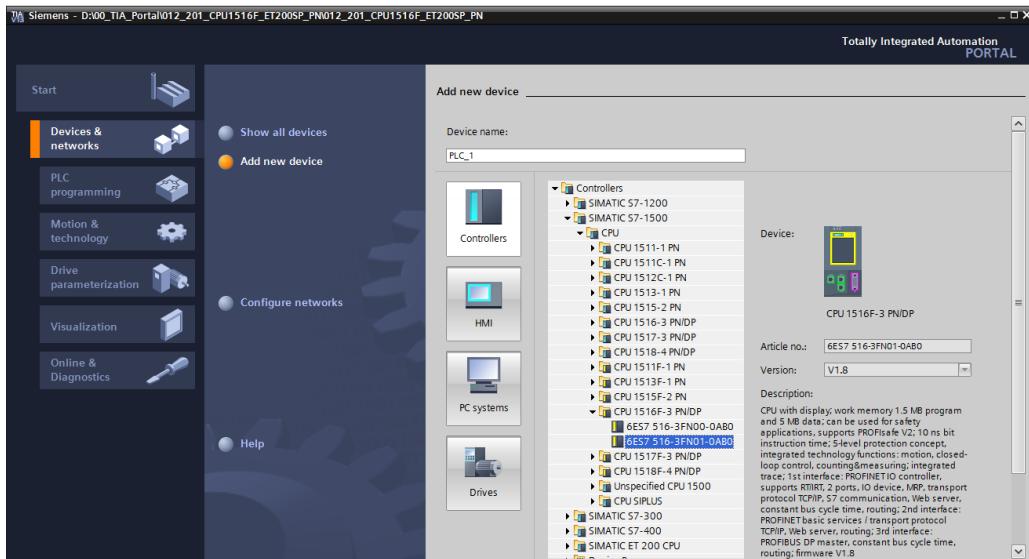


- In the "Devices & Networks" portal, the "Show all devices" menu opens
→ Switch to the "Add new device" menu.

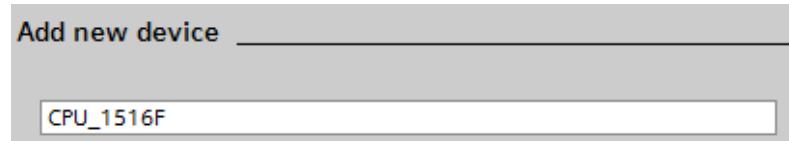


→ The specified model of the CPU will now be added as a new device.

(Controllers → SIMATIC S7-1500 → CPU → CPU 1516F-3 PN/DP → 6ES7516-3FN01-0AB0 → V1.8)



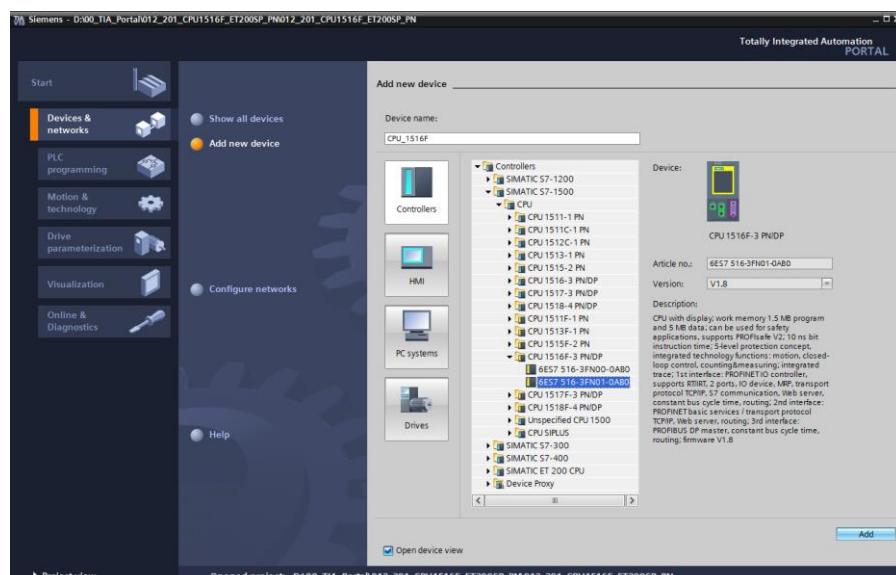
→ Assign a device name (Device name → "CPU_1516F").



→ Select "Open device view".



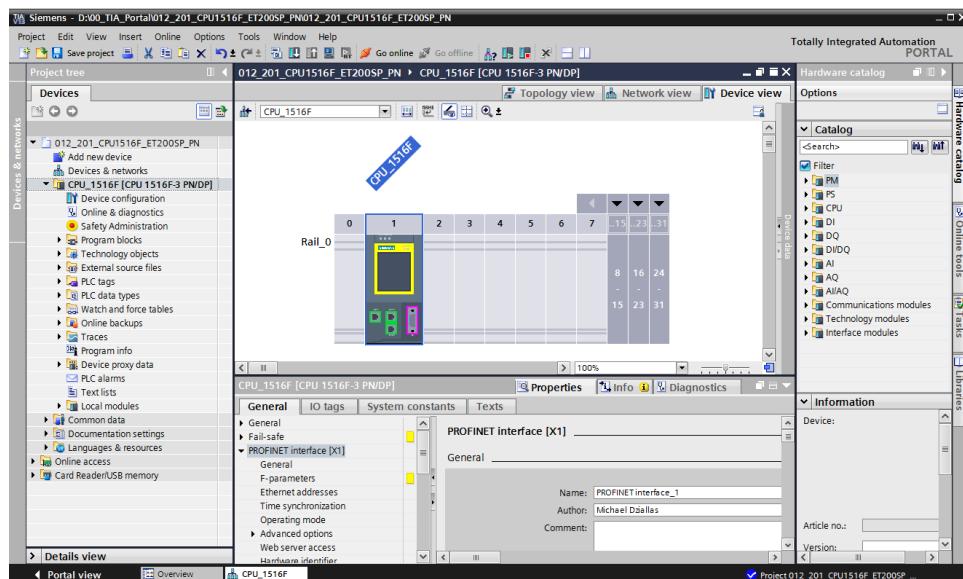
→ Click "Add".



Note: The desired CPU may have multiple versions that differ in functionality (work memory, integrated memory, technology functions, etc.). In this case, you should ensure that the selected CPU corresponds to the existing hardware.

Note: Different firmware versions are often offered for the hardware. In this case, it is recommended that the latest firmware (selected by default) be used and that the CPU be upgraded, if necessary.

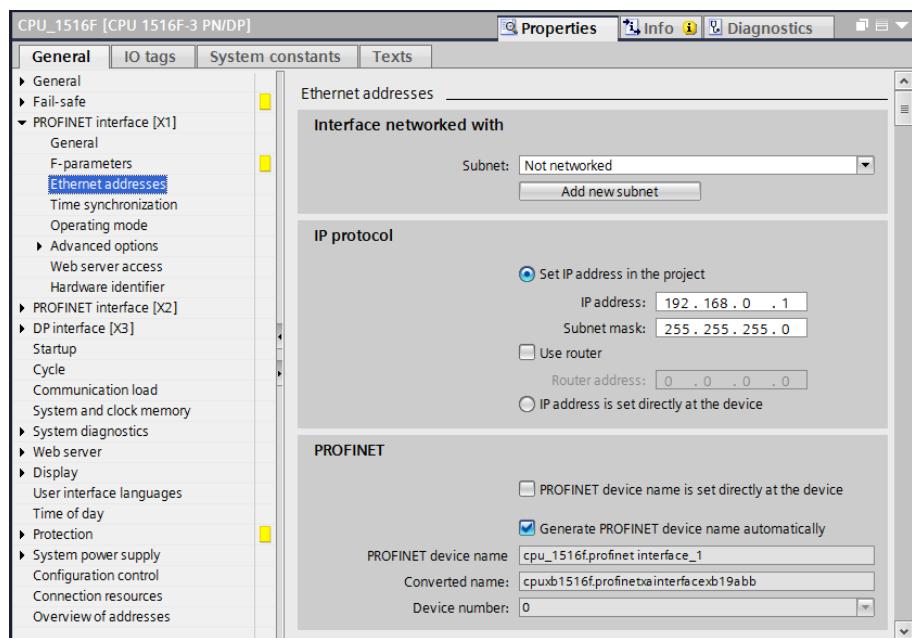
- The TIA Portal now changes automatically to the project view and displays the selected CPU in the device configuration in slot 1 of a rail.
- Select the CPU with a double-click



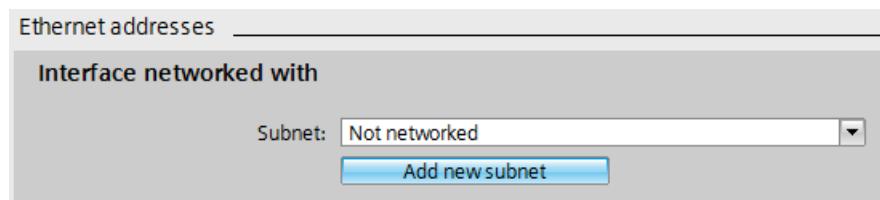
Note: You can now configure the CPU there according to your specifications. Possible settings include the PROFINET and PROFIBUS DP interfaces, startup characteristics, cycle, communication load and many others.

7.3 Configure the Ethernet interface of the CPU 1516F-3 PN/DP

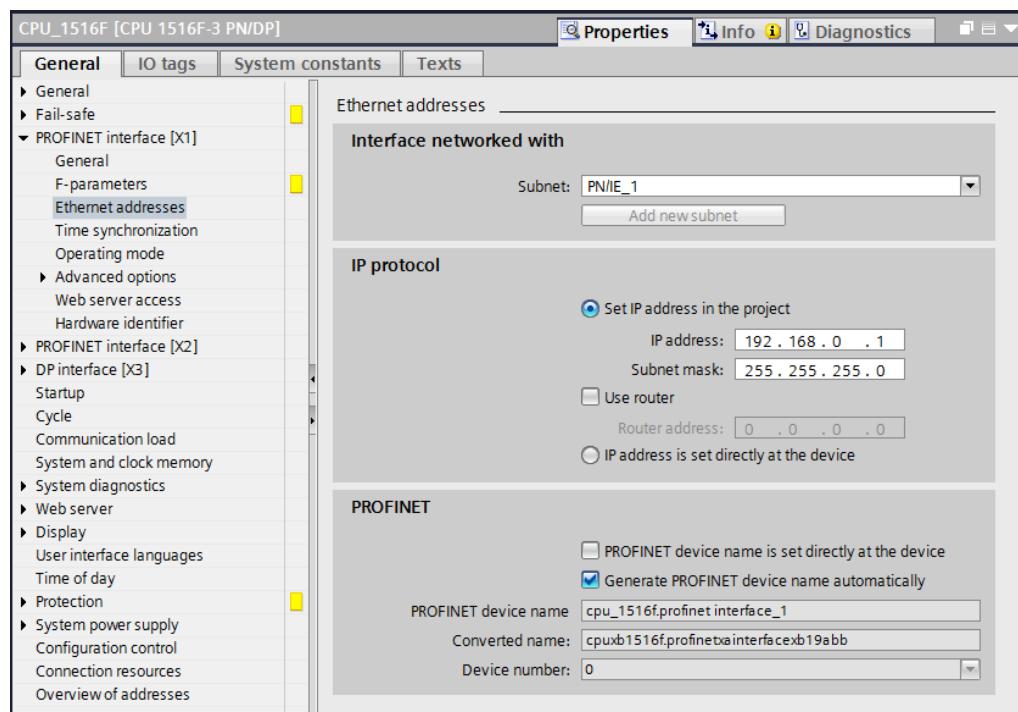
- Double-click the CPU to select it. Then open the → "PROFINET interface [X1]" menu in → "Properties" and select the → "Ethernet addresses" entry there.



- Under "Interface networked with", only the "Not networked" entry is available.
- Add an Ethernet subnet with the → "Add new subnet" button.

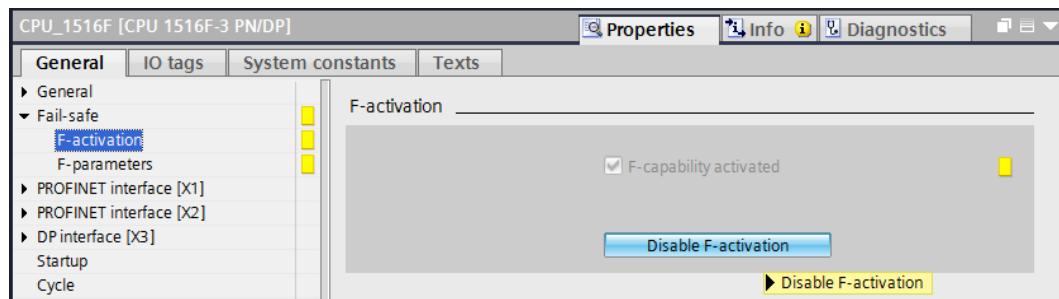


- Keep the preassigned "IP address" and "Subnet mask".

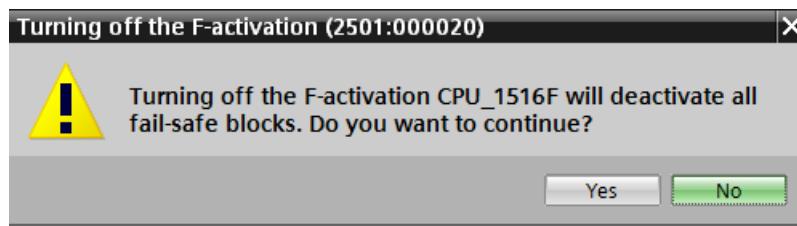


7.4 Configure the fail-safe operation of the CPU 1516F-3 PN/DP

- In the → "Fail-safe" menu, select → "F-activation" and select → "Disable F-activation" there.

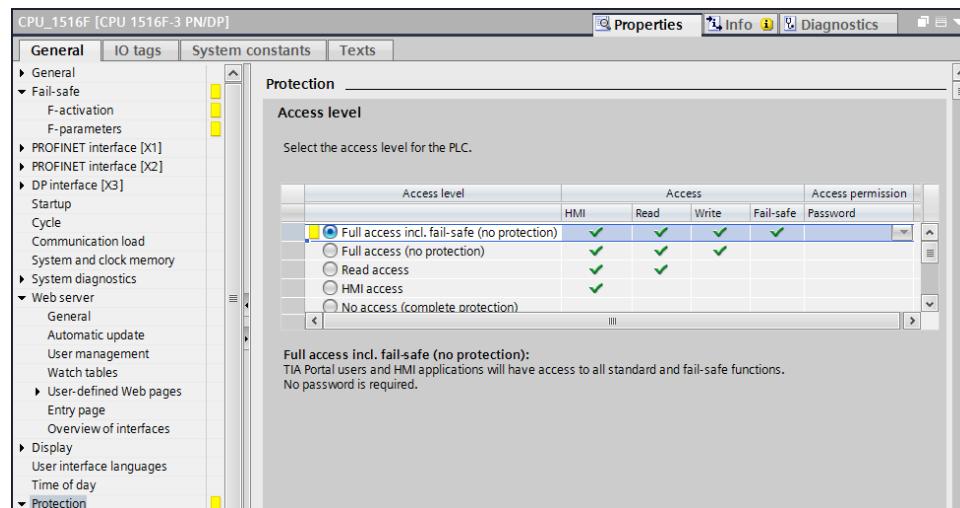


- Confirm the prompt asking if you want to continue with → "Yes".



7.5 Configure the access level for the CPU 1516F-3 PN/DP

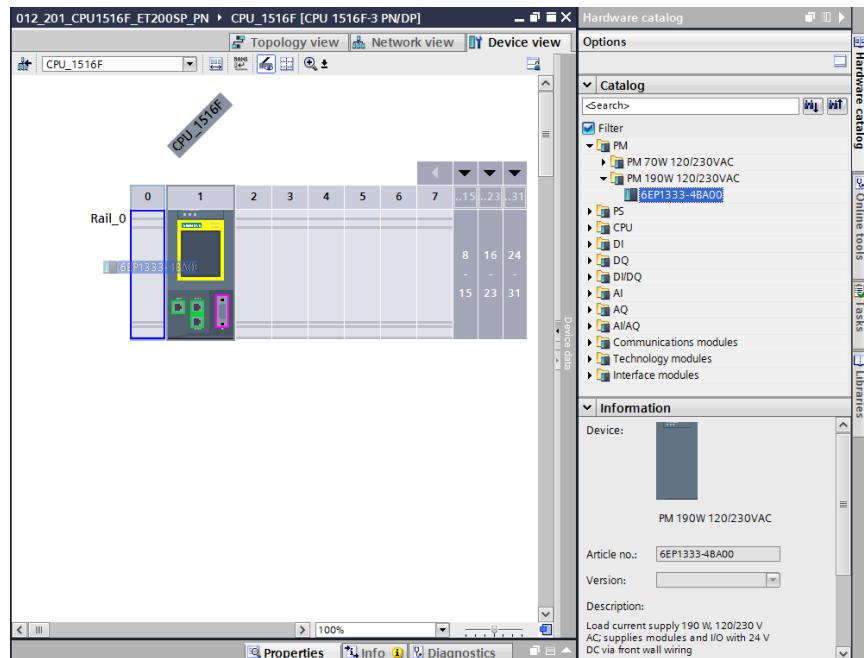
- Switch to the → "Protection" menu and select access level → "Full access incl. fail-safe (no protection)".



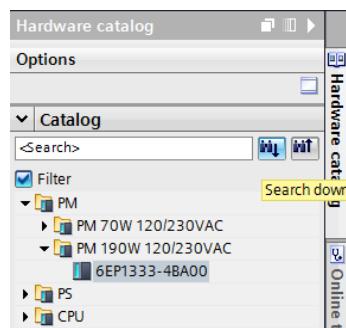
Note: The "Full access incl. fail-safe (no protection)" setting is recommended because we do not have to assign a password.

7.6 Insert power module PM 190W 120/230VAC

- Find the correct module in the hardware catalog and insert the power module in slot 0.
- (→ Hardware Catalog → PM → PM 190W 120/230VAC (order number 6EP1333-4BA00)
- Slot 0)



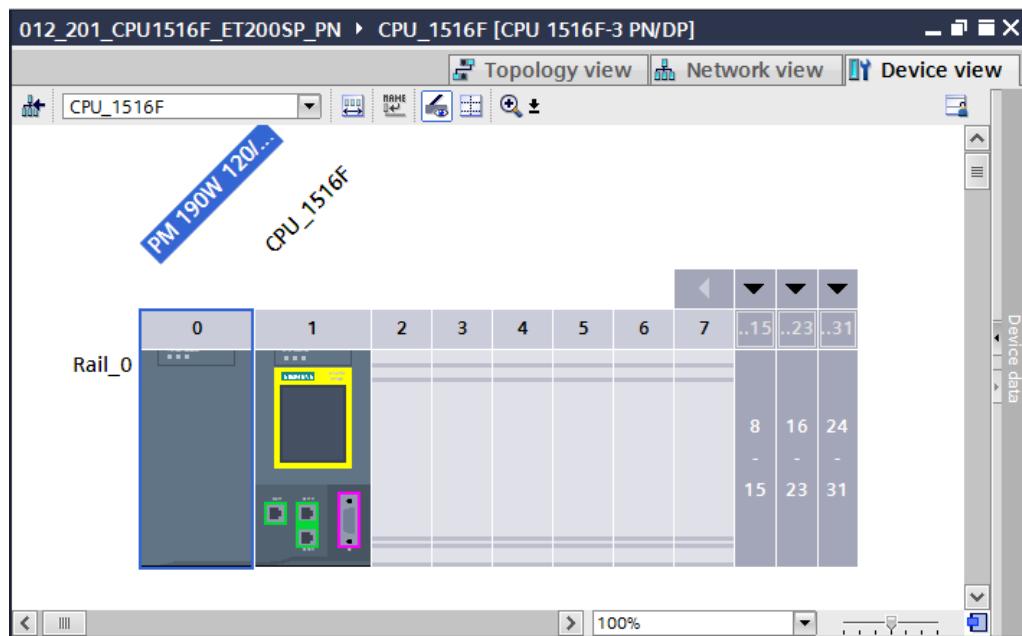
Note: To select a component in the hardware catalog, you can simply enter the order number in the Search field and then click the "Search down" icon . The hardware catalog will open at the correct position.



Note: When you double-click a module in the hardware catalog, you insert it at the next available compatible slot.

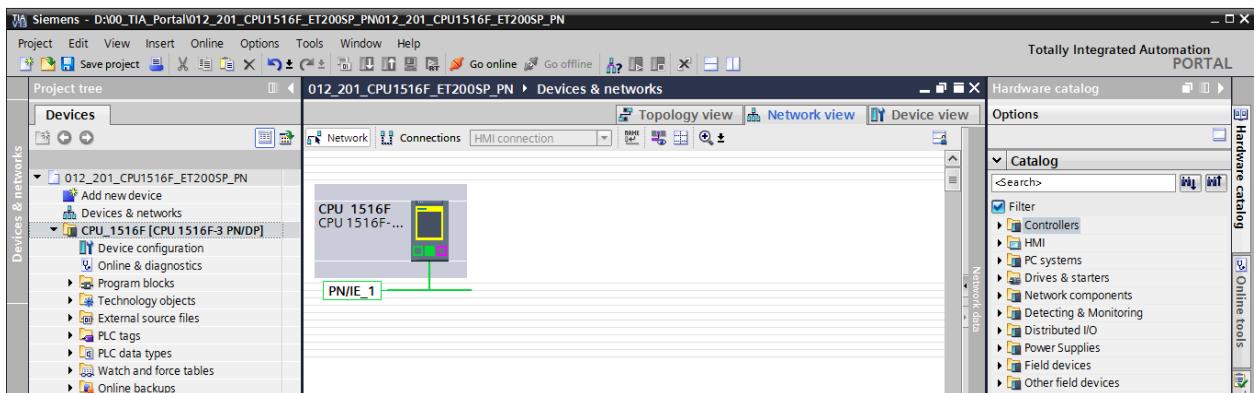
Note: If a module such as the power module is planned only for one slot, it is also not possible to place it at another position in the device configuration.

- Compare your device configuration with the following figure.

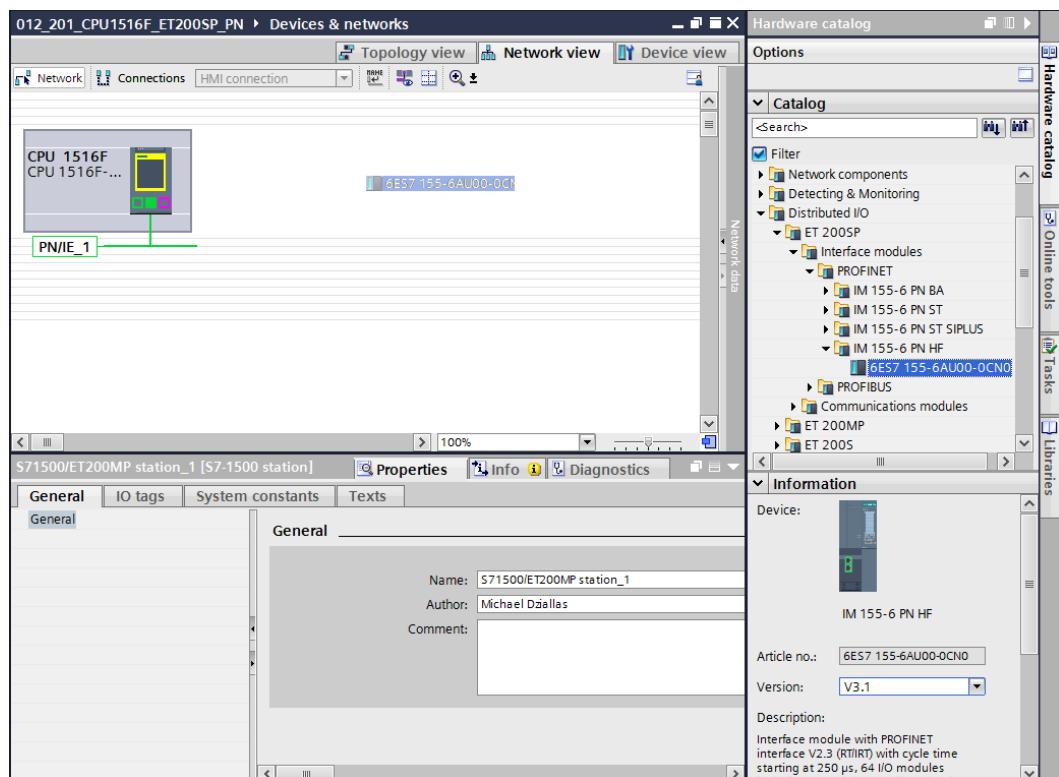


7.7 Insert the ET 200SP interface module IM155-6PN HF

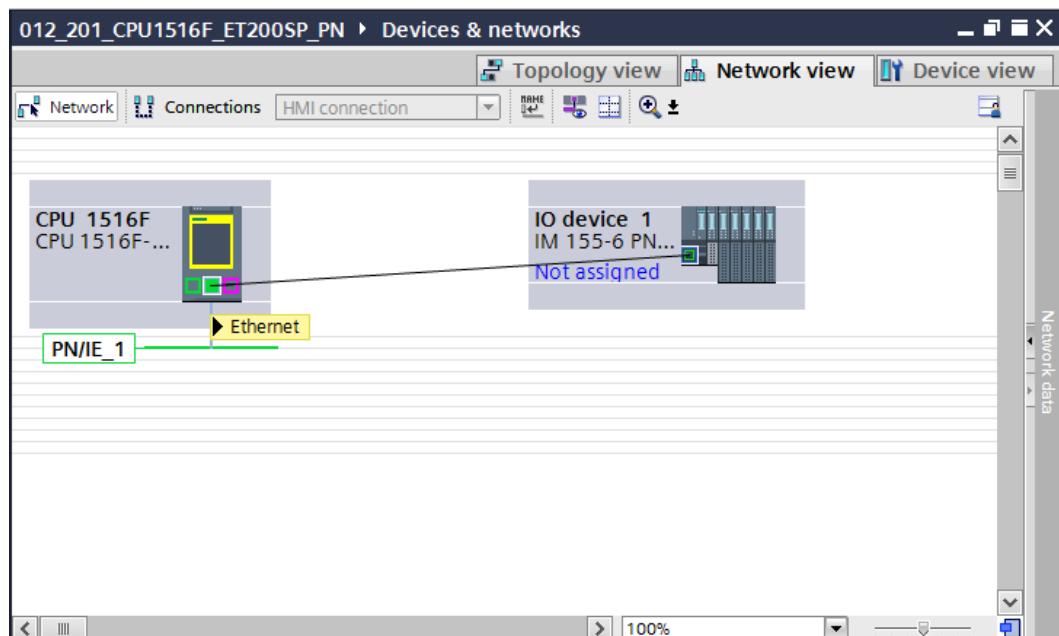
→ Open the network view (→ Network view)



→ Find the correct IM155-6PN HF interface module in the hardware catalog and insert it by moving it to the network view using drag-and-drop. (→ Hardware Catalog → Distributed IO → ET 200SP → Interface modules → PROFINET → IM 155-6 PN HF → 6ES7 155-6AU00-0CN0 → Version: V3.1)

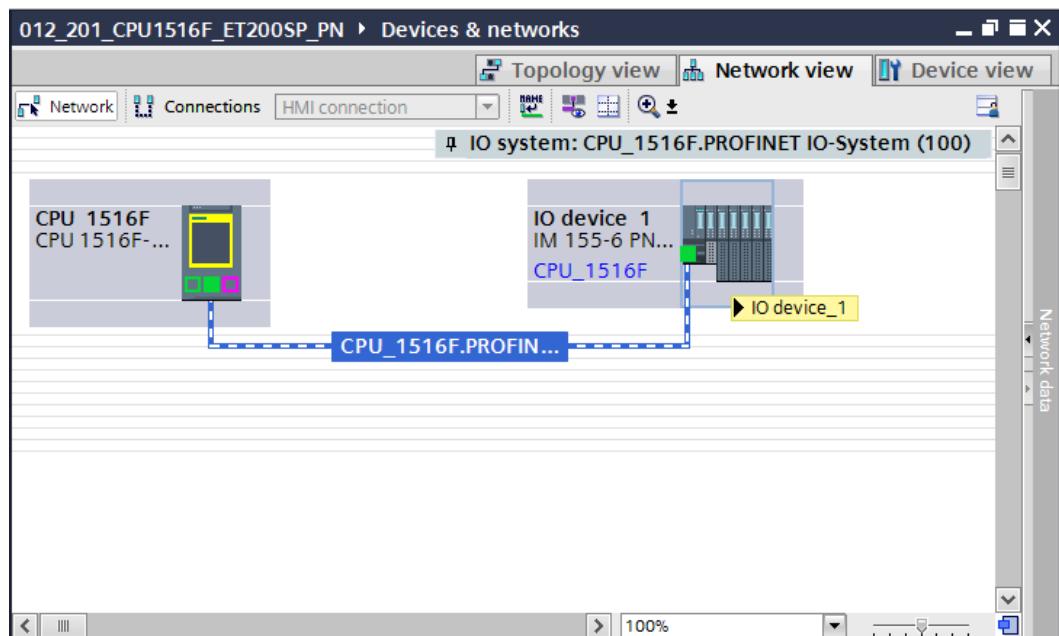


- Assign the field device of the CPU 1516F by first clicking on the interface of the IM155-6PN HF in the network view and then connecting it to the PROFINET interface (X1) of the CPU 1516F.

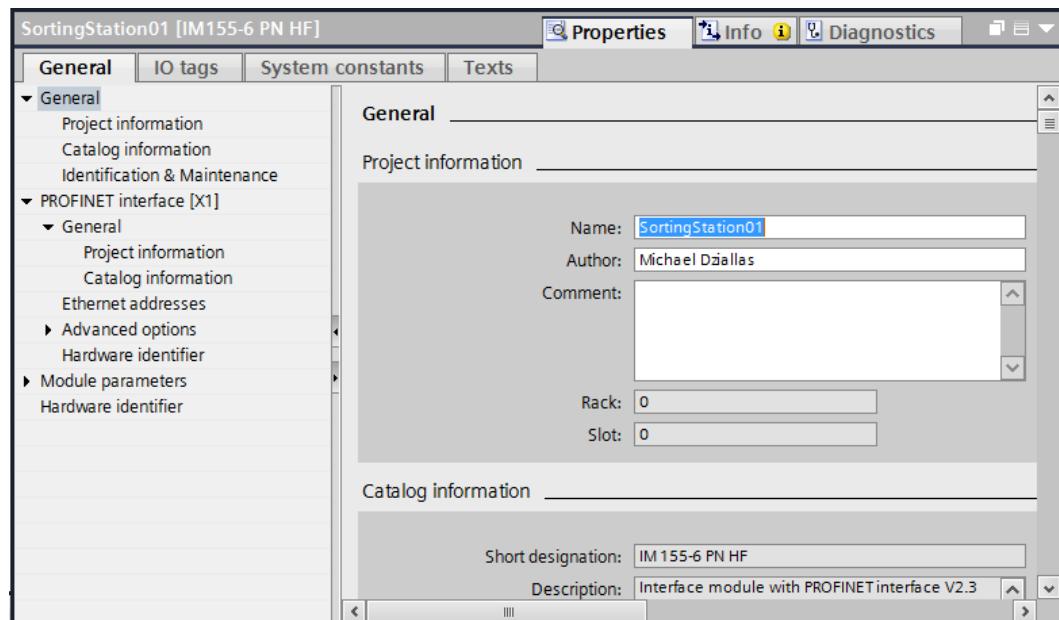


7.8 Configure the ET 200SP / IM 155-6PN HF

- To configure the IM155-6PN HF, first select the IO device. (→ IO device 1)

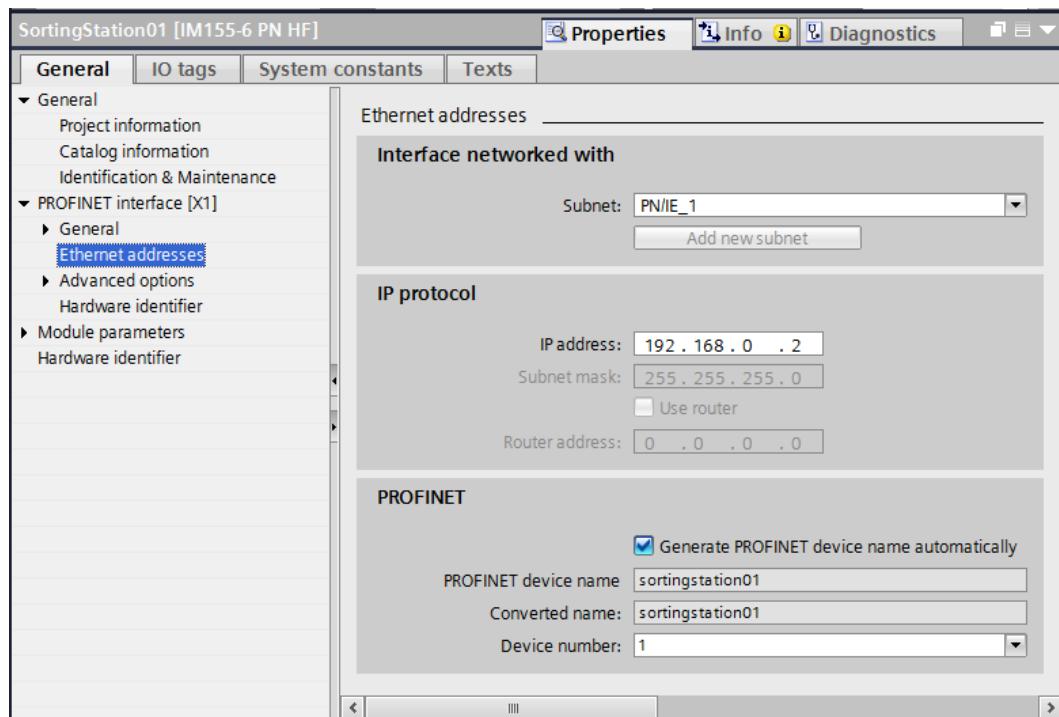


→ In → "Properties", open the → "General" menu and enter the "Name" → "SortingStation01".

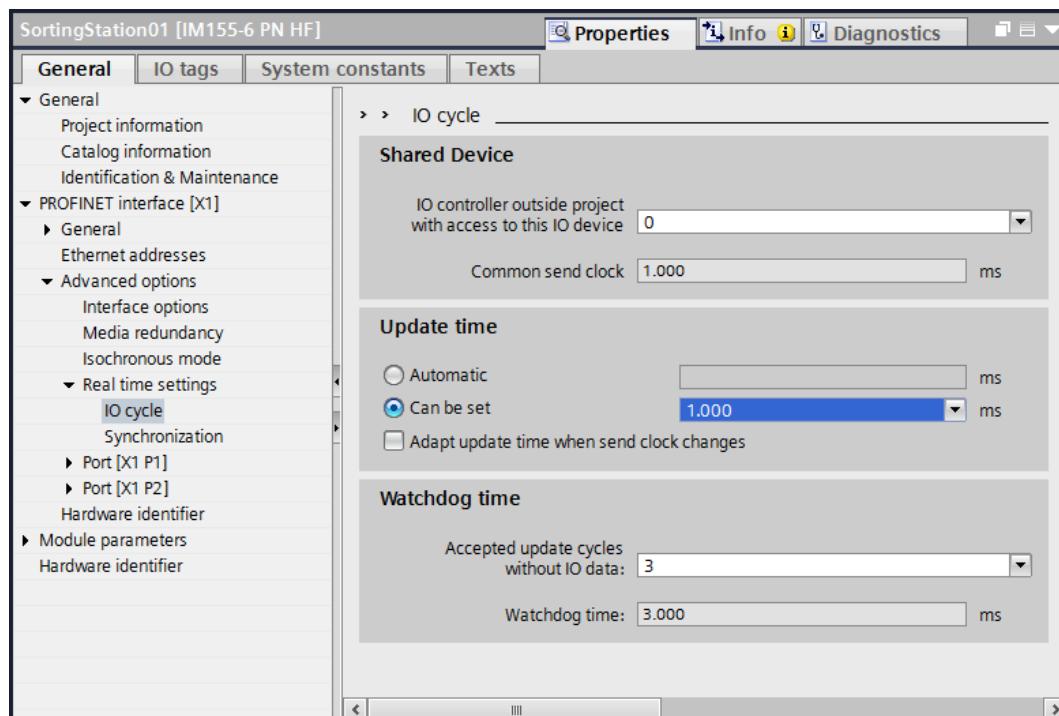


Note: This name is then applied as the device name for PROFINET communication.

- Under "PROFINET device name interface[X1]", the IP address can then be set or the IO device and the "PROFINET device name" can be checked.
(→ PROFINET interface[X1] → Ethernet addresses → IP protocol → IP address:
192.168.0.2 → PROFINET → PROFINET device name → SortingStation01)

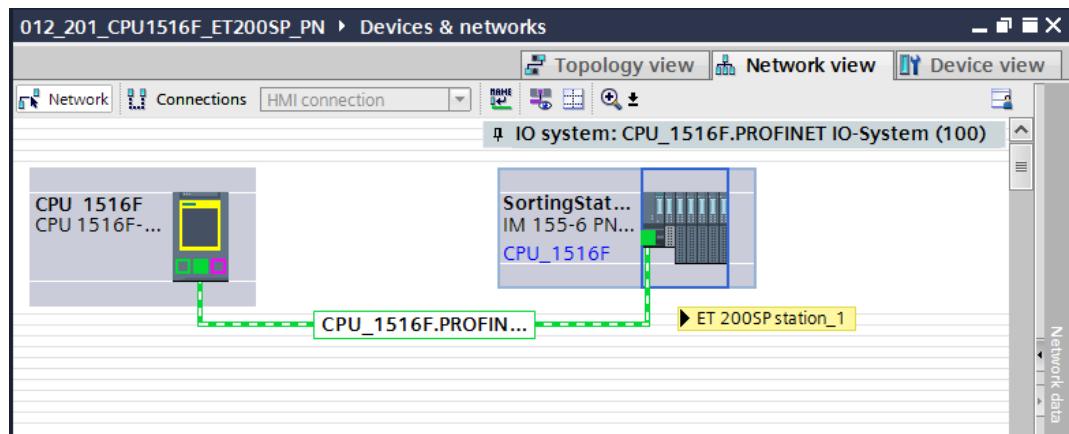


- The settings for the 'IO cycle' such as 'Update time' and 'Watchdog time' can also be set here for the device.
 (→ PROFINET interface[X1] → Advanced options → Real time settings → IO cycle → Update time → 1,000 ms → Watchdog time → 3,000 ms)

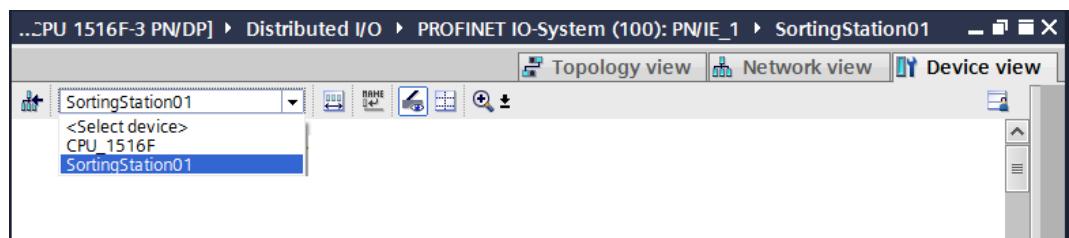


7.9 Insert the 2 digital input modules DI 8x24VDC HF

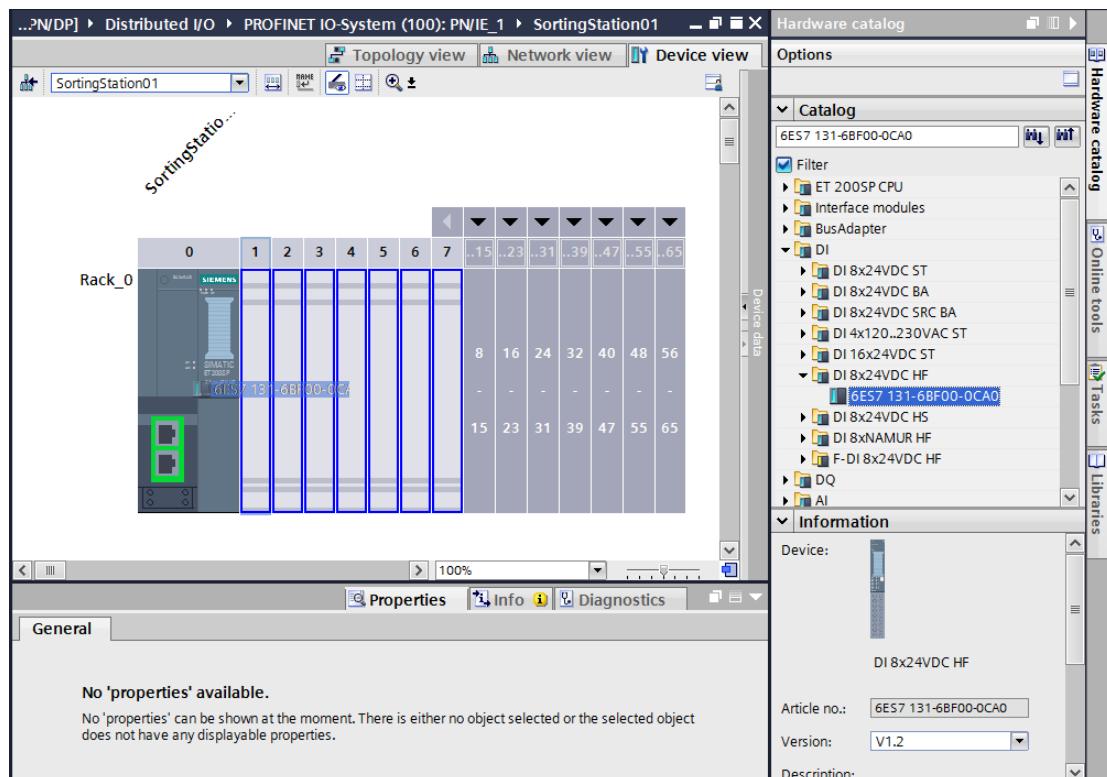
- Double-click the PROFINET device to change to the device view of the ET 200SP.



- **Note:** You can also open the device view of the different devices in the drop-down menu at the top left of the device view.



- Find the correct digital input module with the matching order number and version in the hardware catalog. Insert the digital input module into slot 1. (→Hardware Catalog → DI → DI 8x24VDC HF → 6ES7 131-6BF00-0CA0 → Version: V1.2)



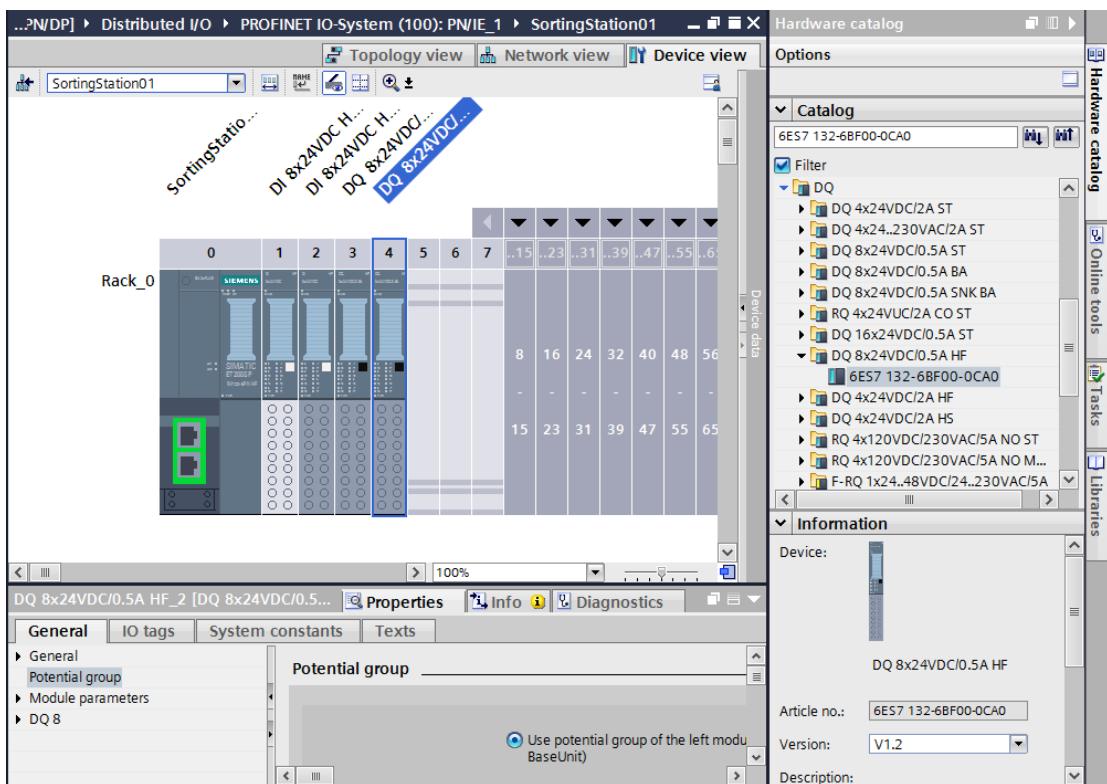
Notes:

When you double-click a module in the hardware catalog, you insert it automatically at the next available compatible slot. Insert a digital input module of the same type into slot 2

If you do not use a slot, you must close the gap before you compile. Otherwise, an error message occurs.

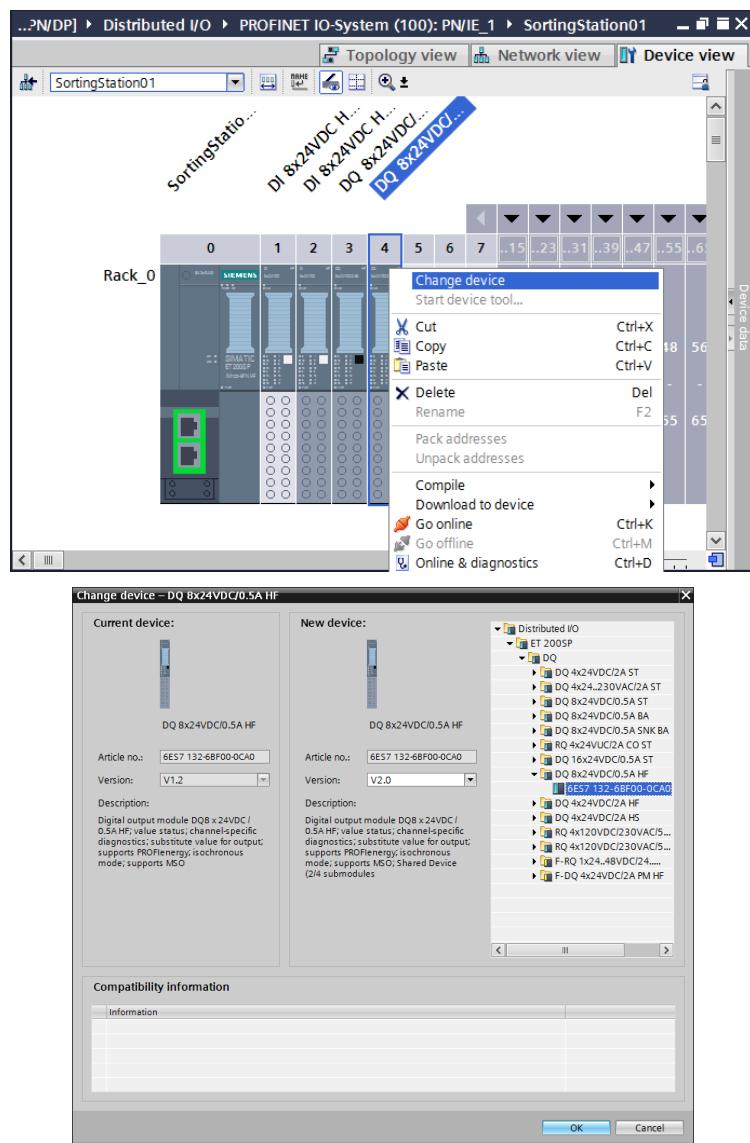
7.10 Insert the 2 digital output modules DQ 8x24VDC/0.5A HF

- Find the correct digital output module with the matching order number and version in the hardware catalog. Insert two digital output modules into slots 3 and 4. (→ Hardware catalog → DQ → DQ 8x24VDC/0.5A HF → (6ES7 132-6BF00-0CA0) → Version: V1.2)



7.11 Replace components in the hardware configuration

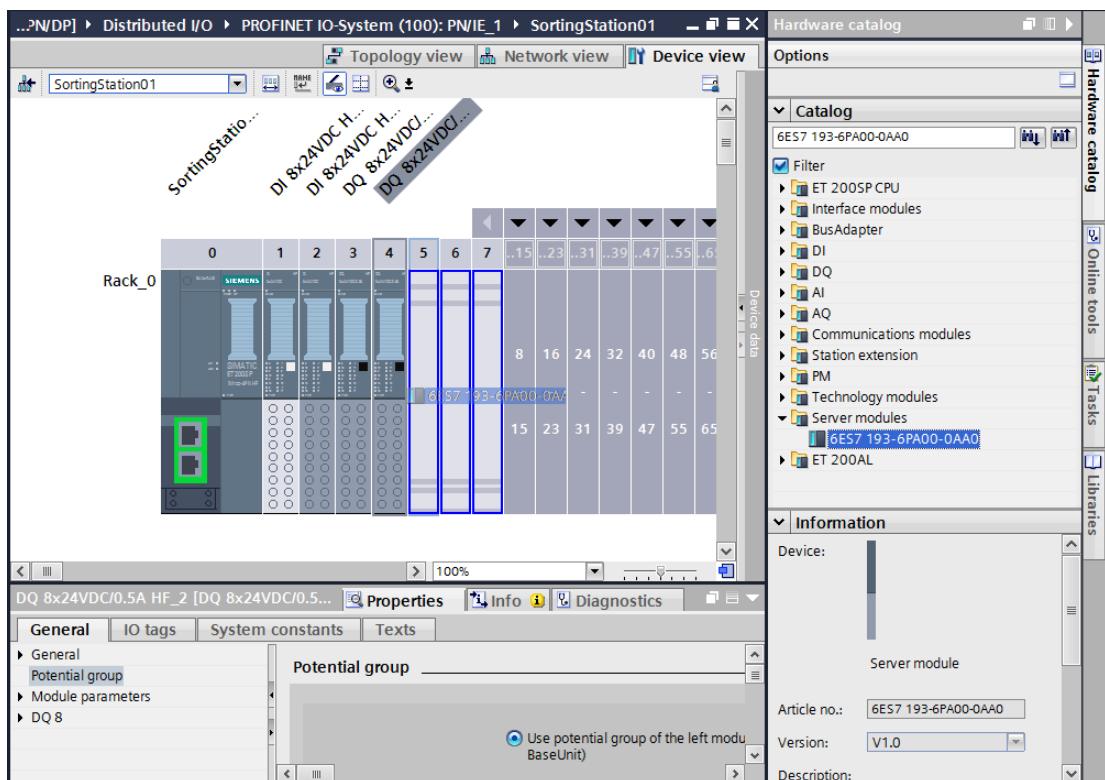
- If it becomes necessary to replace a component in the hardware configuration with a newer version or a different type, this can be done by right-clicking the component and selecting "Change device". The new replacement component can be selected in the displayed dialog and the selection can be applied with "OK". (→ Change device → OK)



Note: If the new component is not displayed for selection, it is not compatible with the previous component. In this case, the old component must be deleted and the new component must then be inserted from the hardware catalog.

7.12 Insert the server module

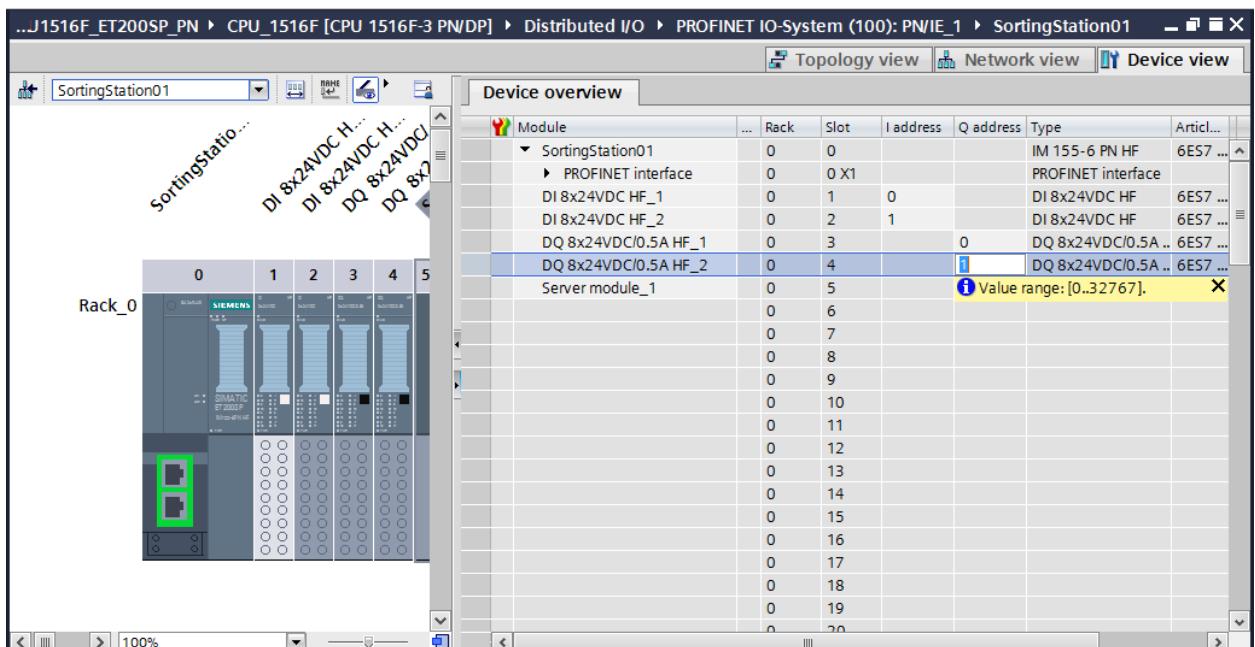
- Find the correct server module with matching order number and version in the hardware catalog. Insert the server module into slot 5.
(→ Hardware Catalog → Server module → 6ES7 193-6PA00-0AA0 → Version: V1.0)



Note: If you forget the server module, it is automatically created when the device configuration is compiled.

7.13 Configure the address areas DI/DQ: 0...1

- The next step is to check the address areas of the inputs and output cards and adapt them if necessary. Inputs and outputs (DI/DQ) should have an address area of 0...1.
(→ Device overview → DI → I addresses: 0/1 → DQ → O addresses: 0/1)

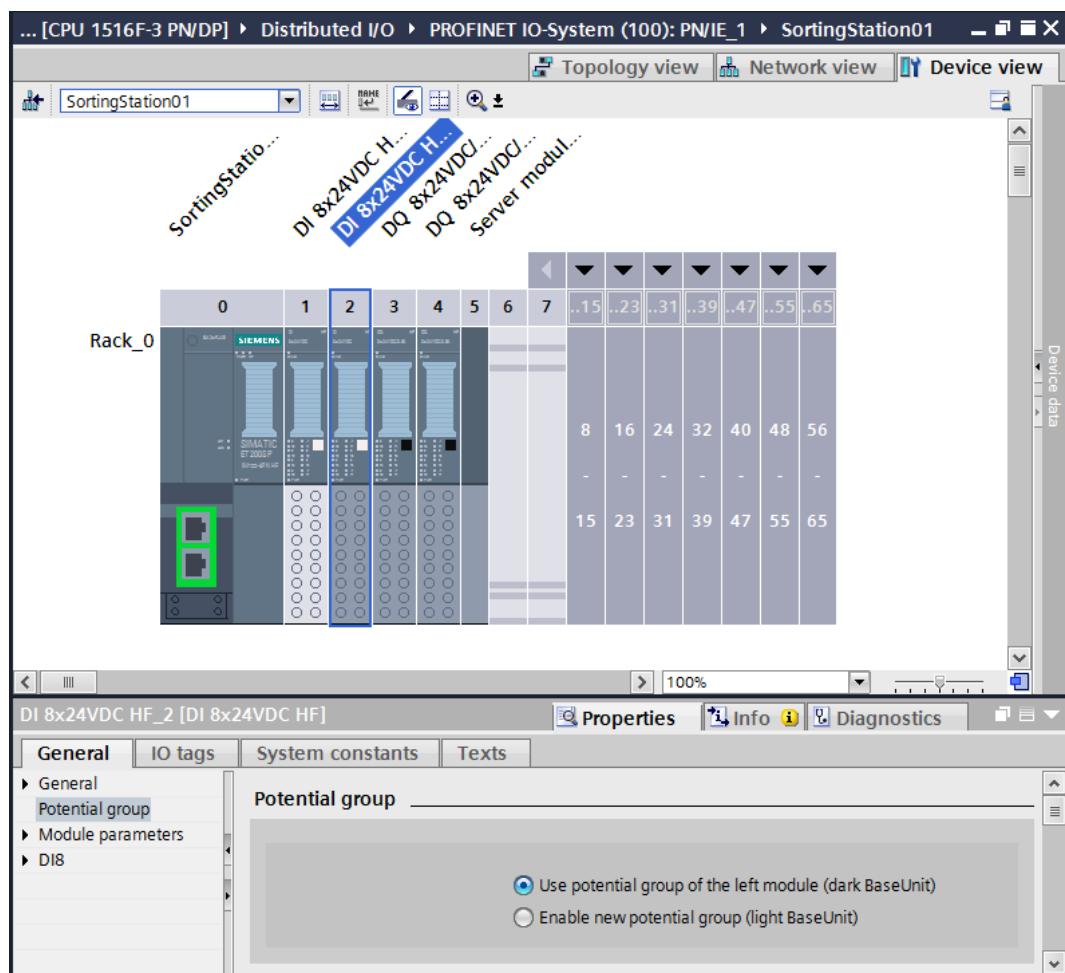


Note: To show and hide the Device overview, you must click the small arrow next to "Device data" on the right side of the hardware configuration.

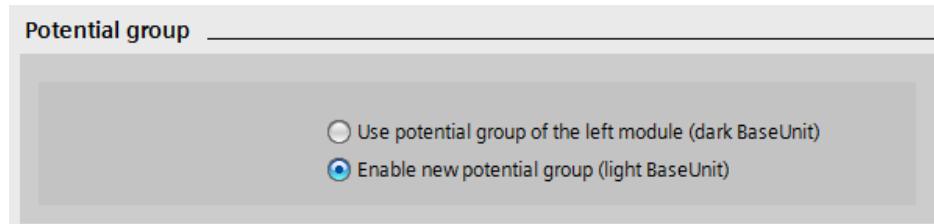


7.14 Configuration of the potential groups of the Base Units

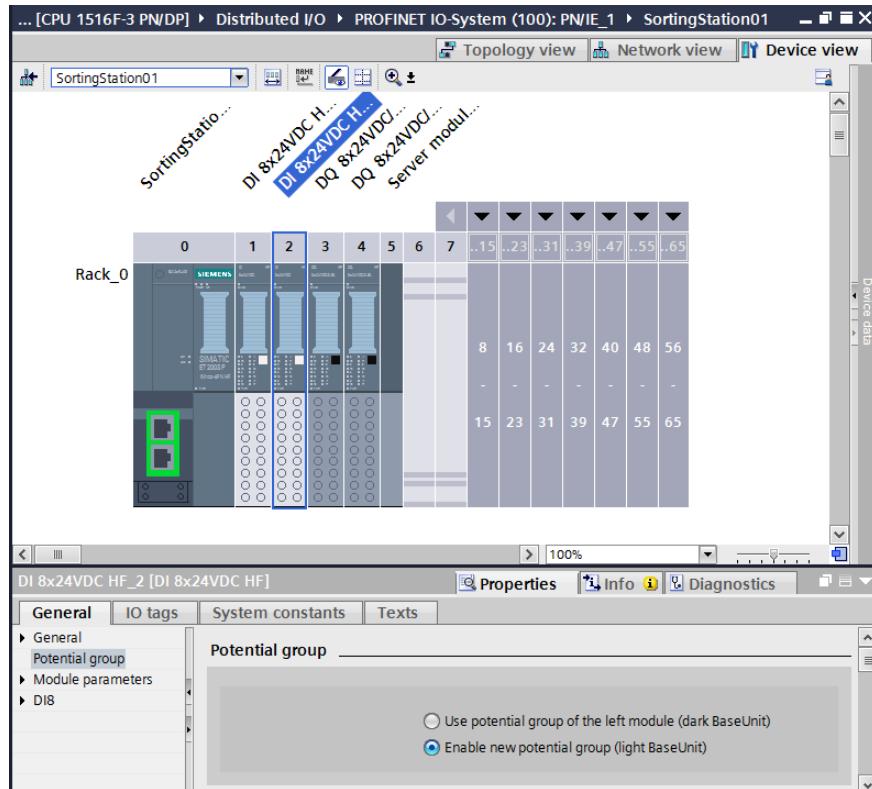
- To change the potential group of a Base Unit, select the associated module and open the Potential group section in the general properties.
- (Slot 2 → Properties → General → Potential group)



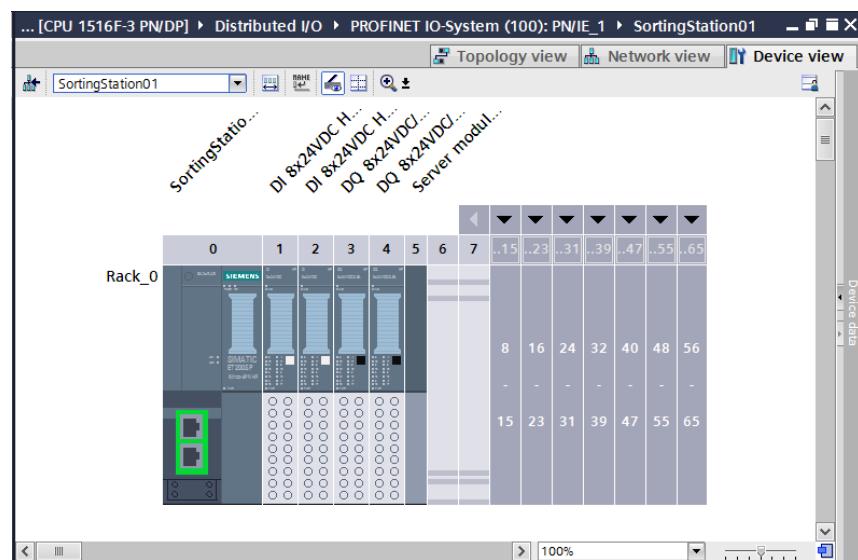
- Select the "Enable new potential group (light BaseUnit)" option.



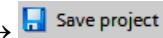
→ The Base Unit now becomes light in the configuration.

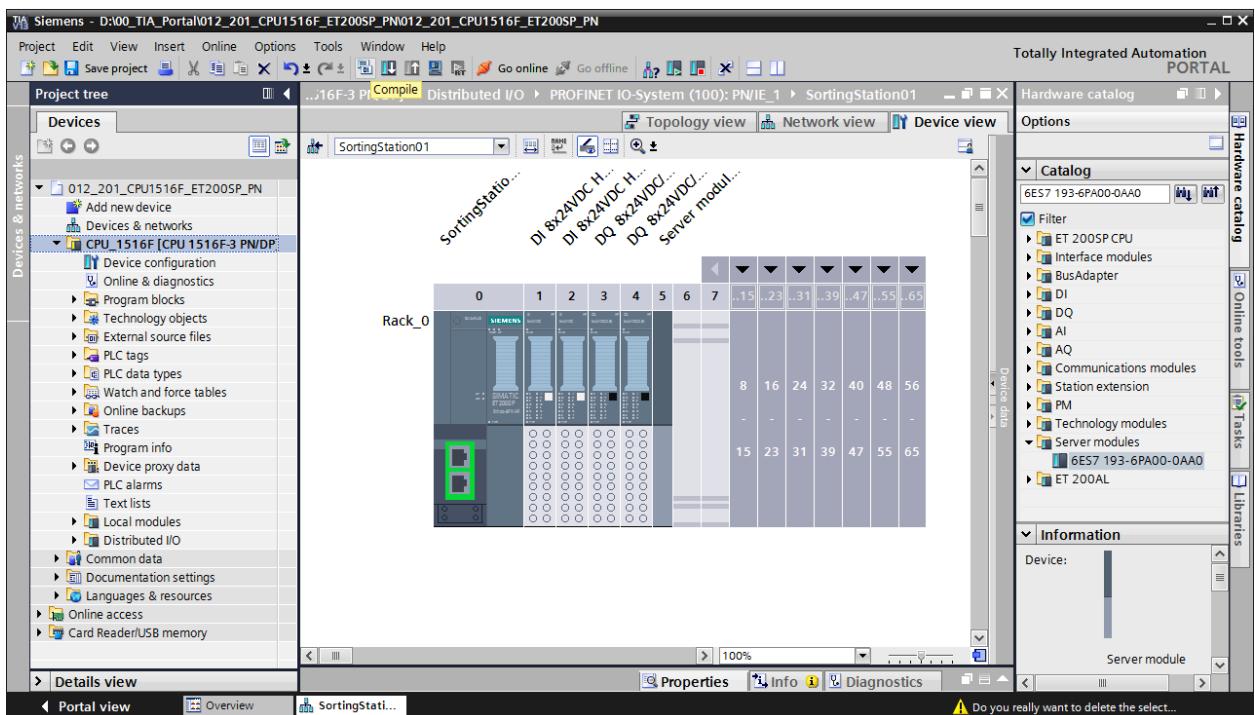


→ Repeat these steps for slots 3 to 4 and compare the device configuration with the following figure.



7.15 Save and compile the hardware configuration

- Before you compile the configuration, you should save your project by clicking the →  button. To compile your CPU with the device configuration, first select the → "CPU_1516F [CPU1516F-3 PN/DP]" folder and click the "Compile" icon → .



Note: "Save project" should be used repeatedly when working on a project since this does not happen automatically. A prompt to save the project only occurs when the TIA Portal is closed.

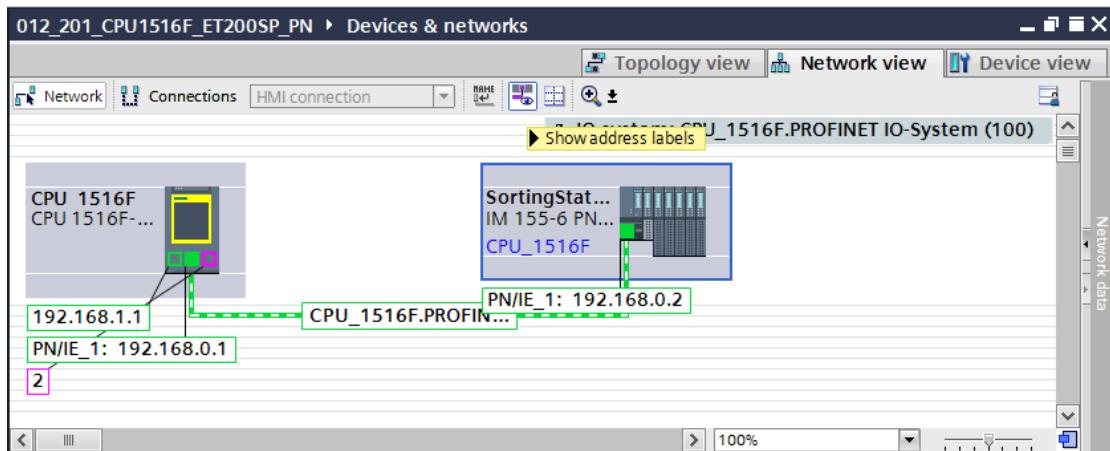
- If the project was compiled without errors, you see the following screen.

Properties							Info		Diagnostics	
General		Cross-references		Compile						
			Show all messages							
Compiling completed (errors:0; warnings:1)										
Path	Description	Go to	?	Errors	Warnings	Time				
CPU_1516F				0	1	11:57:43 PM				
Hardware configuration				0	1	11:57:43 PM				
S71500/ET200MP station_1				0	1	11:57:44 PM				
Rail_0				0	1	11:57:44 PM				
CPU_1516F				0	1	11:57:44 PM				
CPU_1516F	CPU_1516F does not contain a configured protection level			0	1	11:57:44 PM				
Program blocks				0	0	11:57:47 PM				
Main (OB1)	Block was successfully compiled.					11:57:47 PM				
	Compiling completed (errors:0; warnings:1)					11:57:49 PM				

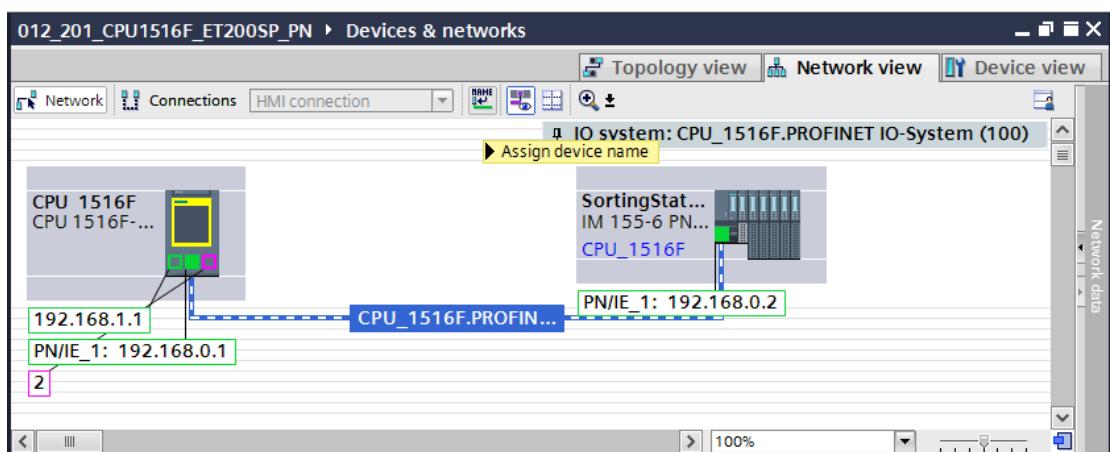
Note: A warning appears here because no protection level has been configured. This warning can be ignored.

7.16 Assign device name to interface module IM 155-6PN HF

- To obtain an overview of the assigned addresses within a project, you can click the " " icon in the "Network view".
(→ Network view → Show addresses)

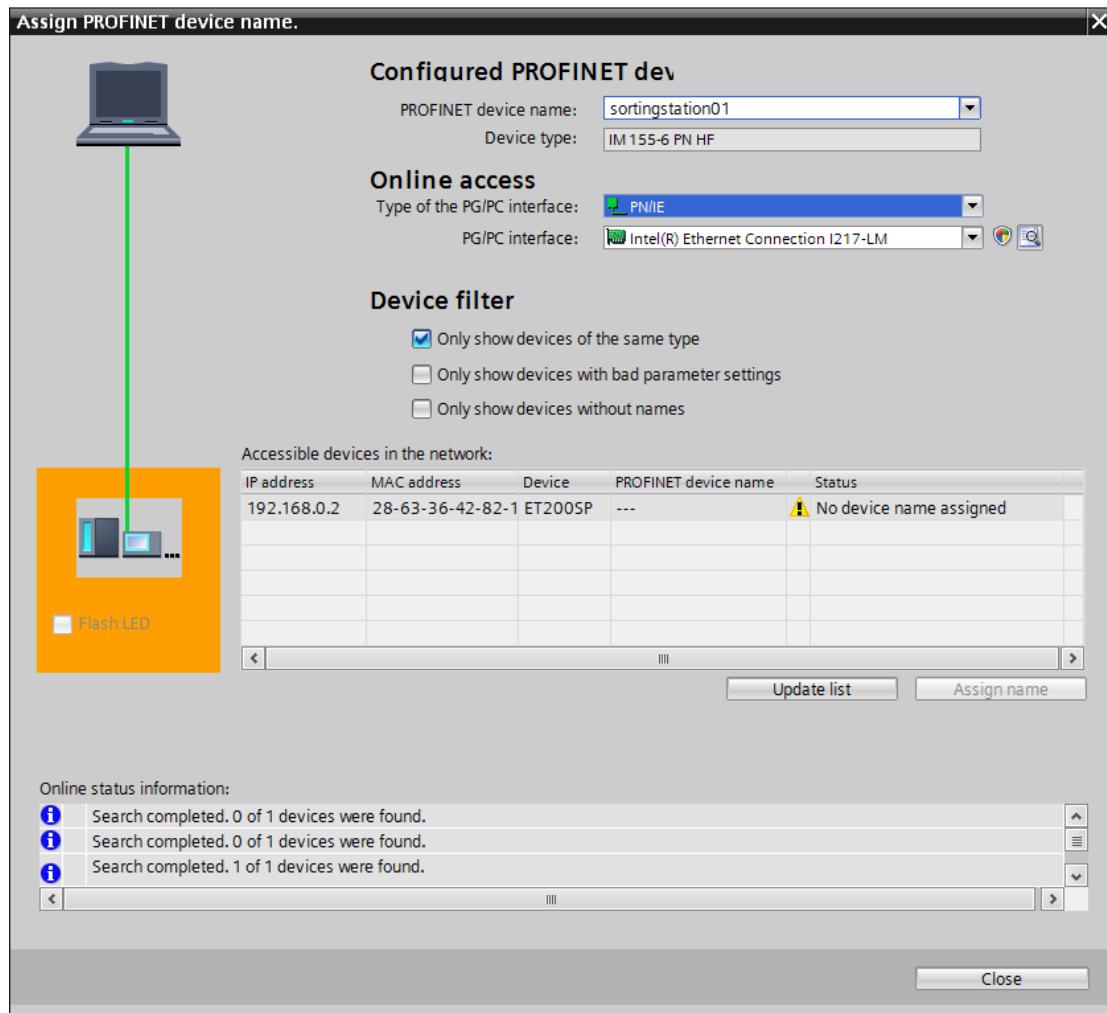


- In order for the controller, CPU1516F-3 PN/DP in this case, to find the assigned PROFINET devices in the network, the device name must be assigned to them. This is done by selecting the network connecting the devices in the "Network view" and clicking the " " icon.
(→ Assign device name)



Note: The IP addresses set in the project are assigned by the controller to the devices later when the communication connection is established.

- Online access must be correctly set in the dialog for assignment of the PROFINET device names. Any device can then be selected individually and filtered by devices of the same type. If a new device is being connected for the first time, the list has to be updated again.
 (→ PROFINET device name: sortingstation01 → Type of PG/PC interface: PN/IE → PG/PC interface: here: Intel(R) Ethernet Connection I217-LM → Only show devices of the same type → **Update list**)

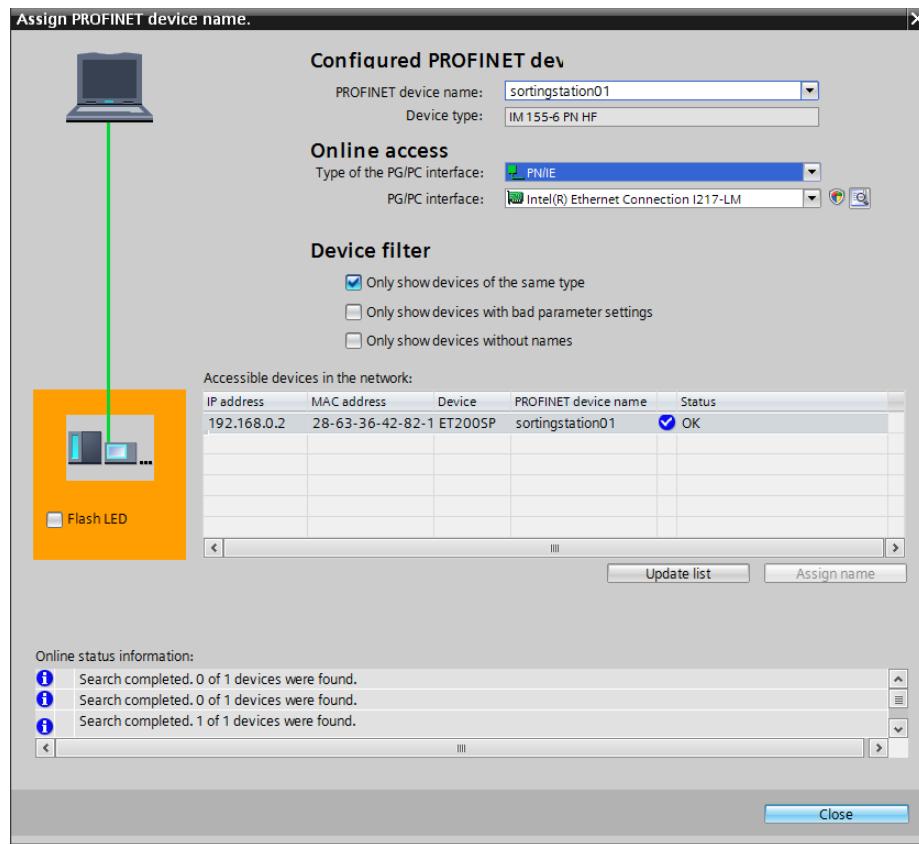


- The correct device must be always be clearly determined by the MAC address printed on the device before the name is assigned. To check this, you can also have the LEDs on the device flash.

(→ Flash LED → **Assign name**)

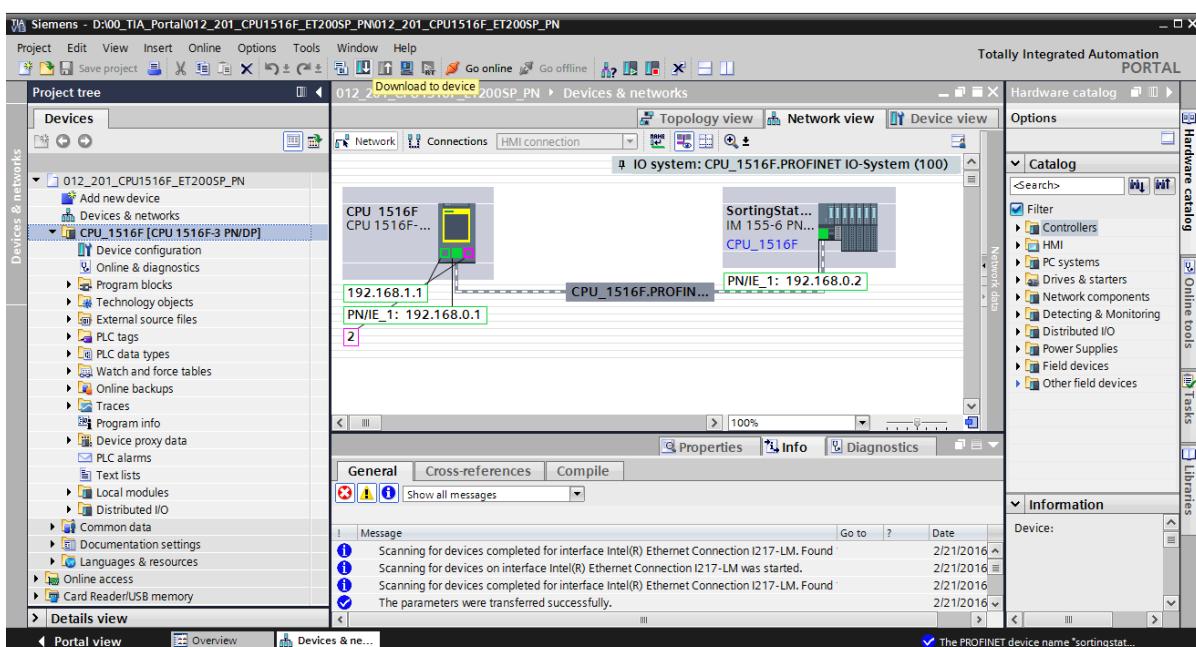
Accessible devices in the network:				
IP address	MAC address	Device	PROFINET device name	Status
192.168.0.2	28-63-36-42-82-1	ET200SP	---	No device name assigned

- Successful assignment of the PROFINET device name should be checked before closing the dialog. (→)

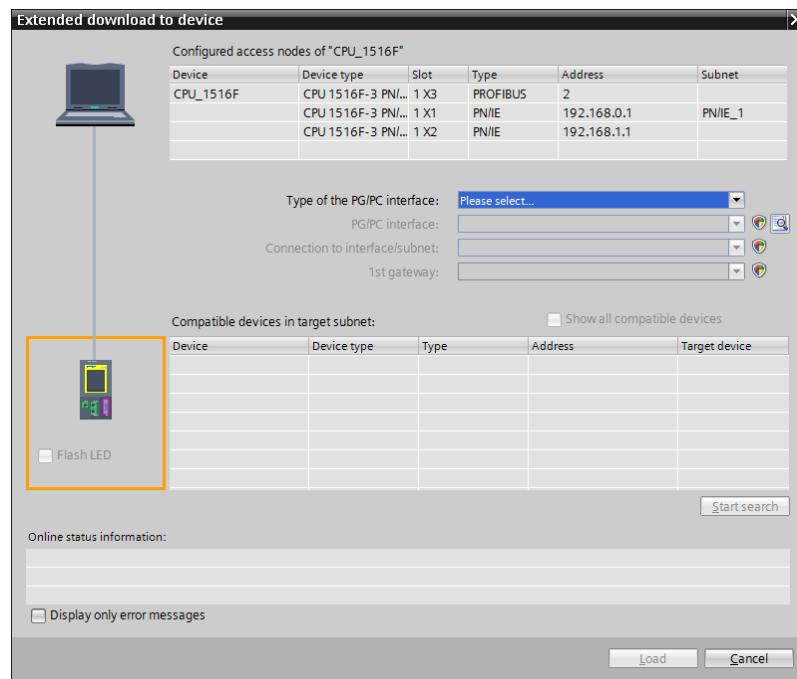


7.17 Download the hardware configuration to the device

- To download your entire CPU, select the → "CPU_1516F [CPU1516F-3 PN/DP]" folder and click the → "Download to device" icon.

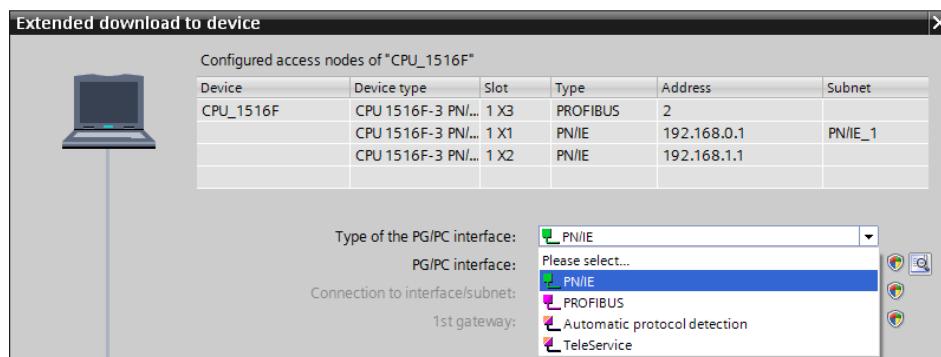


→ The manager for configuring the connection properties (extended download) opens.

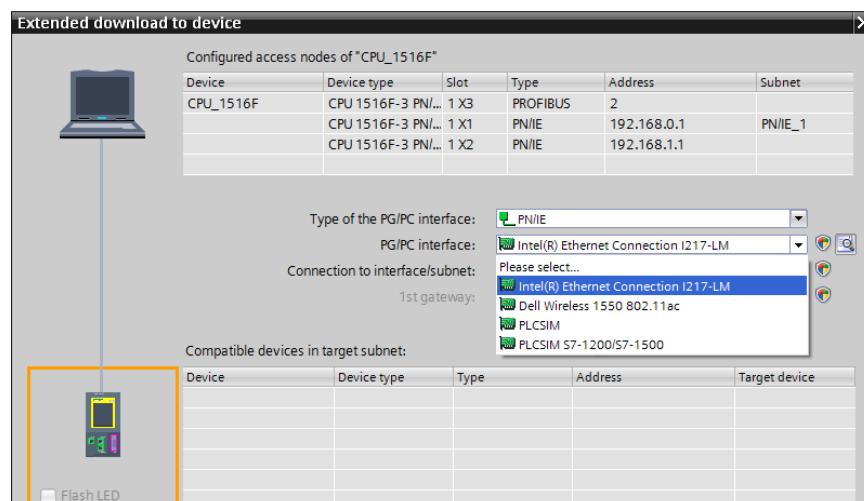


→ First, the interface must be correctly selected. This happens in three steps.

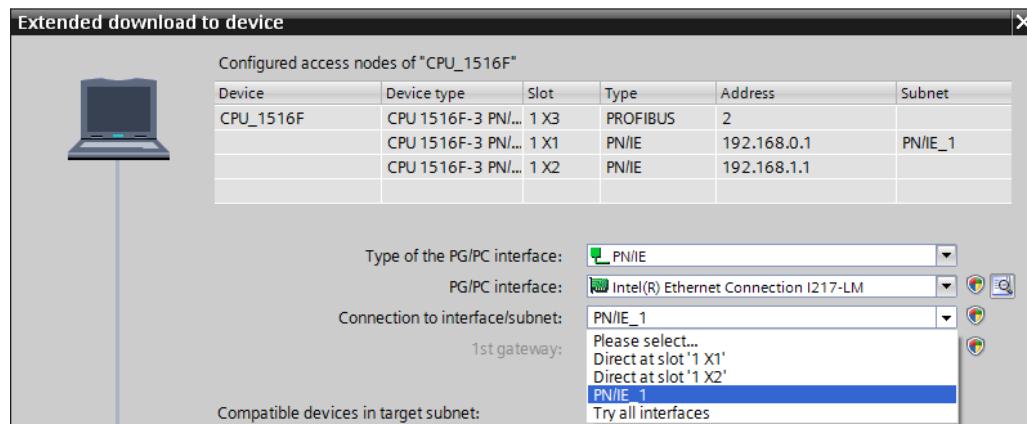
→ Type of the PG/PC interface → PN/IE



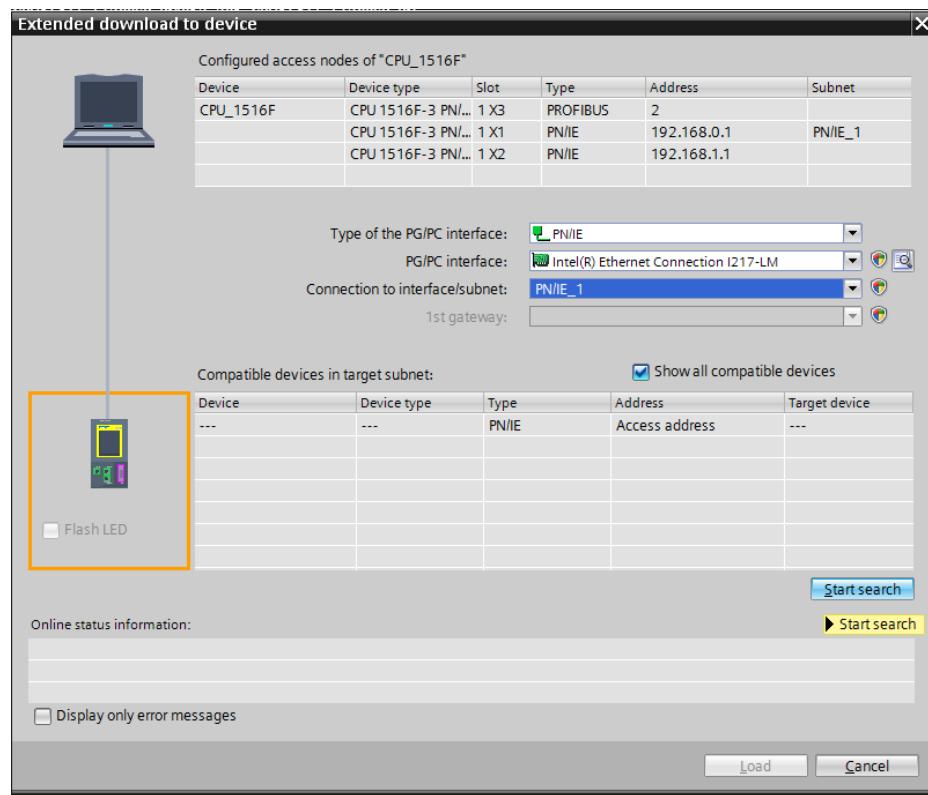
→ PG/PC interface → here: Intel(R) Ethernet Connection I217-LM



→ Connection to interface/subnet → "PN/IE_1"

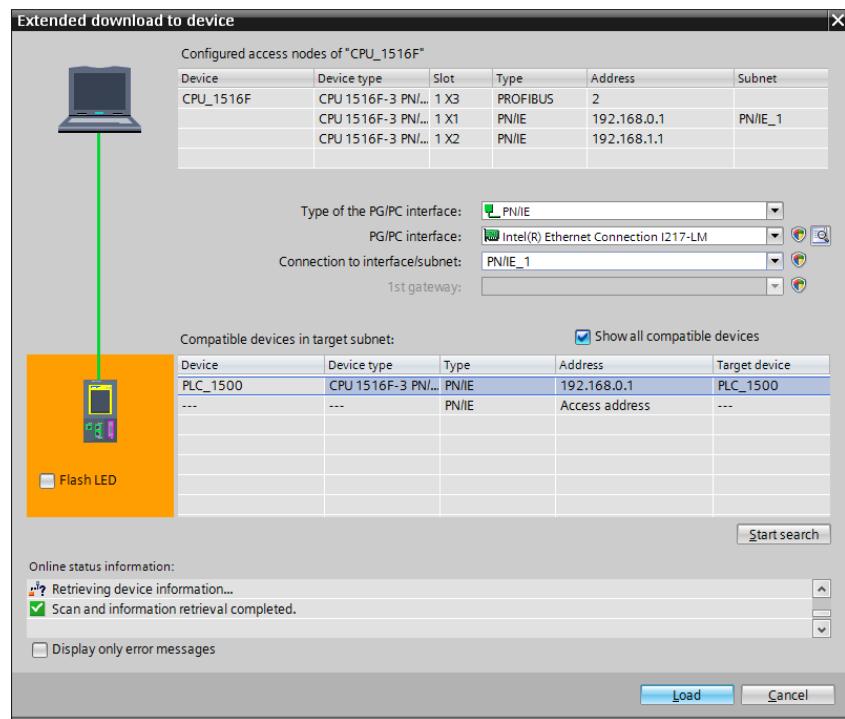


- The → "Show all compatible devices" check box must then be selected. The search for devices in the network is started by clicking the → **Start search** button.

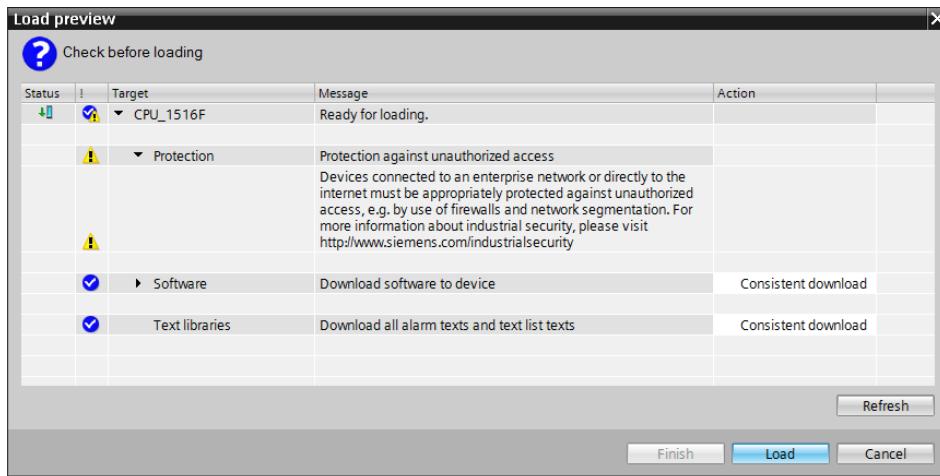


- If your CPU is shown in the "Compatible devices in target subnet" list, it must be selected and the download started.

(→ CPU 1516F-3 PN/DP → **Load**)

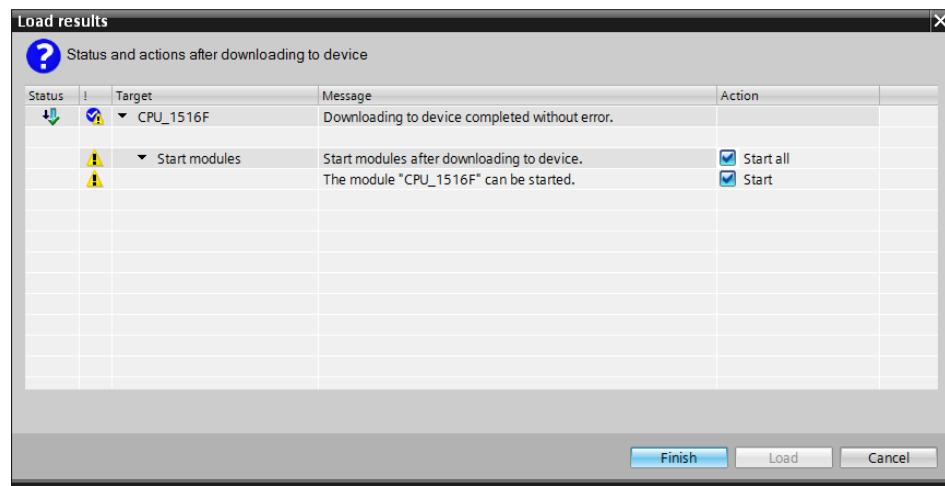


- You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → **Load**.

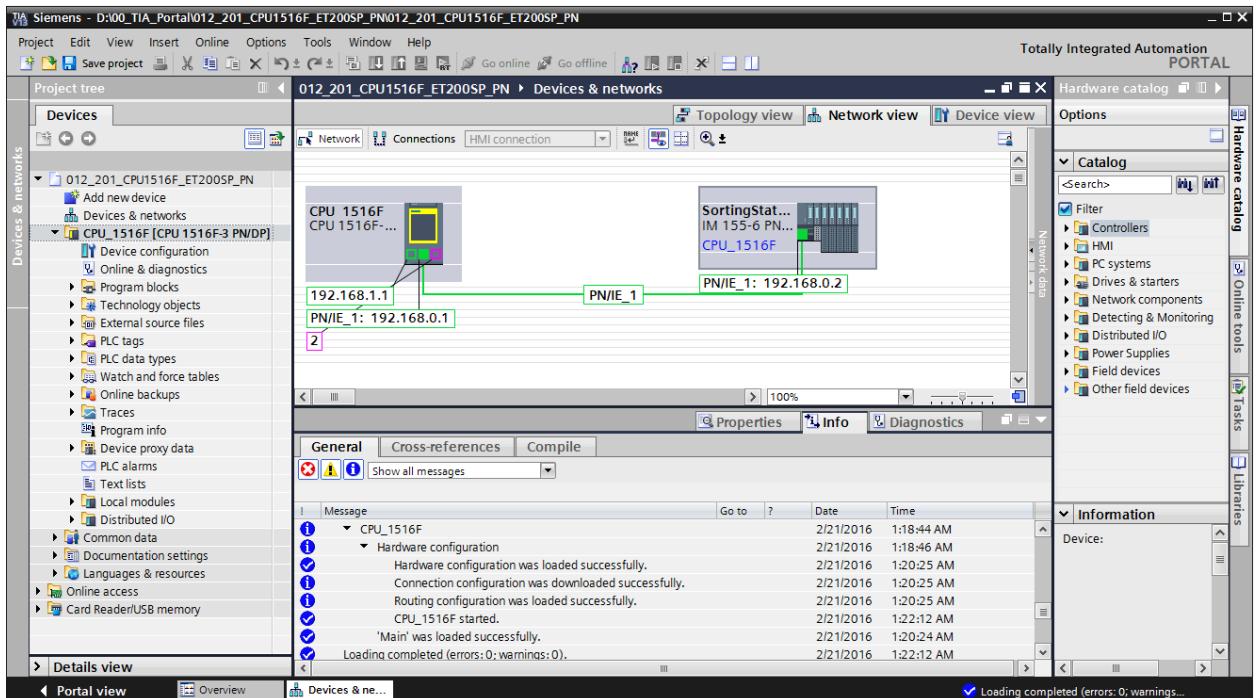


Note: The symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.

- The → "Start all" option will be selected next before the download operation can be completed with → **Finish**.



- After a successful download, the project view will open again automatically. A download report appears in the information field under "General". This can be helpful when troubleshooting an unsuccessful download.



7.18 Checklist

No.	Description	Completed
1	Project was created	
2	Slot 0: Power module with correct order number	
3	Slot 1: CPU with correct order number	
4	Slot 1: CPU with correct firmware version	
5	IM of the ET 200SP created as distributed IO	
6	CPU and IM connected to the same subnet	
7	IM is assigned to the CPU	
8	ET 200SP slot 1...2: Digital input module with correct order number	
9	ET 200SP slot 1...2: Digital input module with correct firmware version	
10	ET 200SP slot 1...2: Address area of the digital input module is correct	
11	ET 200SP slot 3...4: Digital output module with correct order number	
12	ET 200SP slot 3...4: Digital output module with correct firmware version	
13	ET 200SP slot 3...4: Address area of the digital output module is correct	
14	ET 200SP slot 5: Server module with correct order number	
15	ET 200SP slot 5: Server module with correct firmware version	
16	ET 200SP modules have all the correct potential groups set for the Base Units	
17	Hardware configuration was compiled without error message	
18	Hardware configuration was downloaded without error message	
19	Project was successfully archived	



Training Curriculum

TIA Portal Module 004
Basics of FC Programming
with SIMATIC S7-1500

Table of contents

1	Goal.....	139
2	Prerequisite.....	139
3	Required hardware and software.....	139
4	Theory	140
4.1	Operating system and application program.....	140
4.2	Organization blocks.....	140
4.3	Process image and cyclic program processing	142
4.4	Functions	143
4.5	Function blocks and instance data blocks.....	143
4.6	Global data blocks.....	144
4.7	Library-compatible code blocks	145
4.8	Programming languages	146
5	Task.....	147
6	Planning.....	147
6.1	EMERGENCY STOP	147
6.2	Manual mode – Conveyor motor in manual mode	147
7	Structured step-by-step instructions.....	148
7.1	Retrieve an existing project	148
7.2	Create a new tag table	149
7.3	Create new tags within a tag table.....	150
7.4	Import "Tag_table_sorting_station"	151
7.5	Create function FC1 "MOTOR_MANUAL" for the conveyor motor in manual mode	153
7.6	Define the interface of function FC1 "MOTOR_MANUAL"	154
7.7	Program FC1: MOTOR_MANUAL	156
7.8	Program organization block OB1 – Control of the forward belt tracking in manual mode	162
7.9	Program organization block OB1 – Control of the backward belt tracking in manual mode	167
7.10	Save and compile the program.....	168
7.11	Download the program	169
7.12	Monitor program blocks.....	170
8	Checklist	173

BASICS OF FC PROGRAMMING

1 Goal

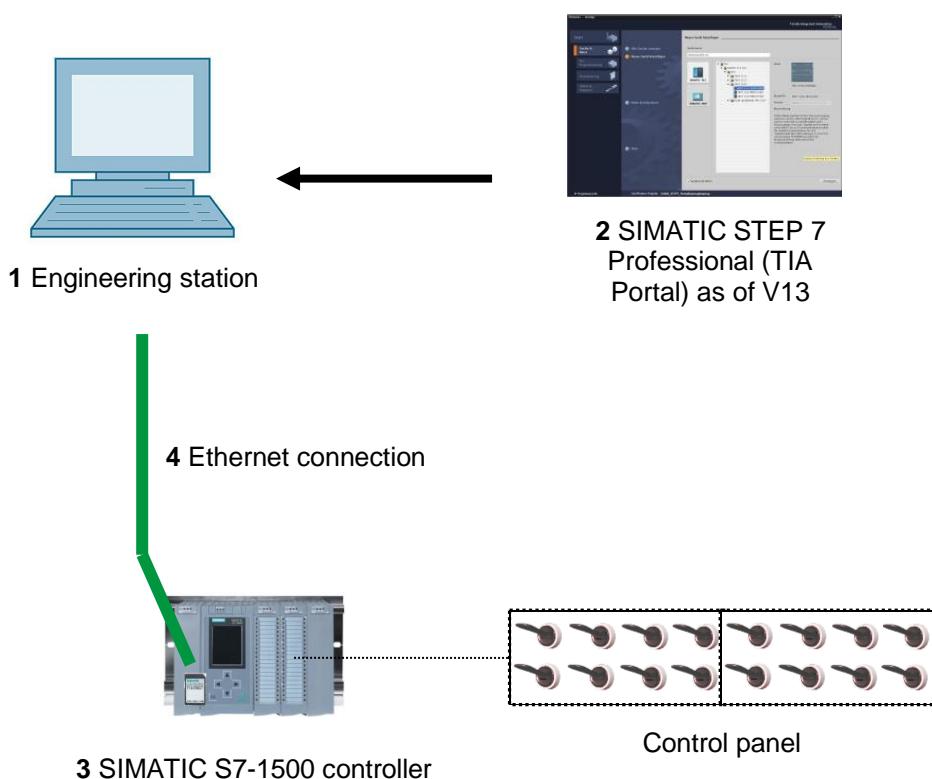
In this chapter, you will get to know the basic elements of a control program – the **organization blocks (OBs)**, **functions (FCs)**, **function blocks (FBs)** and **data blocks (DBs)**. In addition, you will be introduced to **library-compatible** function und function block programming. You will get to know the **Function Block Diagram (FBD)** programming language and use it to program a function (FC1) and an organization block (OB1).

2 Prerequisite

This chapter builds on the hardware configuration of SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations that have digital input and output cards can be used.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V13
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 Operating system and application program

Every controller (CPU) contains an ***operating system***, which organizes all functions and processes of the CPU that are not associated with a specific control task. The tasks of the operating system include the following:

- Performing a warm restart
- Updating the process image of the inputs and outputs
- Cyclically calling the user program
- Detecting interrupts and calling interrupt OBs
- Detecting and handling errors
- Managing memory areas

The operating system is an integral component of the CPU and comes pre-installed.

The ***user program*** contains all functions that are necessary for executing your specific automation task. The tasks of the user program include the following:

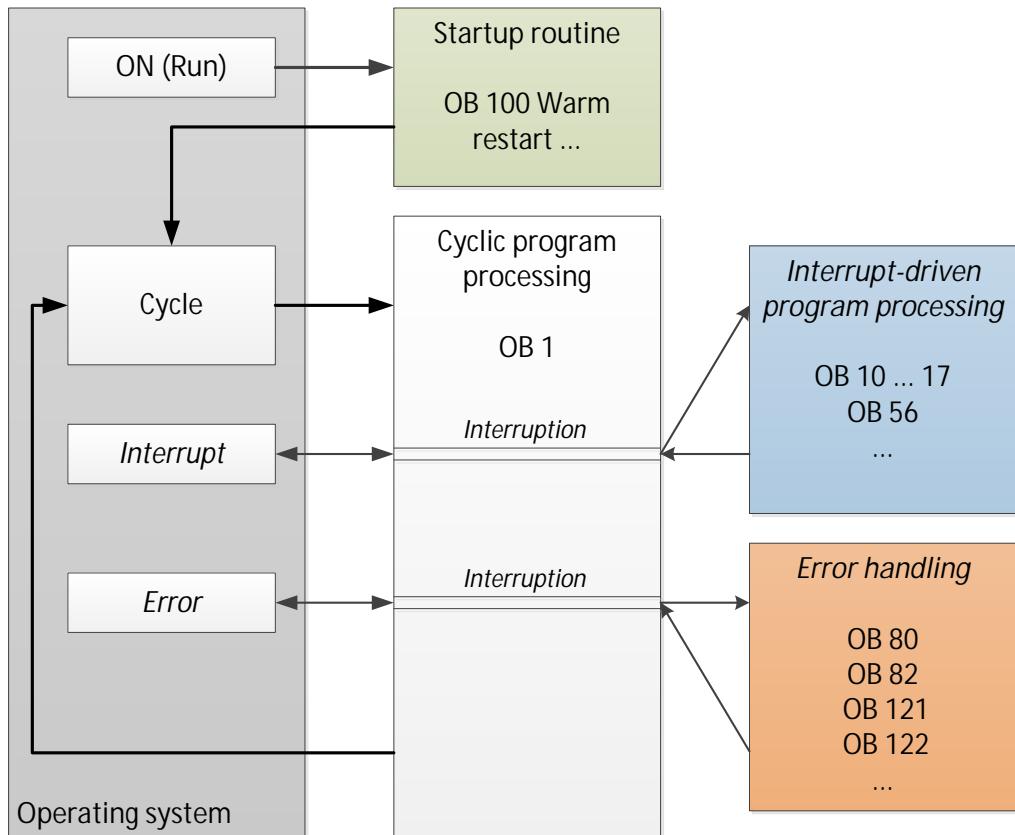
- Checking the basic requirements for a warm restart using startup OBs
- Processing of process data, i.e. activation of output signals as a function of the input signal states
- Reaction to interrupts and interrupt inputs
- Error handling during normal program execution

4.2 Organization blocks

Organization blocks (OBs) form the interface between the operating system of the controller (CPU) and the application program. They are called from the operating system and control the following operations:

- Cyclic program processing (e.g. OB1)
- Startup characteristics of the controller
- Interrupt-driven program processing
- Error handling

A project must have ***an organization block for cyclic program processing*** at a minimum. An OB is called by a ***start event*** as shown in Figure 1. In addition, the individual OBs have defined priorities so that, for example, an OB82 for error handling can interrupt the cyclic OB1.



Figure

1: Start events in the operating system and OB call

When a start event occurs, the following reactions are possible:

- If an OB has been assigned to the event, this event triggers the execution of the assigned OB. If the priority of the assigned OB is greater than the priority of the OB that is currently being executed, it is executed immediately (interrupt). If not, the assigned OB waits until the higher-priority OB has been completely executed.
- If an OB is not assigned to the event, the default system reaction is performed.

Table 1 gives a couple of examples of start events for a SIMATIC S7-1500, their possible OB number(s) and the default system reaction in the event the organization block is not present in the controller.

Start event	Possible OB numbers	Default system reaction
Startup	100, ≥ 123	Ignore
Cyclic program	1, ≥ 123	Ignore
Time-of-day interrupt	10 to 17, ≥ 123	-
Update interrupt	56	Ignore
Scan cycle monitoring time exceeded once	80	STOP
Diagnostic interrupt	82	Ignore
Programming error	121	STOP
IO access error	122	Ignore

Table 1: OB numbers for various start events

4.3 Process image and cyclic program processing

When the cyclic user program addresses the inputs (I) and outputs (O), it does not query the signal states directly from the input/output modules. Instead, it accesses a memory area of the CPU. This memory area contains an image of the signal states and is called the **process image**.

The cyclic program processing sequence is as follows:

1. At the start of the cyclic program, a query is sent to determine whether or not the individual inputs are energized. This status of the inputs is stored in the **process image of the inputs (PII)**. In doing so, the information 1 or "High" is stored for energized inputs and the information 0 or "Low" for de-energized inputs.
2. The CPU now executes the program stored in the cyclic organization block. For the required input information, the CPU accesses the previously read **process image of the inputs (PII)** and the results of logic operation (RLOs) are written to a so-called **process image of the outputs (PIQ)**.
3. At the end of the cycle, the **process image of the outputs (PIQ)** is transferred as the signal state to the output modules and these are energized or de-energized. The sequence then continues again with Item 1.

1. Save status of inputs in PII.

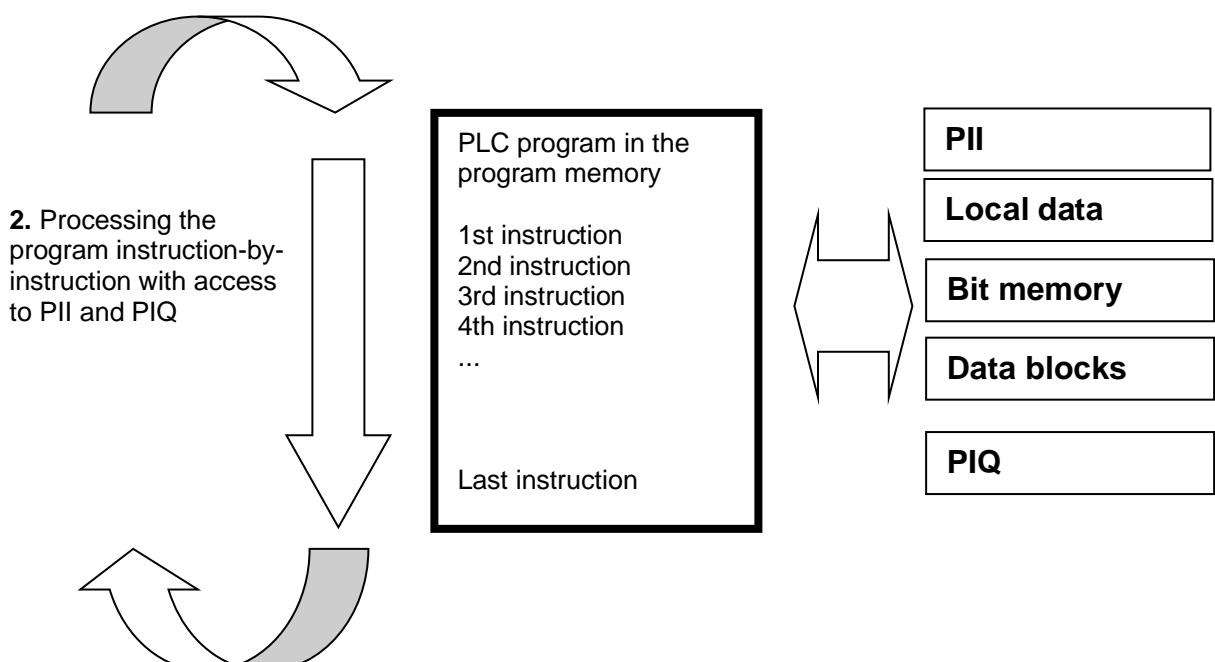


Figure 2: Cyclic program processing

Note: The time the CPU needs for this sequence is called *cycle time*. This depends, in turn, on the number and type of instructions and the processor performance of the controller.

4.4 Functions

Functions (FCs) are logic blocks without memory. They **have no data memory** in which values of block parameters can be stored. Therefore, all interface parameters must be connected when a function is called. To store data permanently, global data blocks must be created beforehand.

A function contains a program that is executed whenever the function is called from another code block.

Functions can be used, for example, for the following purposes:

- Math functions – that return a result dependent on input values.
- Technological functions – such as individual controls with binary logic operations.

A function can also be called several times at different points within a program.

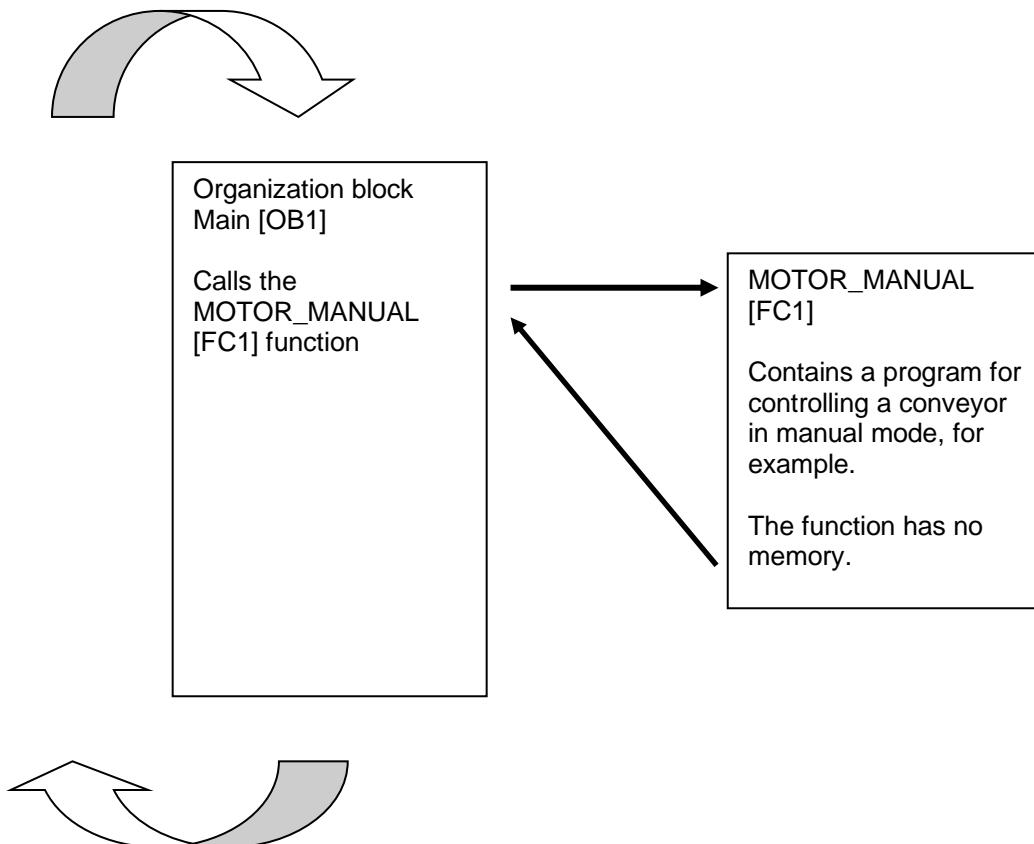


Figure 3: Function with call from organization block Main [OB1]

4.5 Function blocks and instance data blocks

Function blocks are code blocks that store their input, output and in-out tags as well as static tags permanently in instance data blocks, so that they **are available after the block has been executed**. For this reason, they are also referred to as blocks with "memory".

Function blocks can also operate with temporary tags. Temporary tags are not stored in the instance DB, however. Instead, they are only available for one cycle.

Function blocks are used for tasks that cannot be implemented with functions:

- Whenever timers and counters are required in the blocks.
- Whenever information must be saved in the program, such as pre-selection of the operating mode with a button.

Function blocks are always executed when called from another code block. A function block can also be called several times at different points within a program. This facilitates the programming of frequently recurring complex functions.

A call of a function block is referred to as an instance. Each instance of a function block is assigned a memory area that contains the data that the function block uses. This memory is made available by data blocks created automatically by the software.

It is also possible to provide memory for multiple instances in one data block in the form of a **multi-instance**. The maximum size of instance data blocks varies depending on the CPU. The tags declared in the function block determine the structure of the instance data block.

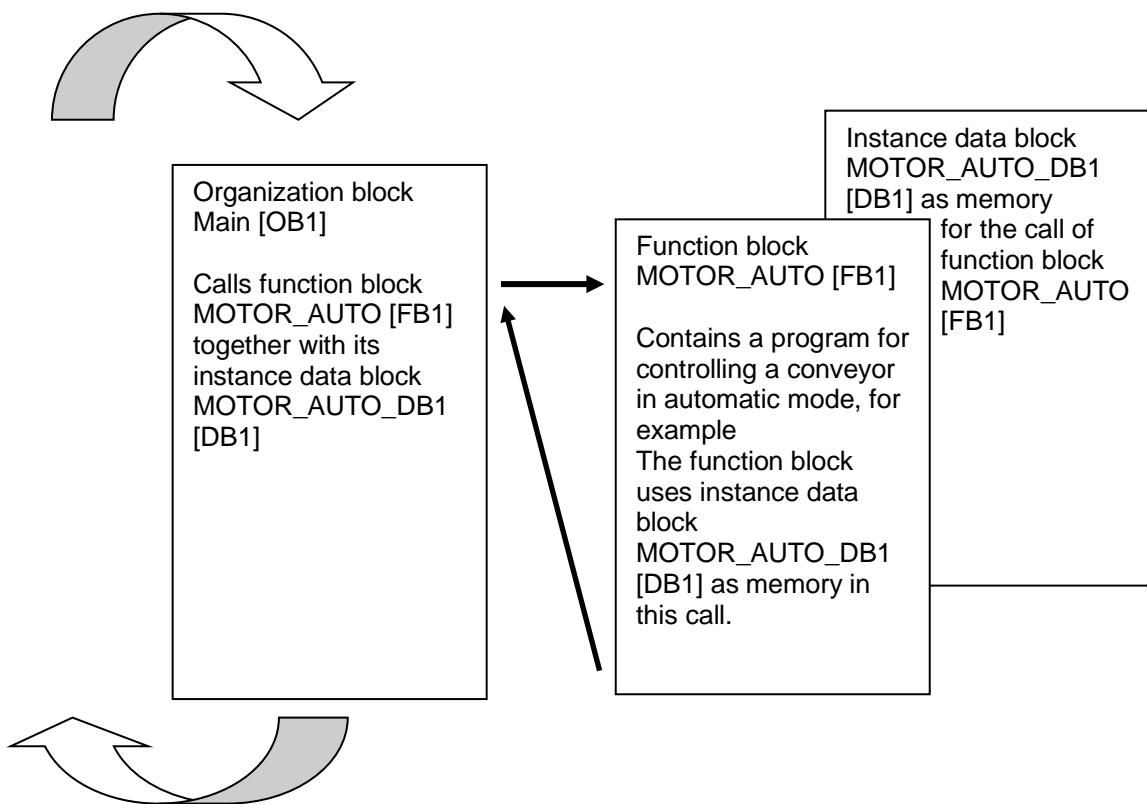


Figure 4: Function block and instance with call from organization block Main [OB1]

4.6 Global data blocks

In contrast to logic blocks, data blocks contain no instructions. Rather, they serve as memory for user data.

Data blocks thus contain variable data that is used by the user program. You can define the structure of global data blocks as required.

Global data blocks store data that can be used **by all other blocks** (see Figure 5). Only the associated function block should access instance data blocks. The maximum size of data blocks varies depending on the CPU.

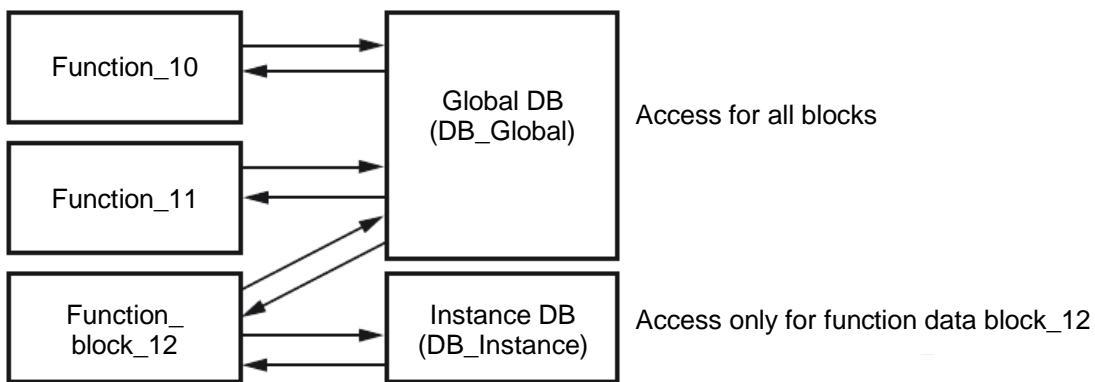


Figure 5: Difference between global DB and instance DB.

Application examples for **global data blocks** are:

- Saving of information about a storage system. "Which product is located where?"
- Saving of recipes for particular products.

4.7 Library-compatible code blocks

A user program can be created with linear or structured programming. **Linear programming** writes the entire user program in the cycle OB, but is only suitable for very simple programs for which other less expensive control systems, such as LOGO!, can now be used.

Structured programming is always recommended for more complex programs. Here, the overall automation task can be broken down into small sub-tasks in order to implement a solution for them in functions and function blocks.

In this case, library-compatible logic blocks should be created preferentially. This means that the input and output parameters of a function or function block are defined generally and only supplied with the current global tags (inputs/outputs) when the block is used.

MOTOR_MANUAL				
	Name	Data type	Default value	Comment
1	Input			
2	Manual_mode_active	Bool		Manual mode activated
3	Pushbutton_manual_mode	Bool		Pushbutton manual mode conveyor on
4	Enable_OK	Bool		All enable conditions OK
5	Safety_shutoff_active	Bool		Safety shutoff active e.g. emergency stop operated
6	Output			
7	Conveyor_motor_manual...	Bool		Control of the conveyor motor in manual mode

Block title:

Conveyor motor in manual mode: If the pushbutton_manual_mode is operated, the enable conditions are granted and the safety shutoff is not activated the output Conveyor_motor_manual_mode is activated

Network 1: Control of the conveyor motor in manual mode

```

Comment

#Manual_mode_active
#Pushbutton_manual_mode
#Enable_OK
#Safety_shutoff_active & = #Conveyor_motor_manual_mode

```

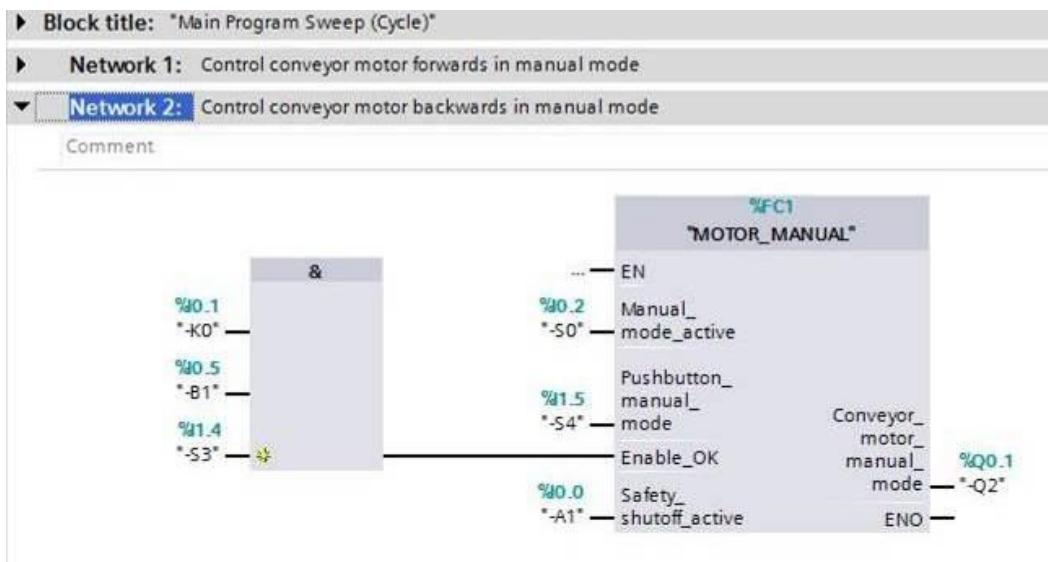


Figure 6: Library-compatible function with call in OB1

4.8 Programming languages

The available programming languages for programming functions are Function Block Diagram (FBD), Ladder Logic (LAD), Statement List (STL) and Structured Control Language (SCL). For function blocks, the GRAPH programming language is additionally available for programming graphical step sequences.

The **Function Block Diagram (FBD)** programming language will be presented in the following.

FBD is a graphical programming language. The representation is based on electronic switching systems. The program is mapped in networks. A network contains one or more logic operation paths. Binary and analog signals are linked by boxes. The graphical logic symbols known from Boolean algebra are used to represent the binary logic.

You can use binary functions to query binary operands and to logically combine their signal states. The following instructions are examples of binary functions: "AND operation", "OR operation" and "EXCLUSIVE OR operation". These are shown in Figure 7.

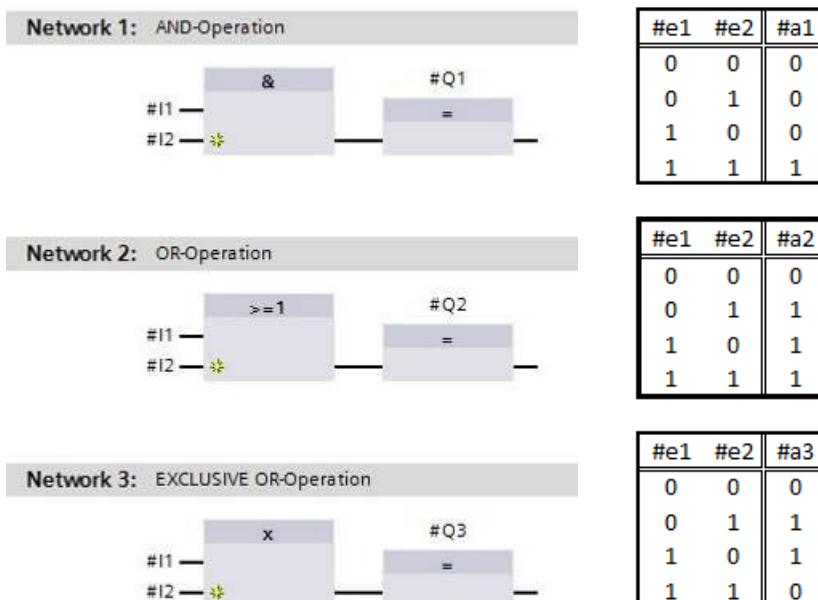


Figure 7: Binary functions in FBD and associated logic table

You can thus use simple instructions, for example, to control binary outputs, evaluate edges and execute jump functions in the program.

Program elements such as IEC timers and IEC counters provide complex instructions.

The empty box serves as a placeholder in which you can select the required instruction.

Enable input EN (enable)/ Enable output ENO (enable output) mechanism:

- An instruction without EN/ENO mechanism is executed independent of the signal state at the box inputs.
- Instructions with EN/ENO mechanism are only executed if enable input "EN input has signal state "1". When the box is processed correctly, enable output "ENO" has signal state "1". As soon as an error occurs during the processing, the "ENO" enable output is reset. If enable input EN is not connected, the box is always executed.

5 Task

The following functions of the sorting station process description will be planned, programmed and tested in this chapter:

- Manual mode – Conveyor motor in manual mode

6 Planning

The programming of all functions in OB1 is not recommended for reasons of clarity and reusability. The majority of the program code will therefore be moved into functions (FCs) and function blocks (FBs). The decision on which functions are to be moved to FCs and which is to run in OB 1 is planned below.

6.1 EMERGENCY STOP

The EMERGENCY STOP does not require a separate function. Just like the operating mode, the current state of the EMERGENCY STOP relay can be used directly at the blocks.

6.2 Manual mode – Conveyor motor in manual mode

Manual mode of the conveyor motor is to be encapsulated in a function (FC) "MOTOR_MANUAL". On the one hand, this preserves the clarity of OB1. On the other hand, it enables reuse if another conveyor belt is added to the station. Table 2 lists the planned parameters.

Input	Data	Comment
Manual_mode_active	BOOL	Manual mode activated
Pushbutton_manual_mode	BOOL	Pushbutton manual mode conveyor on
Enable_OK	BOOL	All enable conditions OK
Safety_shutoff_active	BOOL	Safety shutoff active, e.g. emergency stop pressed
Output		
Conveyor_motor_manual_mode	BOOL	Control of the conveyor motor in manual mode

Table 2: Parameters for FC "MOTOR_MANUAL"

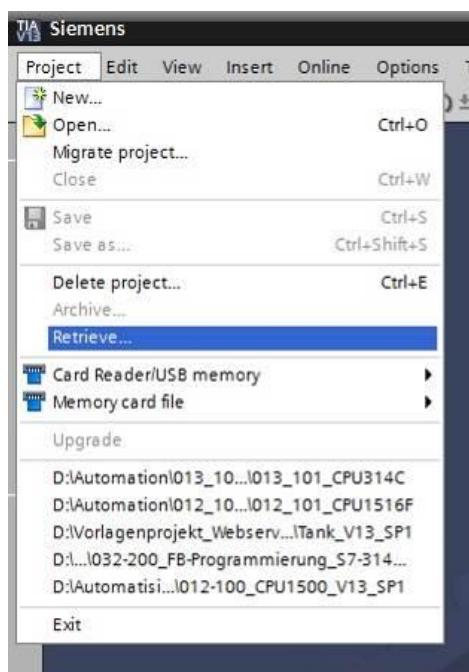
Output Conveyor_motor_manual_mode is ON as long as Pushbutton_manual_mode is pressed, manual mode is activated, the enable conditions are OK and the safety shutoff is not active.

7 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

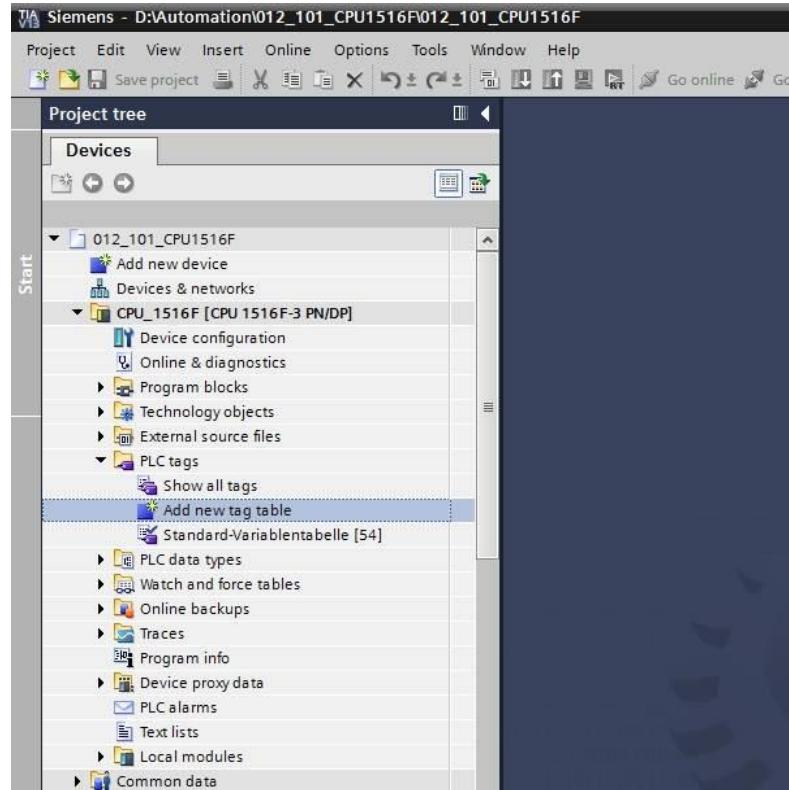
- Before we can start programming the function (FC) "MOTOR_MANUAL", we need a project with a hardware configuration (e.g. SCE_EN_012_101_Hardware_Configuration_S7-1516F_R1502.zap). To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view Confirm your selection with Open. (→ Project → Retrieve → Select a .zap archive → Open)



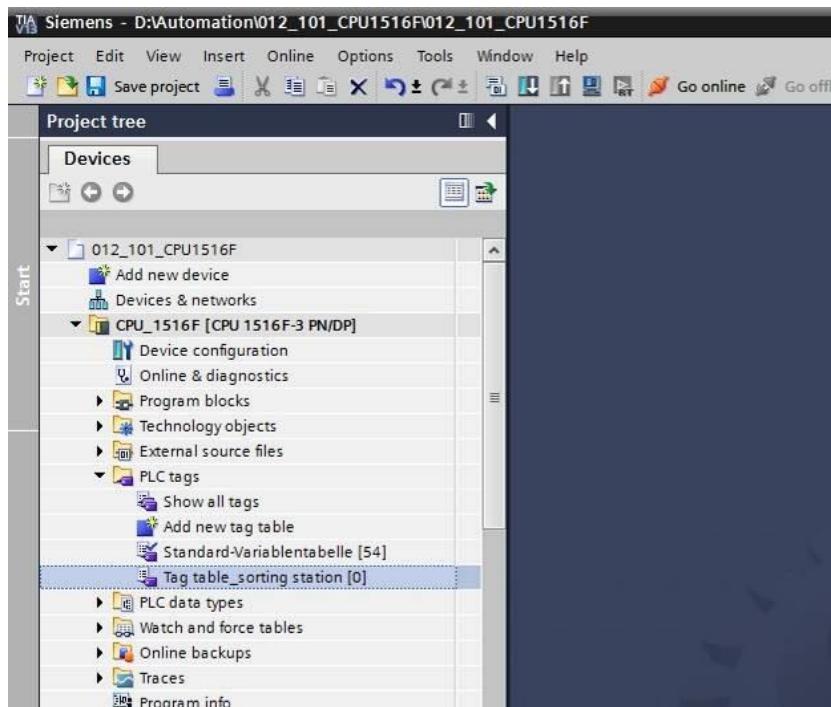
- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK". (→ Target directory → OK)

7.2 Create a new tag table

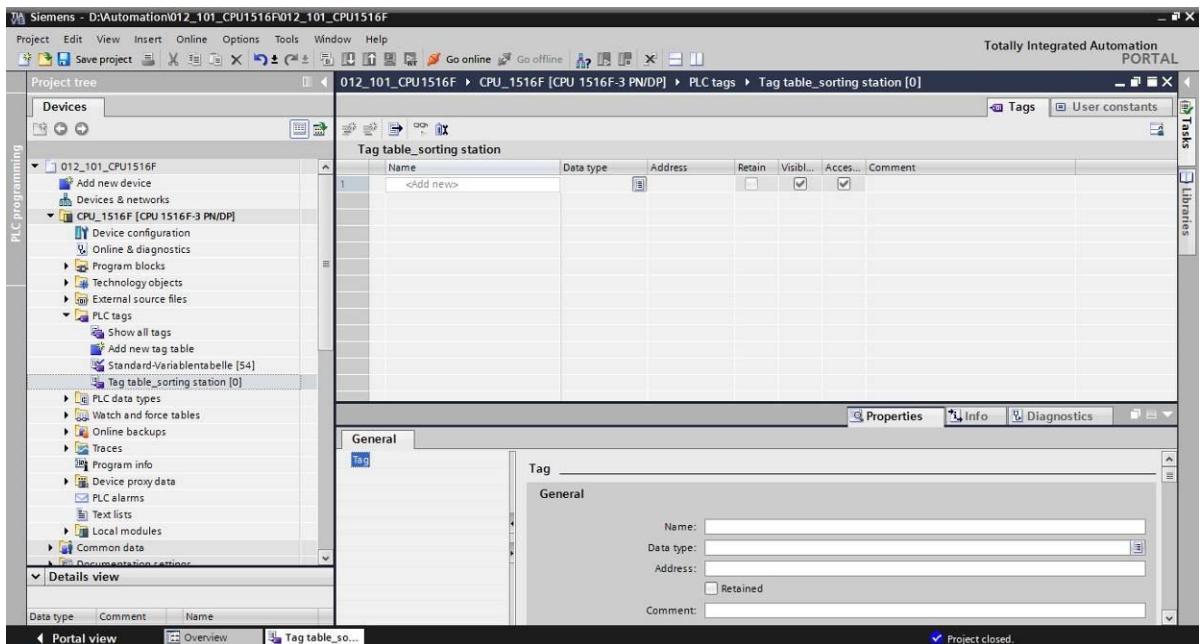
- In the project view, navigate to the → PLC tags of your controller and create a new tag table by double-clicking → Add new tag table.



- Rename the tag table you just created as "Tag_table_sorting_station" (→ right-click "Tag_table_1" → "Rename" → Tag_table_sorting_station).



- Open this tag table with a double-click. (→ Tag_table_sorting_station)



7.3 Create new tags within a tag table

- Add the name Q1 and confirm the entry with the Enter key. If you have not yet created additional tags, TIA Portal now automatically assigns data type "Bool" and address %I0.0 (I 0.0) (→ <Add> → Q1 → Enter).

Name	Data type	Address	Retain	Visible...	Access...	Comment
Q1	Bool	%I0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<Add new>						

- Change the address to %Q0.0 (Q 0.0) by entering this directly or by clicking the drop-down arrow to open the Addressing menu, changing the operand identifier to Q and confirming with Enter or by clicking the check mark. (→ %I0.0 → Operand identifier → Q →

Name	Data type	Address	Retain	Visible...	Access...	Comment
Q1	Bool	%I0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<Add new>						

Operand identifier: I
Operand type: I
Address: Q
Bit number: M
0

- Enter the "Conveyor motor M1 forwards fixed speed" comment for the tag.

	Name	Data type	Address	Retain	Visibl...	Acces...	Comment
1	Q1	Bool	%Q0.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
2	<Add new>						

- Add a new Q2 tag in line 2. TIA Portal has automatically assigned the same data type as in line 1 and has incremented the address by 1 to %Q0.1 (Q0.1). Enter the comment "Conveyor motor M1 backwards fixed speed".

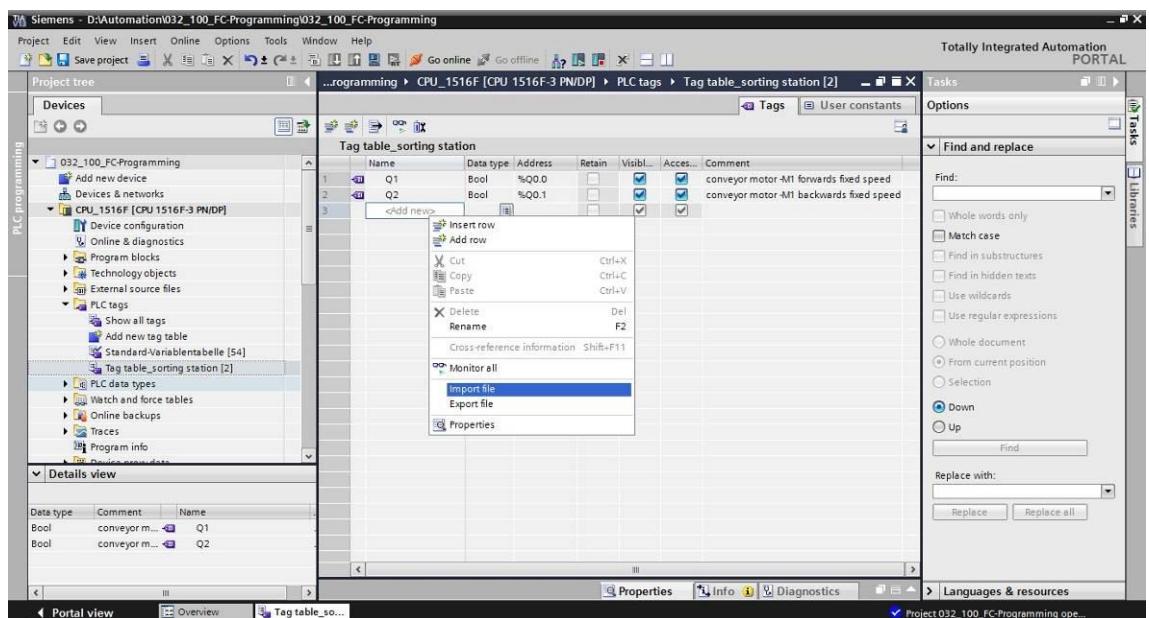
(→ <Add> → Q2 → Enter → Comment → Conveyor motor M1 backwards fixed speed)

	Name	Data type	Address	Retain	Visibl...	Acces...	Comment
1	Q1	Bool	%Q0.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
2	Q2	Bool	%Q0.1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
3	<Add new>						

7.4 Import "Tag_table_sorting_station"

- To insert an existing symbol table, right-click on an empty field of the created "Tag_table_sorting_station". Select "Import file" in the shortcut menu.

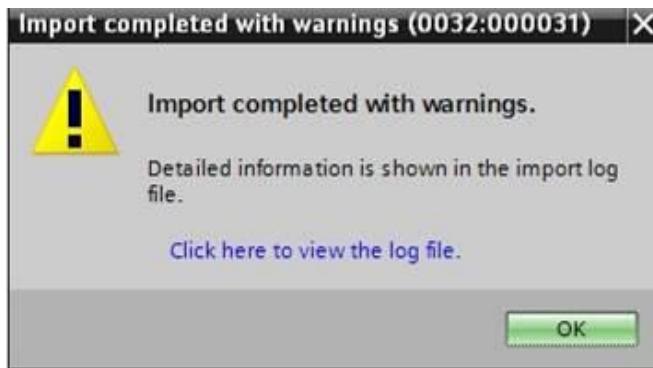
(→ Right-click in an empty field of the tag table → Import file)



- Select the desired symbol table (e.g. in .xlsx format) and confirm the selection with "Open".

(→ SCE_EN_020-100_Tag_table_sorting_station... → Open)

- When the import is finished, you will see a confirmation window and have an opportunity to view the log file for the import. Click → OK.



- You can see that some addresses have been highlighted in orange. These are duplicate addresses and the names of the associated tags have been numbered automatically to avoid confusion.
- Delete the duplicate tags by selecting the lines and pressing the Del key on your keyboard or by selecting "Delete" in the shortcut menu.

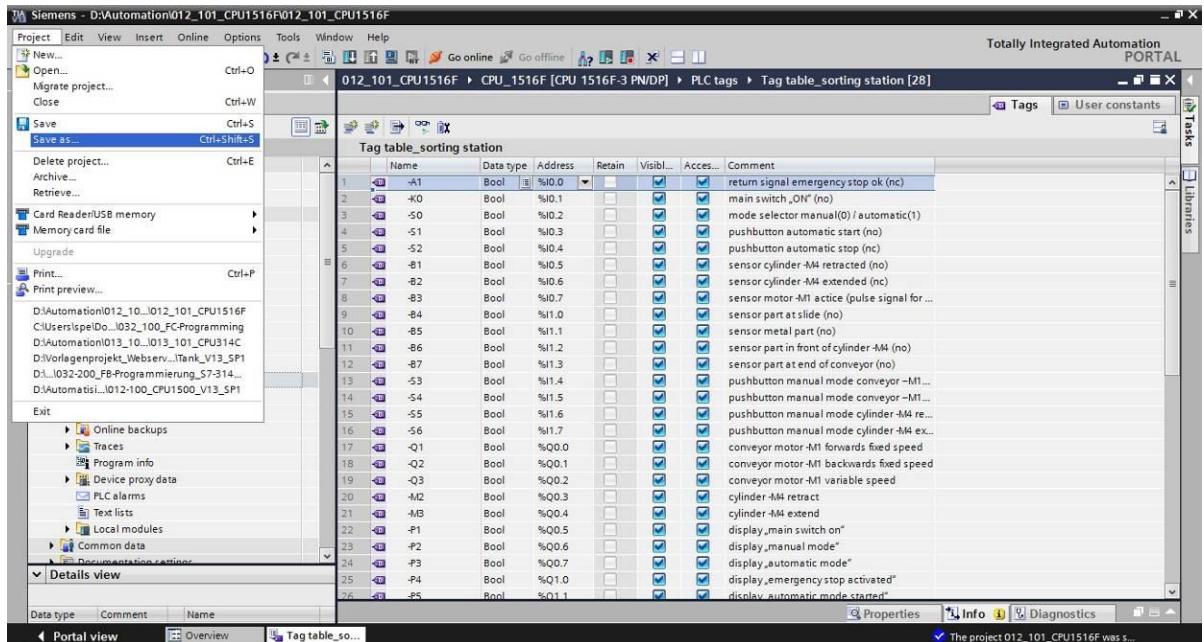
(→ Right-click on selected tags → Delete)

The screenshot shows the TIA Portal interface with the project tree on the left and the tag table on the right. The tag table is titled "Tag table_sorting station" and contains 30 entries. The first two entries, Q1 and Q2, are highlighted in orange, indicating they are duplicates. The table includes columns for Name, Data type, Address, Retain, Visibility, Access, and Comment. A context menu is open over the second row (Q2), with "Delete" selected. Other options in the menu include Insert row, Cut, Copy, Paste, Rename, Cross-reference information, Monitor all, Import file, Export file, and Properties.

Name	Data type	Address	Retain	Visibility	Access	Comment
Q1	Bool	%Q0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor-M1 forwards fixed speed
Q2	Bool	%Q0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor-M1 backwards fixed speed
						return signal emergency stop ok (no)
						main switch „ON“ (no)
						mode selector manual(0) / automatic(1)
						pushbutton automatic start (no)
						pushbutton automatic stop (no)
						sensor cylinder-M4 retracted (no)
						sensor cylinder-M4 extended (no)
						sensor motor-M1 active (pulse signal for positioning) (no)
						sensor part at slide (no)
						sensor metal part (no)
						sensor part in front of cylinder-M4 (no)
						sensor part at end of conveyor (no)
						pushbutton manual mode conveyor-M1 forwards (no)
						pushbutton manual mode conveyor-M1 backwards (no)
						pushbutton manual mode cylinder-M4 retract (no)
						pushbutton manual mode cylinder-M4 extend (no)
						conveyor motor-M1 forwards fixed speed
						conveyor motor-M1 backwards fixed speed
						conveyor motor-M1 variable speed
						cylinder-M4 retract
						cylinder-M4 extend
						display „main switch on“
						display „manual mode“
						display „automatic mode“

- You now have a complete symbol table of the digital inputs and outputs in front of you. Save your project under the name 032-100_FCProgramming.

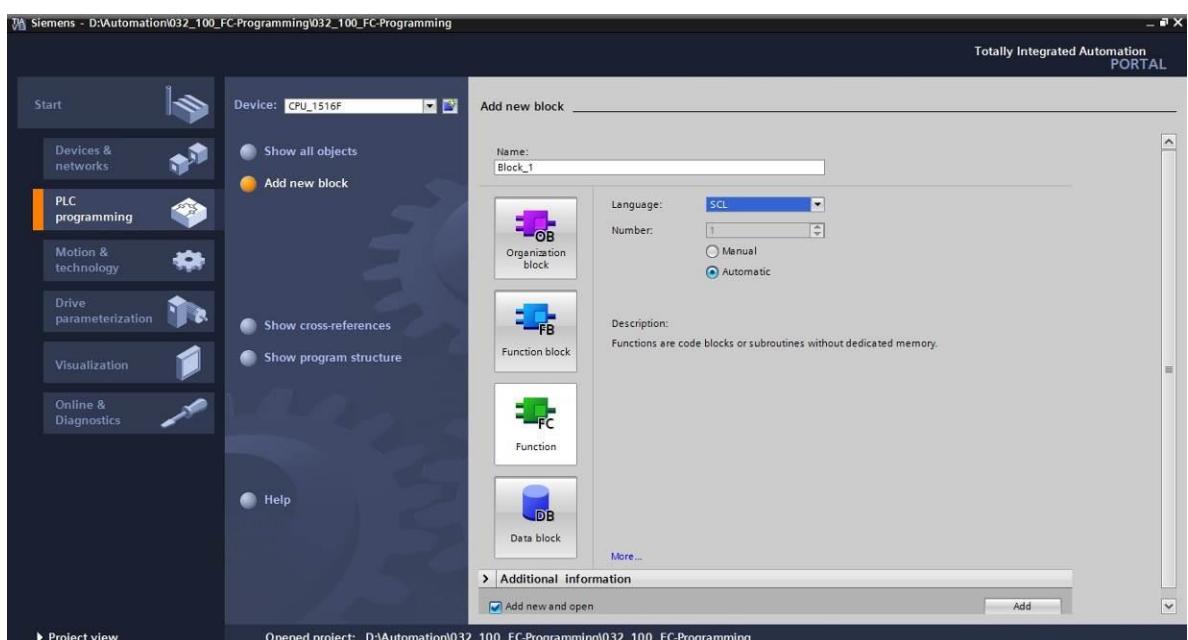
(→ Project → Save as ... → 032-100_FCProgramming → Save)



7.5 Create function FC1 "MOTOR_MANUAL" for the conveyor motor in manual mode

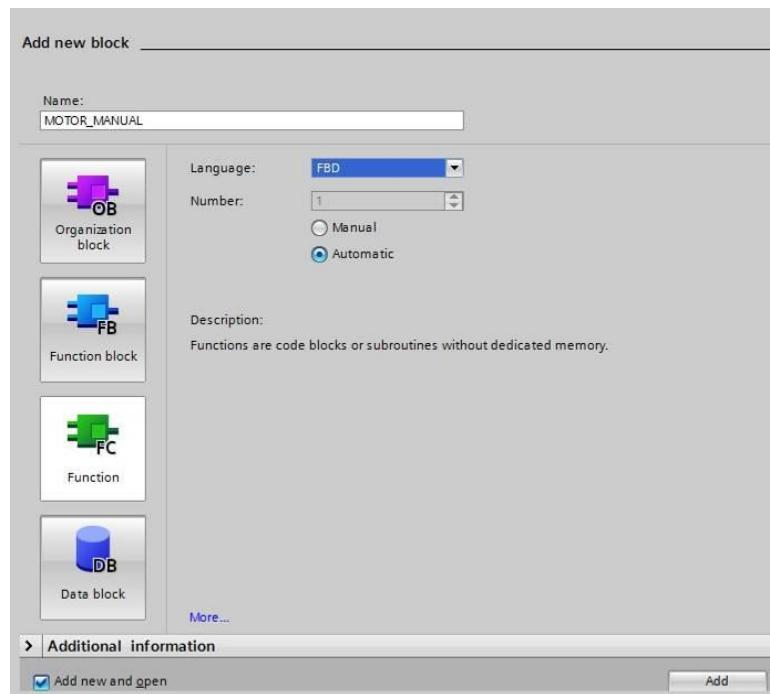
- In the PLC programming section of the portal view, click "Add new block" to create a new function.

(→ PLC programming → Add new block → FC)



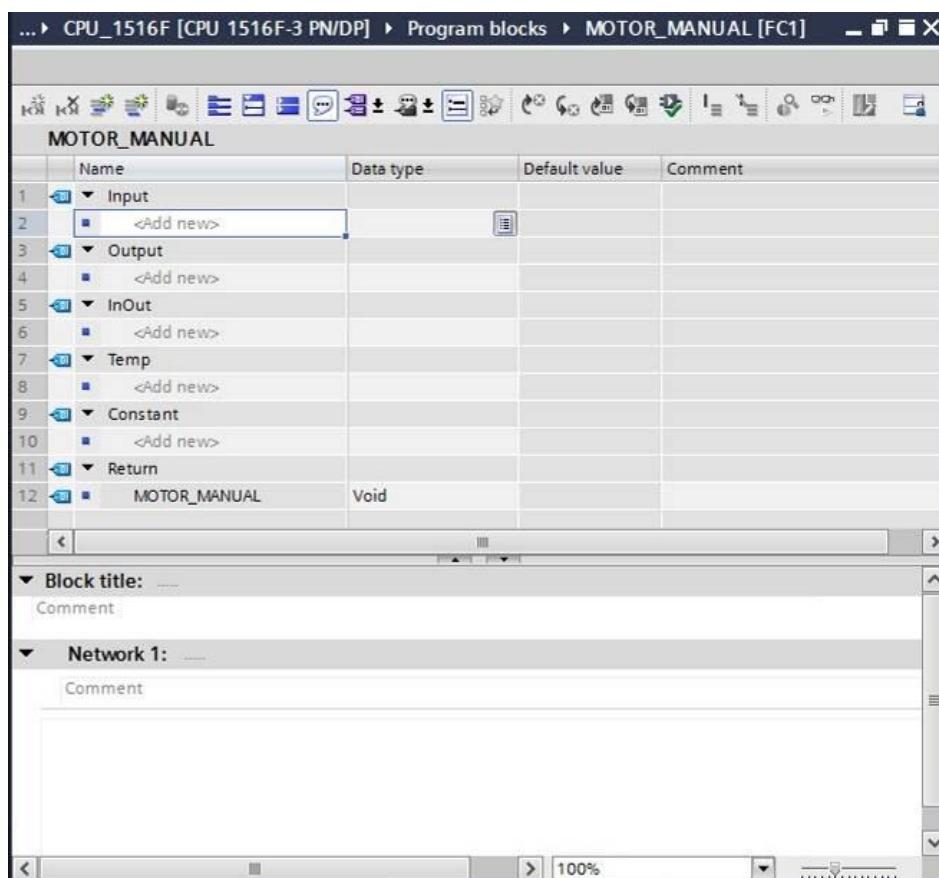
- Rename your new block to: "MOTOR_MANUAL", set the language to FBD and keep automatic assignment of the number. Select the "Add new and open" check box. You are then taken automatically to your created function block in the project view. Click "Add".

(→ Name: MOTOR_MANUAL → Language: FBD → Number: Automatic → Add new and open → Add)



7.6 Define the interface of function FC1 "MOTOR_MANUAL"

- If you selected "Add new and open", the project view opens with a window for creating the block you just added.
- You can find the interface description of your function in the upper section of your programming view.



→ A binary output signal is needed for controlling the conveyor motor. For this reason, we first create local output tag #Conveyor_motor_manual_mode of the "Bool" type. Enter comment "Control of the conveyor motor in manual mode" for the parameter.

(→ Output: Conveyor_motor_manual_mode → Bool → Control of the conveyor motor in manual mode)

	Name	Data type	Default value	Comment
1	Input			
2	<Add new>			
3	Output			
4	Conveyor_motor_manual... Bool			Control of the conveyor motor in manual mode
5	<Add new>			
6	InOut			
7	<Add new>			
8	Temp			
9	<Add new>			
10	Constant			
11	<Add new>			
12	Return			
13	MOTOR MANUAL	Void		

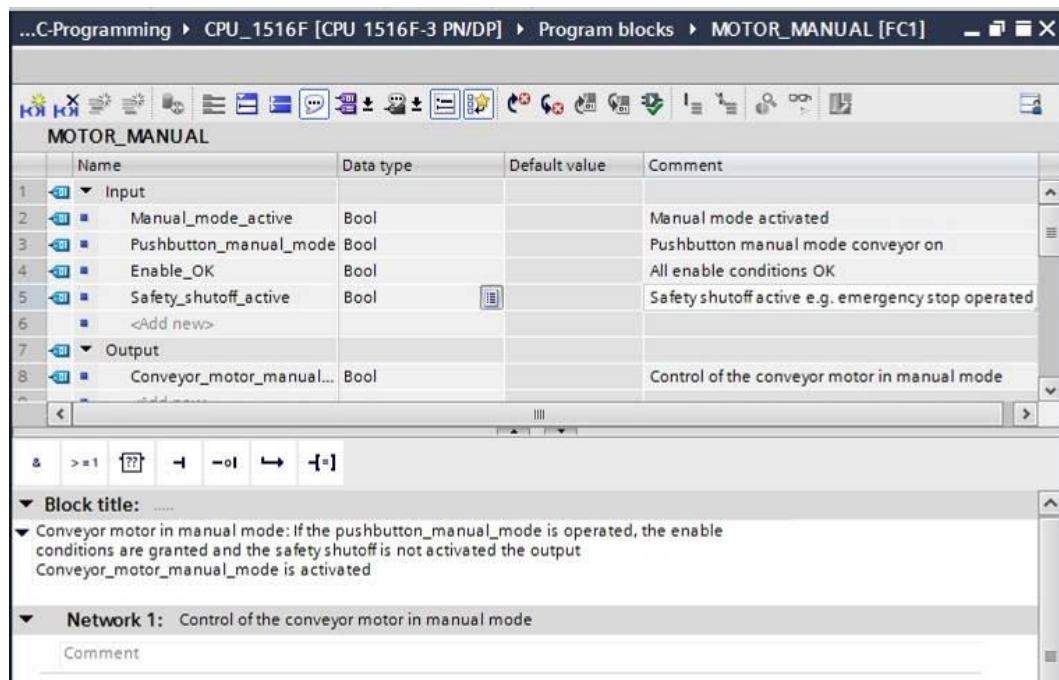
→ Add parameter #Manual_mode_active as the input interface under Input and confirm the entry with the Enter key or by exiting the entry field. Data type "Bool" is assigned automatically. This will be retained. Next, enter the associated comment "Manual mode activated".

(→ Manual_mode_active → Enter → Bool → Manual mode activated)

→ Add parameters #Pushbutton_manual_mode, #Enable_OK and #Safety_shutoff_active as additional binary input parameters under Input and check their data types. Add descriptive comments.

	Name	Data type	Default value	Comment
1	Input			
2	Manual_mode_active	Bool		Manual mode activated
3	Pushbutton_manual_mode	Bool		Pushbutton manual mode conveyor on
4	Enable_OK	Bool		All enable conditions OK
5	Safety_shutoff_active	Bool		Safety shutoff active e.g. emergency stop operated
6	<Add new>			
7	Output			
8	Conveyor_motor_manual... Bool			Control of the conveyor motor in manual mode
9	<Add new>			
10	InOut			
11	<Add new>			
12	Temp			
13	<Add new>			
14	Constant			
15	<Add new>			
16	Return			
17	MOTOR MANUAL	Void		

- For purposes of program documentation, assign the block title, a block comment and a helpful network title for Network 1.
- (→ Block title: Motor control in manual mode → Network 1: Control of the conveyor motor in manual mode)



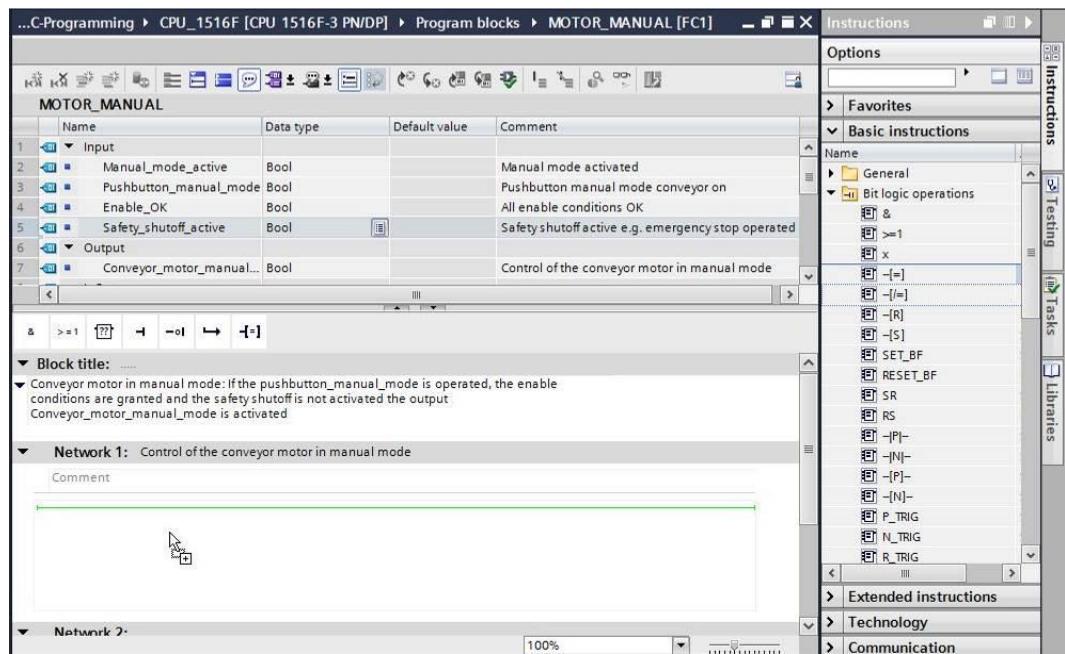
7.7 Program FC1: MOTOR_MANUAL

- Below the interface description, you see a toolbar in the programming window with various logic functions and below that an area with networks. We have already specified the block title and the title for the first network there. Programming is performed within the networks using individual logic blocks. Distribution among multiple networks helps to preserve the clarity of the program. In the following, you will get to know the various ways you can insert logic blocks.



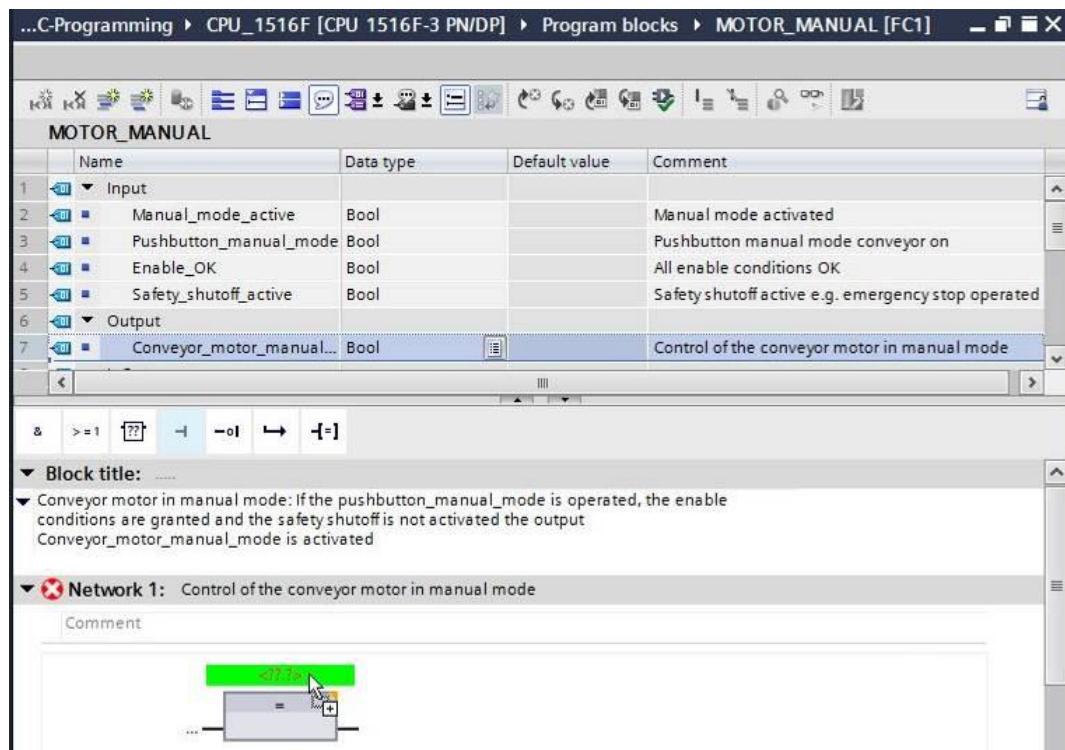
- You can see a list of instructions you can use in the program on the right side of your programming window. Under → Basic instructions → Bit logic operations, find function $=[]$ (Assignment) and use a drag-and-drop operation to move it to Network 1 (green line appears, mouse pointer with + symbol).

(→ Instructions → Basic instructions → Bit logic operations → $-[=]$)



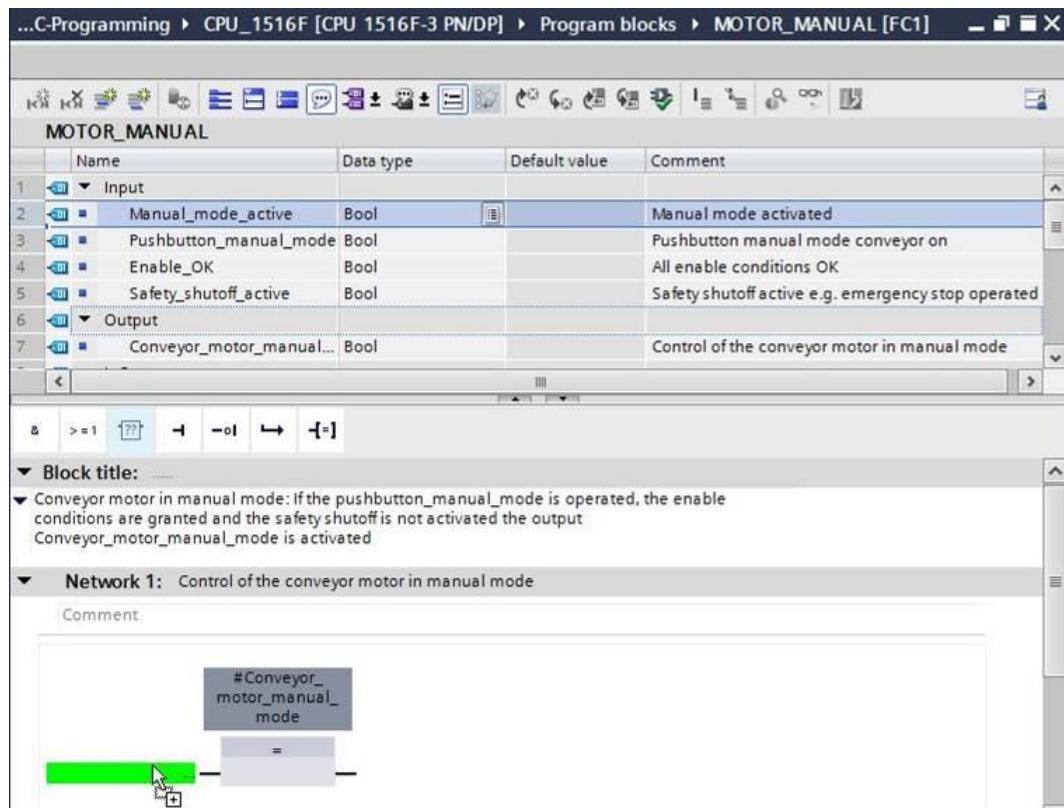
- Now use drag-and-drop to move your output parameter #Conveyor_motor_manual_mode onto <???.?> above the block you just inserted. The best way to select a parameter in the interface description is by "grabbing" it at the blue symbol

(→ Conveyor_motor_manual_mode)



- This determines that the #Conveyor_motor_manual_mode parameter is written by this block. Still missing, however, are the input conditions so that this actually happens. For this, use drag-and-drop to move input parameter #Manual_mode_active to the left side of the assignment block.

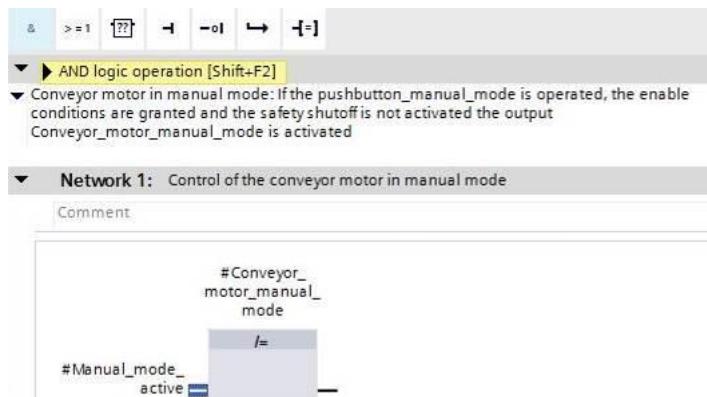
(→ Manual_mode_active)



- The input of the assignment block will also be logically combined with other parameters by an AND logic operation. To do this, first click the input of the block to which #Manual_mode_active is already connected, so that the input line has a blue background.

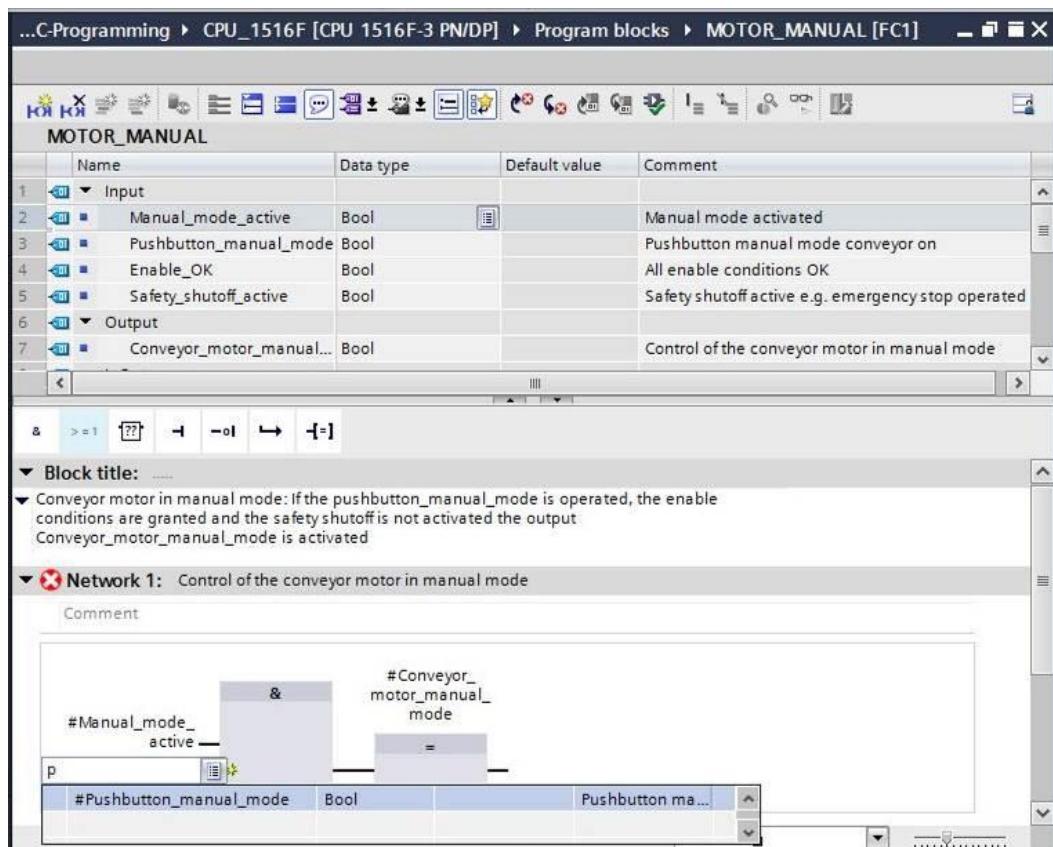


- Click the  icon in your logic toolbar to insert an AND logic operation between the #Manual_mode_active tag and your assignment block.



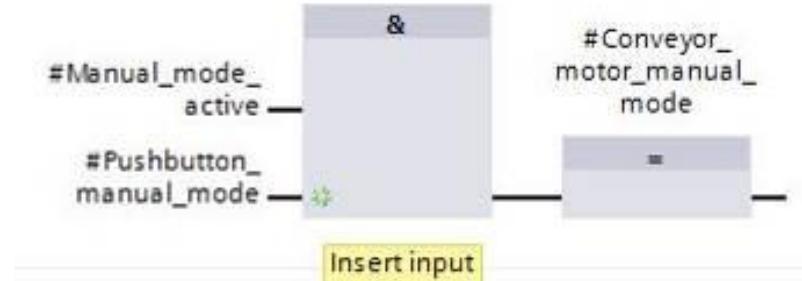
- Double-click the second input of the & logic operation  and enter the letter "P" in the field that appears in order to see a list of available tags starting with "P". Click the #Pushbutton_manual_mode tag and apply with → Enter.

(→ &- block →  → P → #Pushbutton_manual_mode → Enter)

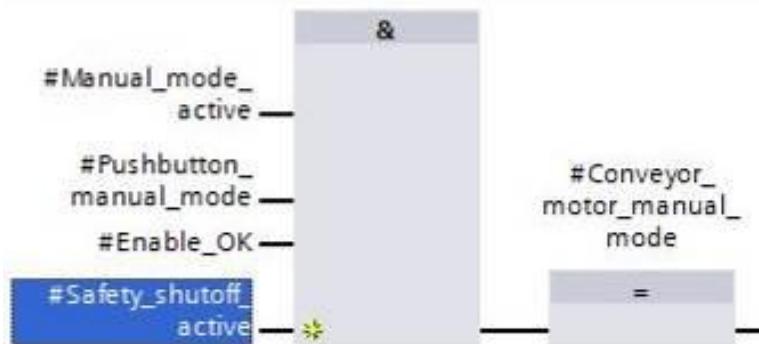


Note: When assigning tags in this way, there is a risk of a mix-up with the global tags from the tag table. The previously presented procedure using drag and drop from the interface description should therefore be used preferentially.

- To ensure that the output can only be controlled when the enable conditions are met and the safety shutoff is not active, the #Enable_OK and #Safety_shutoff_active input tags are logically combined with the AND logic operation. To do this, click twice on the yellow star  of your AND block to add two additional inputs.



- Add input tags #Enable_OK and #Safety_shutoff_active to your newly created inputs of the AND block.



- Negate the input connected to parameter #Safety_shutoff_active by selecting it and clicking .

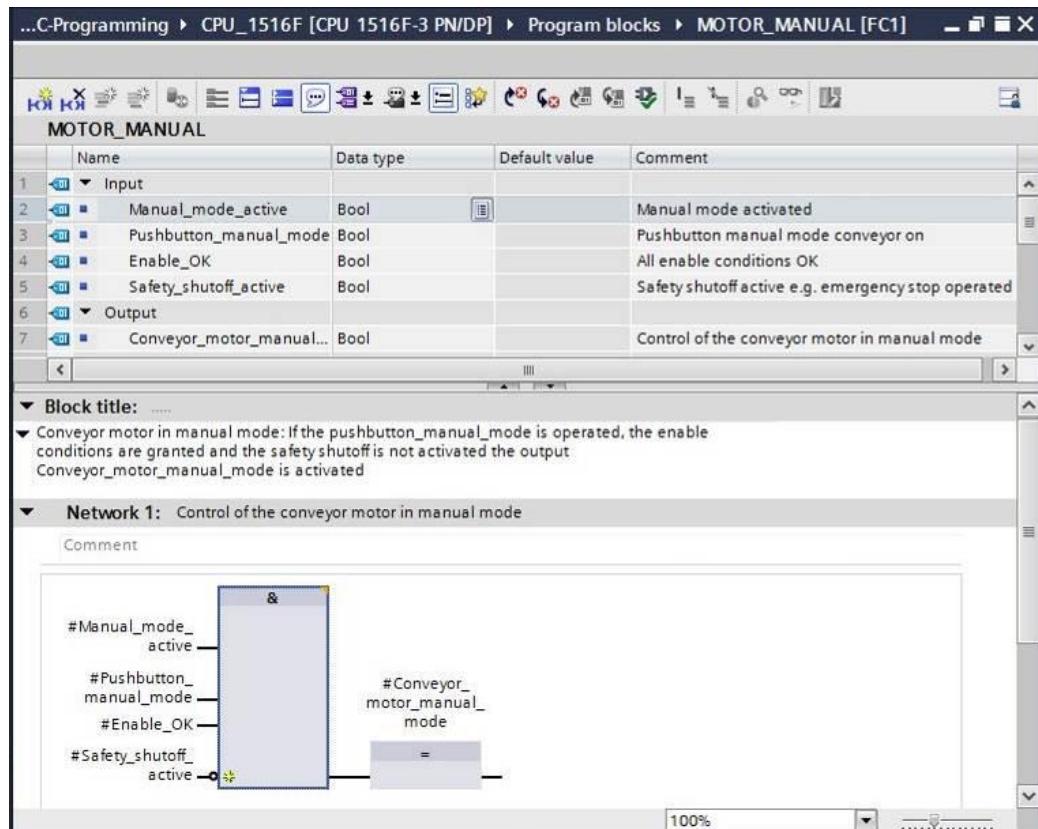
& > = 1 ?? -o! -o -{:}

▼ Block title: ... ▶ Invert RLO [Ctrl+Shift+4]
Conveyor motor in manual mode. When pushbutton_manual_mode is operated, the enable conditions are granted and the safety shutoff is not activated the output Conveyor_motor_manual_mode is activated

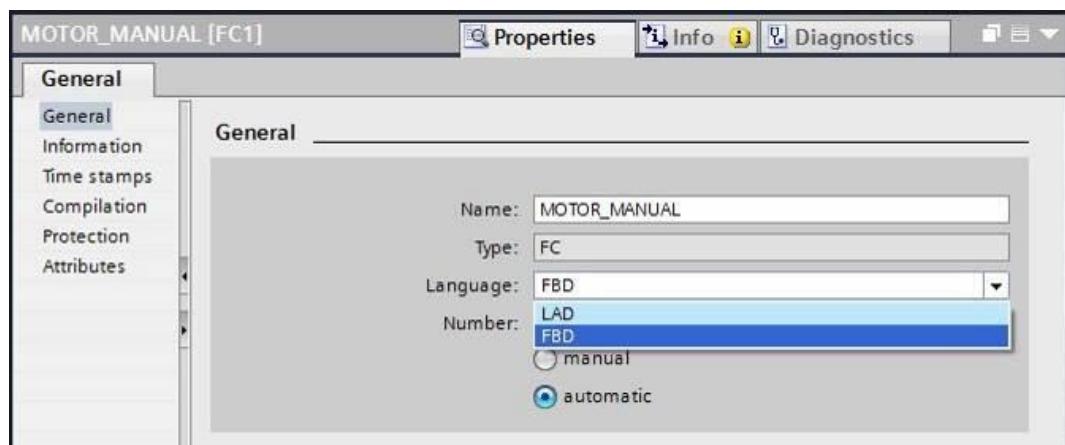
▼ Network 1: Control of the conveyor motor in manual mode

#Manual_mode_active
#Pushbutton_manual_mode
#Enable_OK
#Safety_shutoff_active

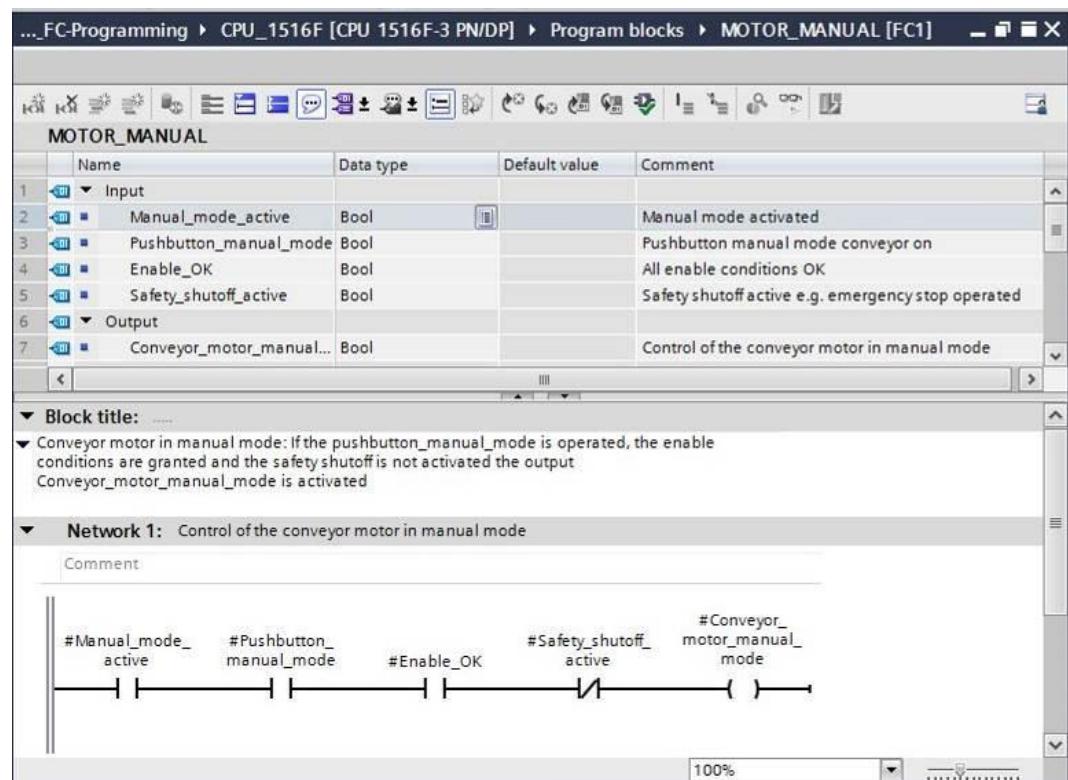
- Do not forget to click Save project. The finished function "MOTOR_MANUAL" [FC1] in FBD is shown below.



- Under "General" in the properties of the block, you can change the "Language" to LAD (Ladder Logic) (→Properties → General → Language: LAD)



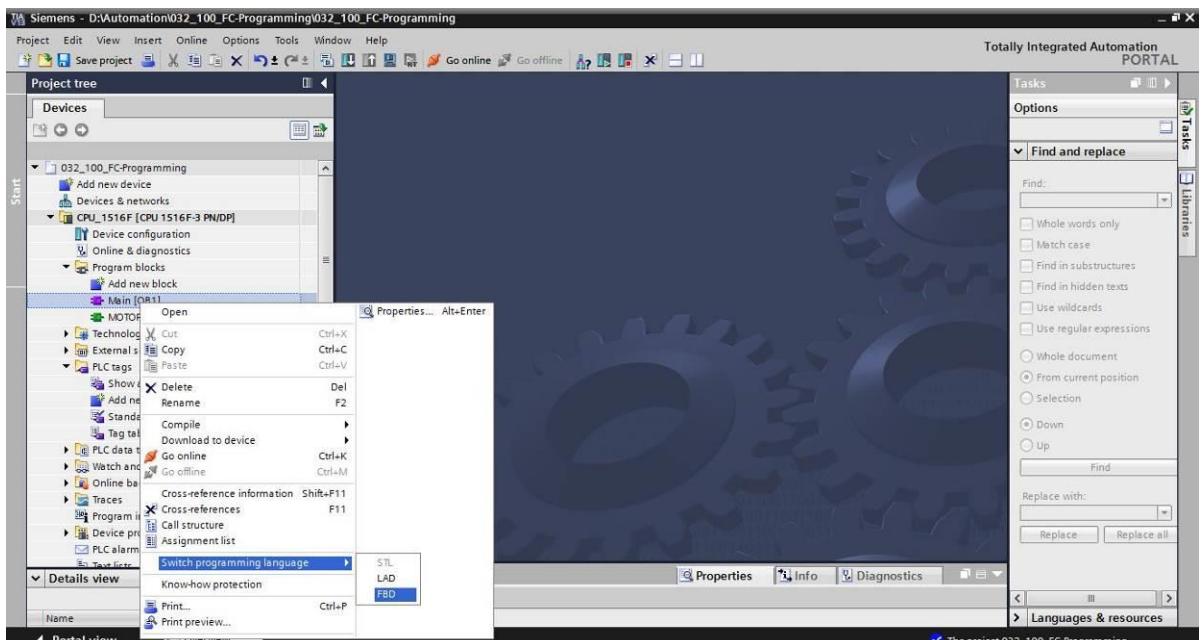
- The program has the following appearance in LAD.



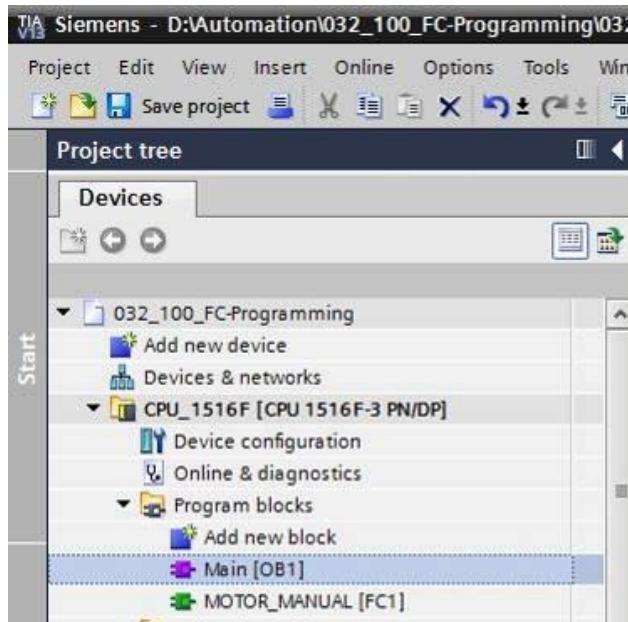
7.8 Program organization block OB1 – Control of the forward belt tracking in manual mode

- Before programming organization block "Main [OB1]", we switch the programming language to FBD (Function Block Diagram). To do so, first click on "Main [OB1]" in the "Program blocks" folder.

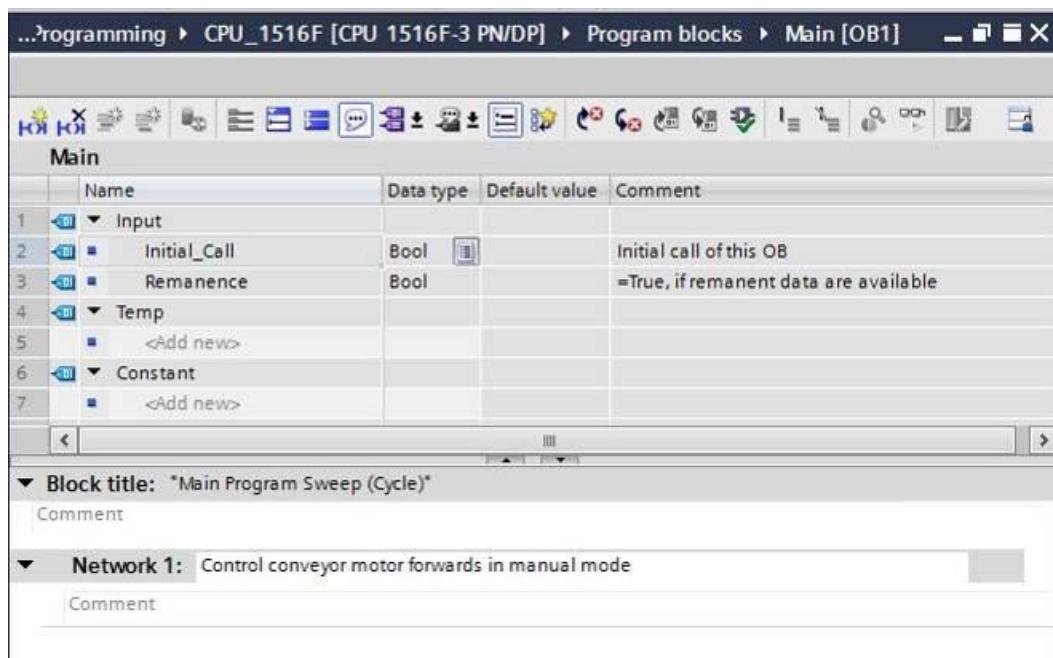
(→ CPU_1516F[CPU 1516F-3 PN/DP → Program blocks → Main [OB1] → Switch programming language → FBD)



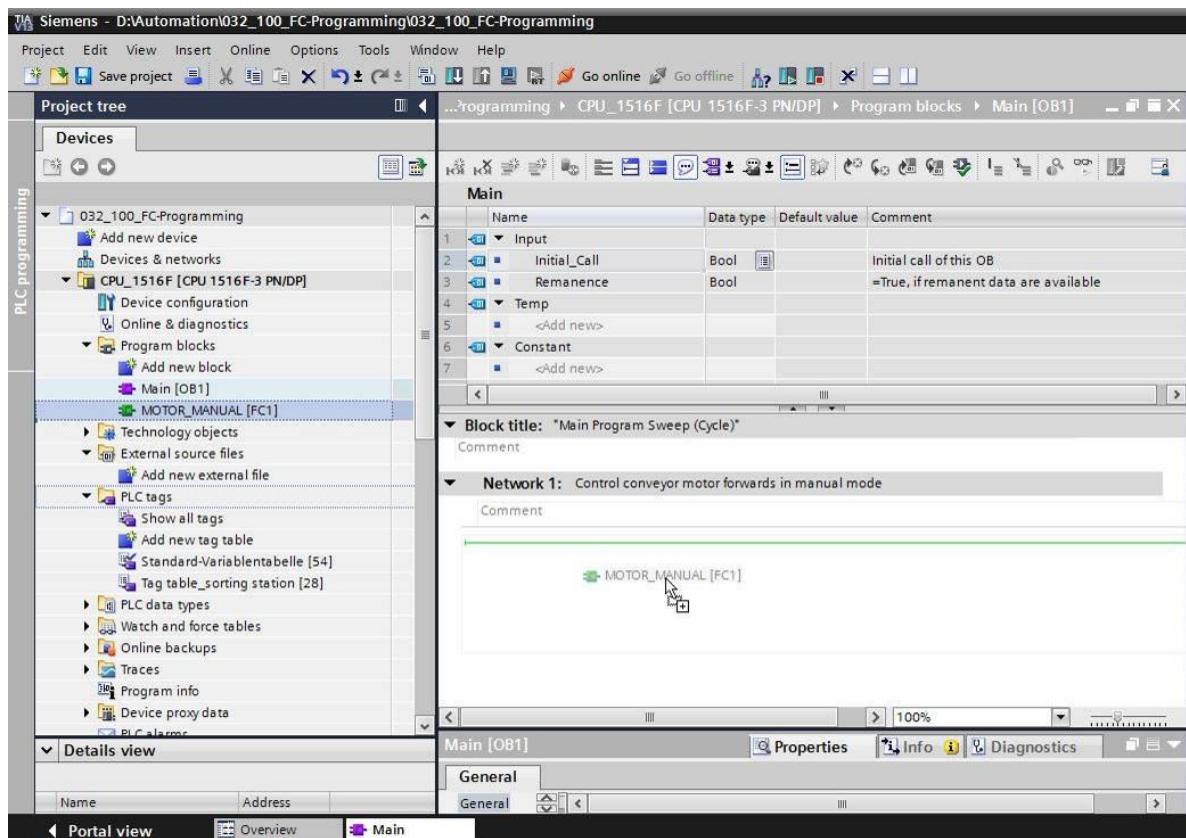
- Open the "Main [OB1]" organization block with a double-click.



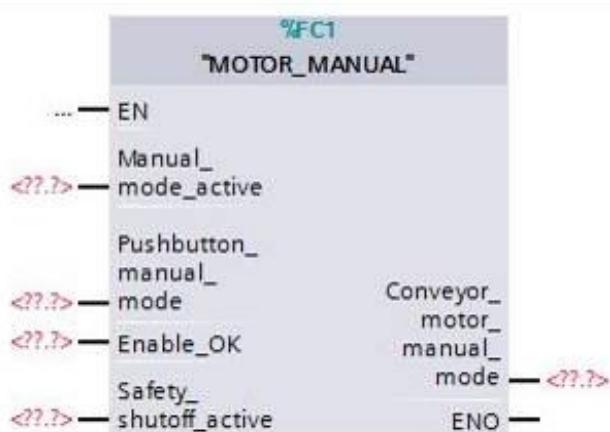
- Assign Network 1 the name "Control conveyor tracking forward in manual/jog mode"
- (→ Network 1:... →Control conveyor motor forwards in manual mode)



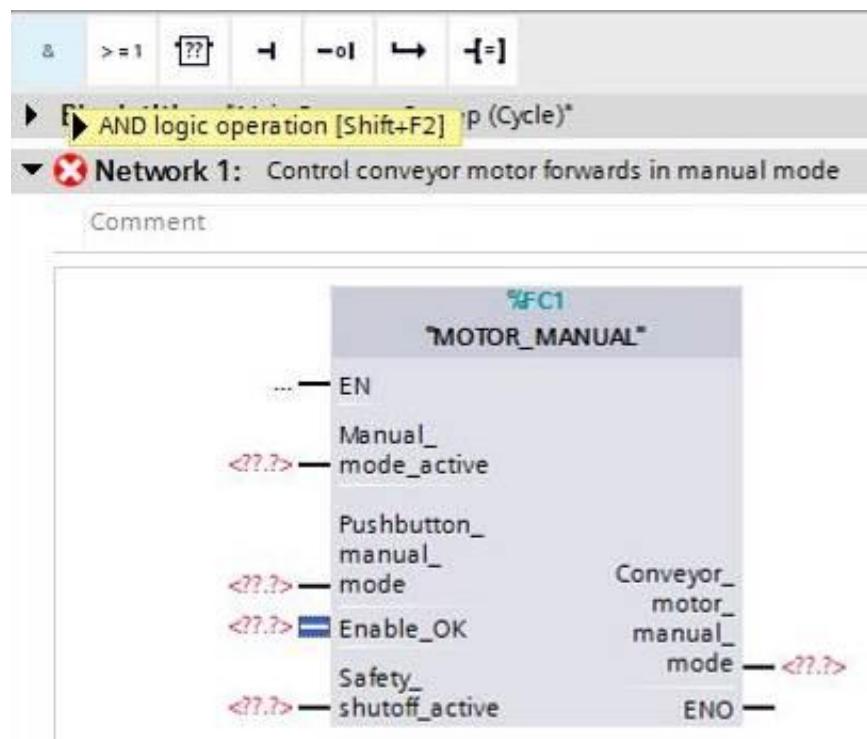
- Use drag-and-drop to move your "MOTOR_MANUAL [FC1]" function onto the green line in Network 1.



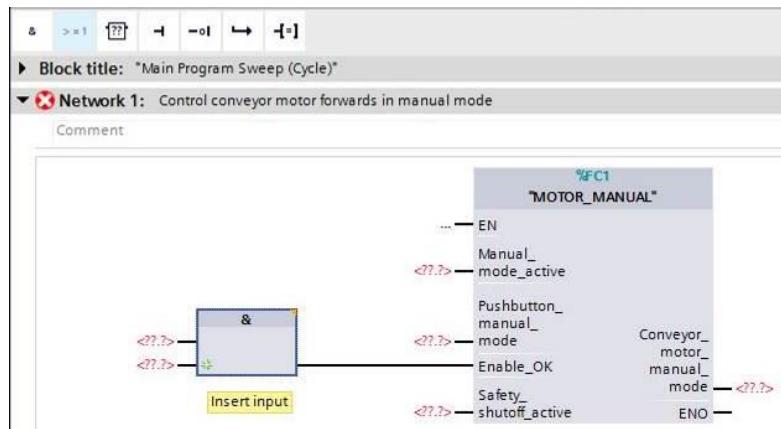
- A block with the interface you defined and connections EN and ENO are inserted in Network 1.



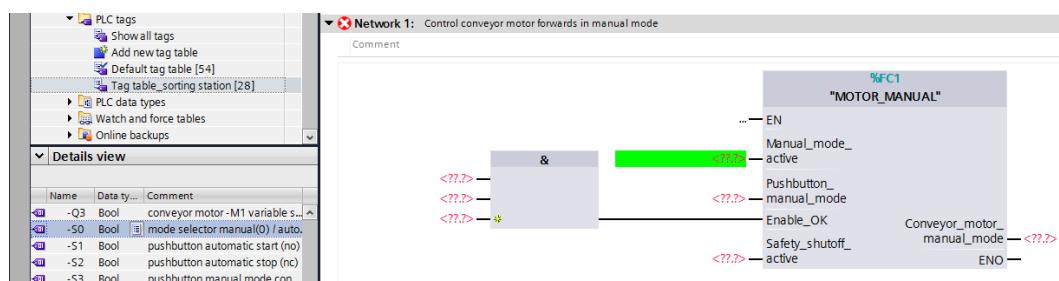
- To insert an AND before input parameter "Enable_OK", select this input and insert the AND by clicking the icon in your logic toolbar ($\rightarrow \wedge$).



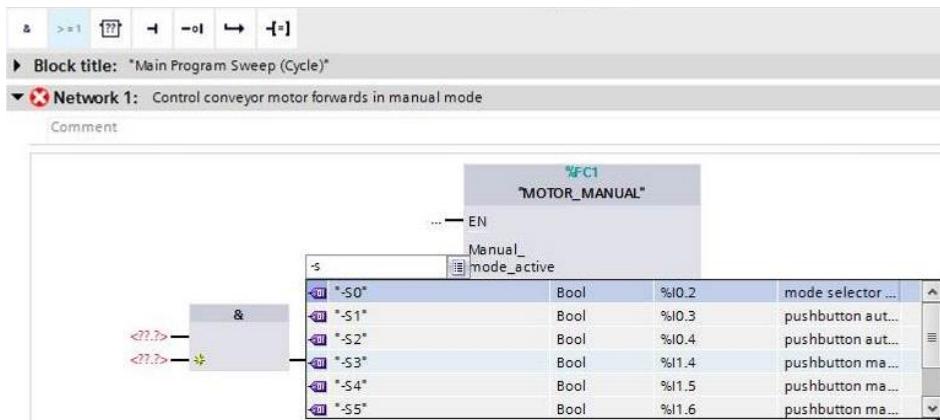
- Click the yellow star of the AND block to add another input (\rightarrow).



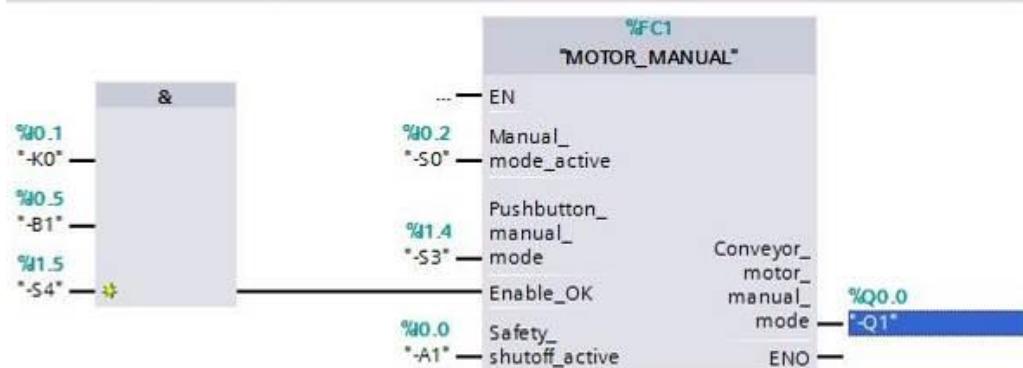
- To connect the block to the global tags from "Tag_table_sorting_station", we have two options:
- Either select the "Tag_table_sorting_station" in the project tree and use drag-and-drop to move the desired global tag from the Details view to the interface of FC1
 $(\rightarrow \text{Tag_table_sorting_station} \rightarrow \text{Details view.} \rightarrow \text{-SO2} \rightarrow \text{Manual_mode_active})$



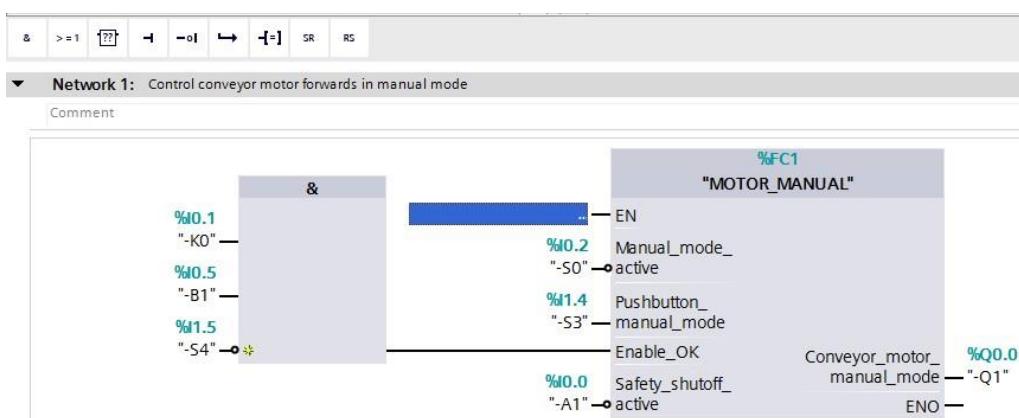
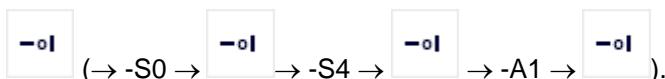
- Or, enter the starting letters (e.g. "-S") of the desired global tag for <???.?> and select the global input tag "-S0" (%I0.2) from the displayed list (→ Manual_mode_active → -S → -S0).



- Insert the other input tags "-S3", "-K0", "-B1", "-S4" and "-A1" and insert output tag "-Q1" (%Q0.0) at output "Conveyor_motor_manual_mode".

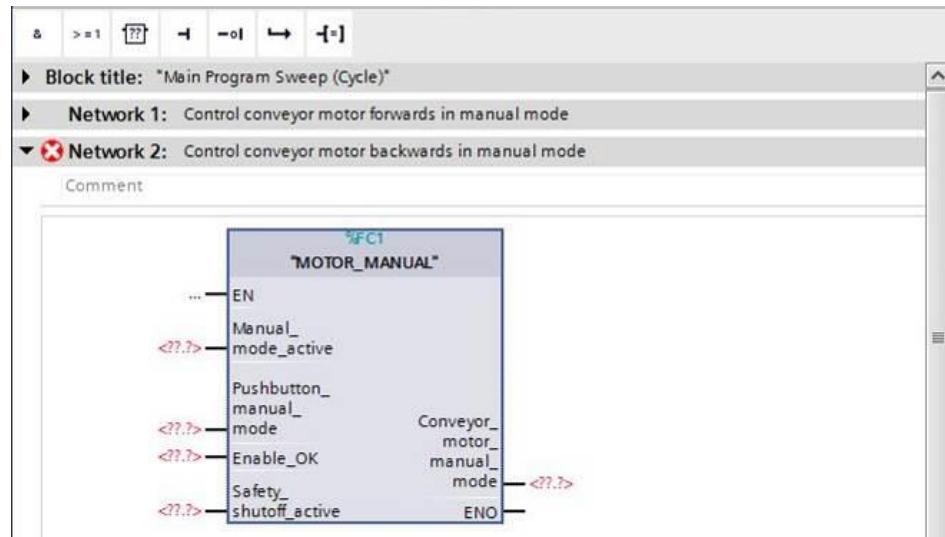


- Negate the querying of input tags "-S0", "-S4" and "-A1" by selecting them and clicking

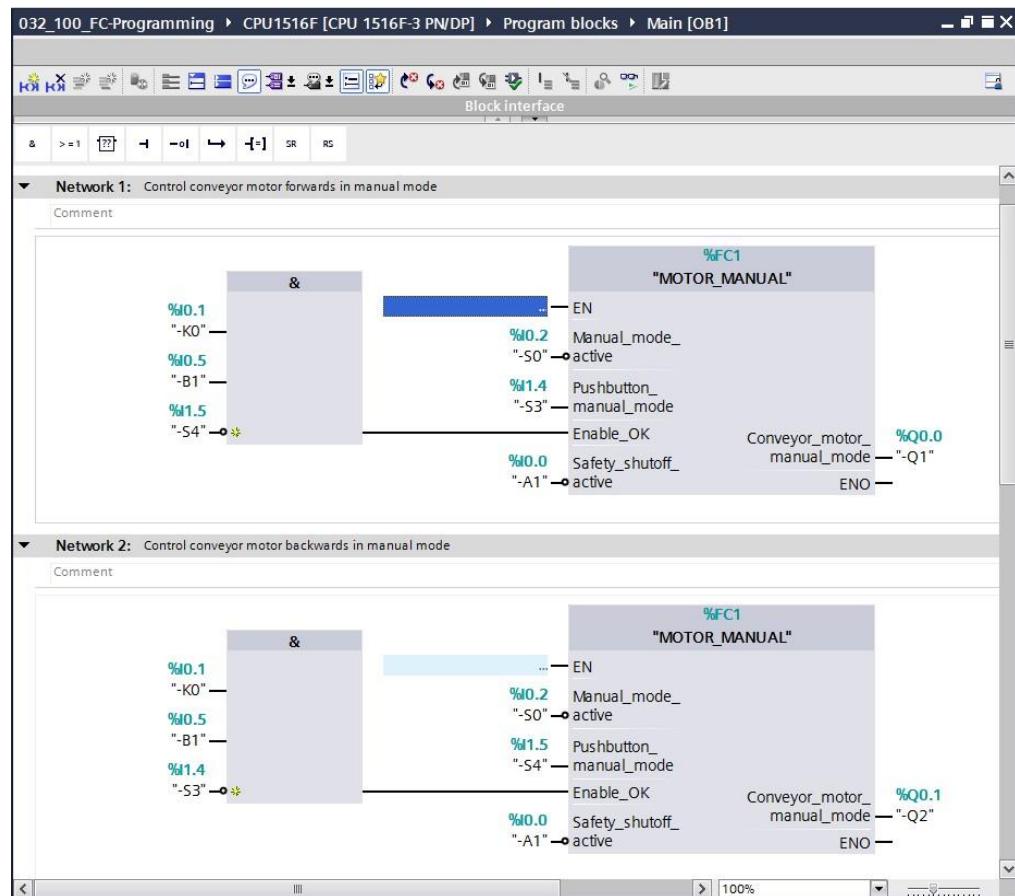


7.9 Program organization block OB1 – Control of the backward belt tracking in manual mode

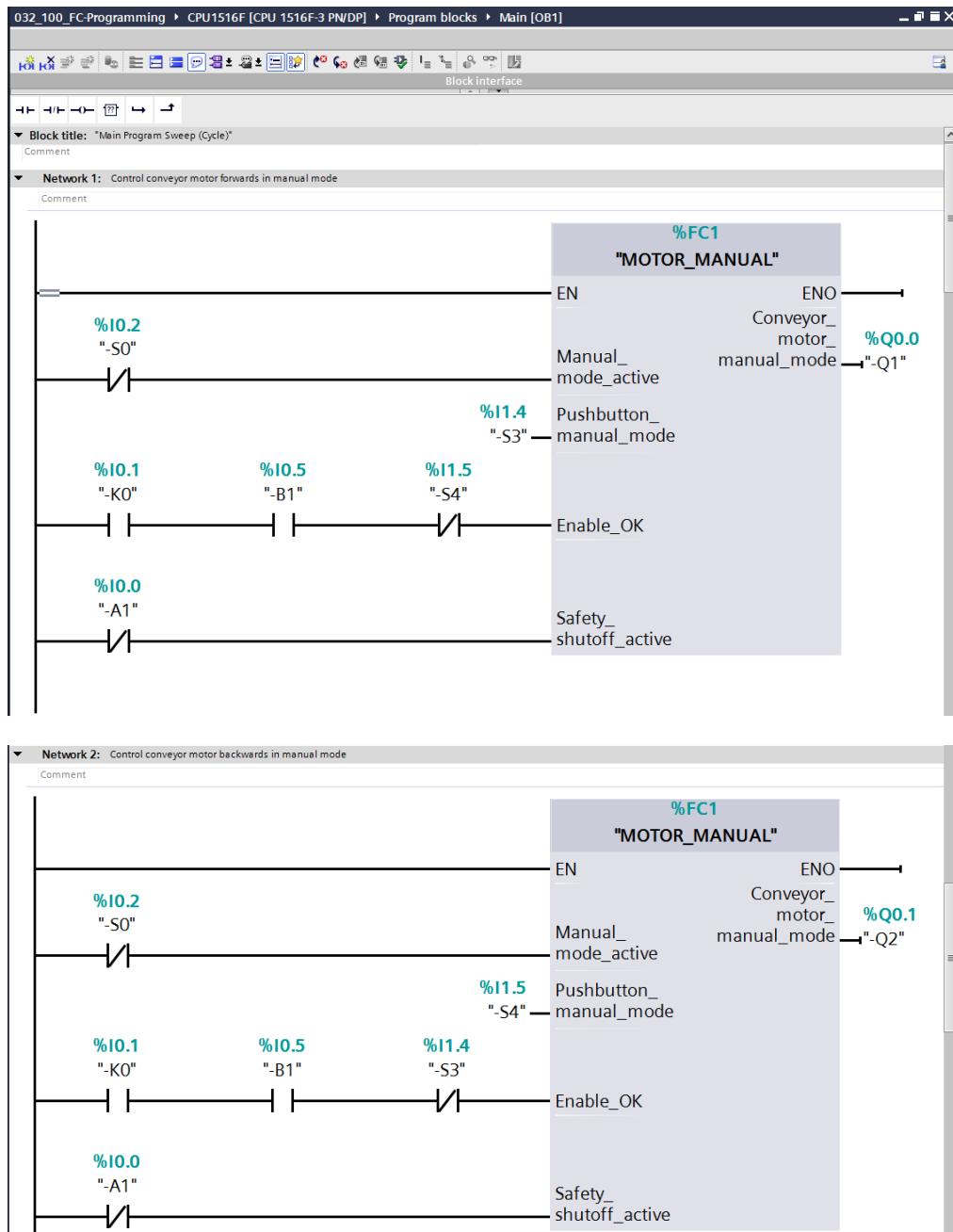
- Assign Network 2 the name "Control conveyor motor backwards in manual mode" and insert your "MOTOR_MANUAL [FC1]" function using drag-and-drop, as you did previously in Network 1.



- Connect your function as shown here. You obtain the following result in the FBD (Function Block Diagram) programming language.

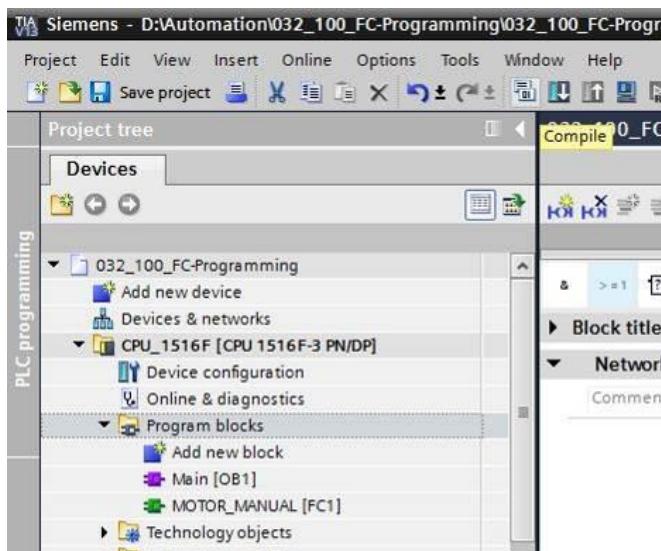


- The result in the LAD (Ladder Logic) programming language has the following appearance.



7.10 Save and compile the program

- To save your project, select the  **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu (→  **Save project** → Program blocks → .

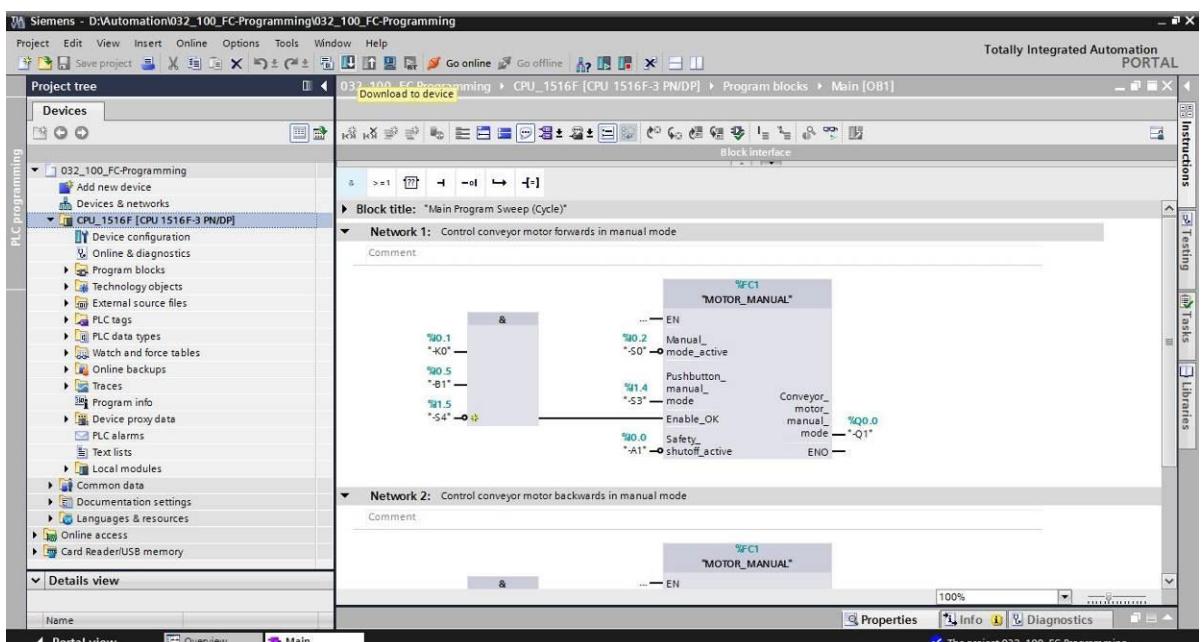


- The "Info", "Compile" area shows which blocks were successfully compiled.

Properties						Info	Diagnostics
General		Cross-references		Compile	Syntax		
			Show all messages				
Compiling completed (errors: 0; warnings: 0)							
!	Path	Description		Go to	?	Errors	Warnings
	CPU_1516F					0	0
	Program blocks					0	0
	MOTOR_MANUAL (FC1)	Block was successfully compiled.					12:16:44 PM
	Main (OB1)	Block was successfully compiled.					12:16:44 PM
		Compiling completed (errors: 0; warnings: 0)					12:16:44 PM

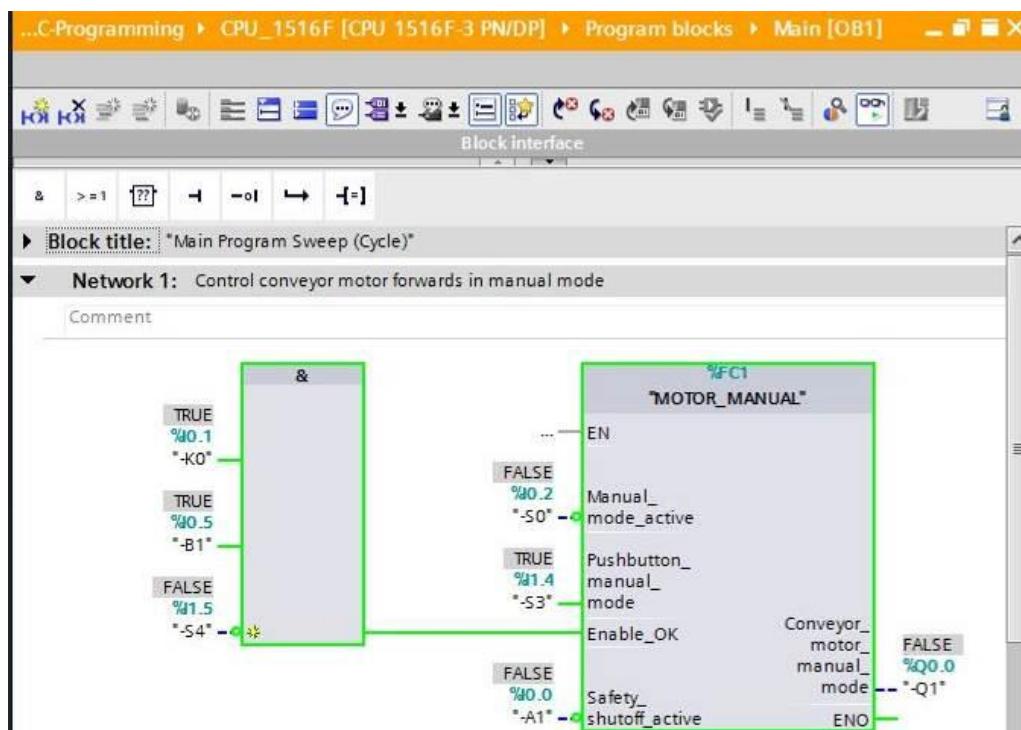
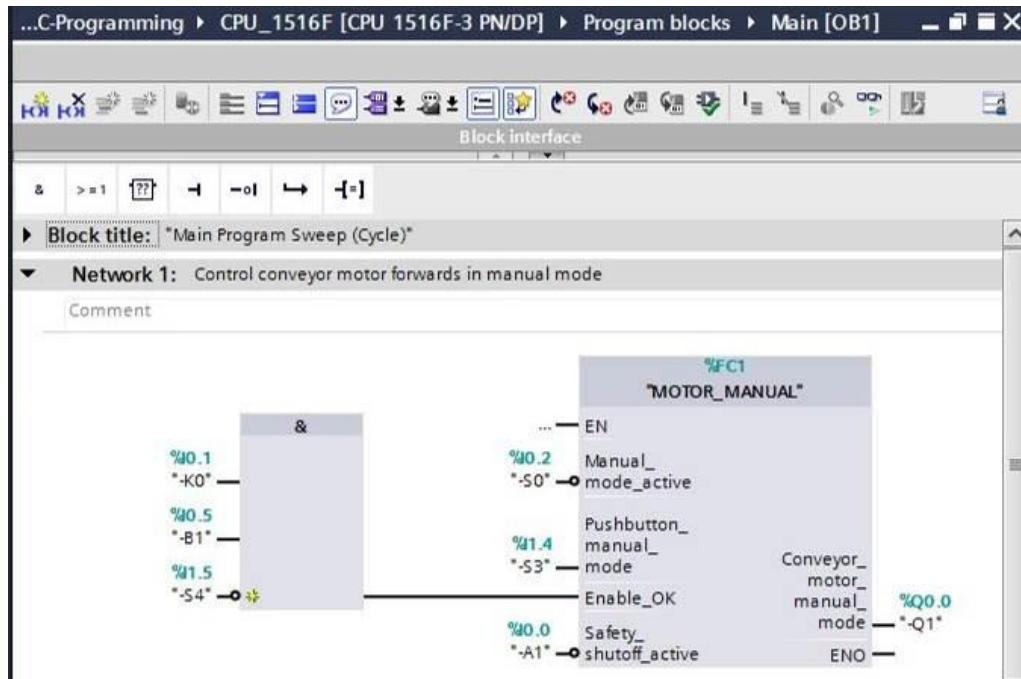
7.11 Download the program

- After successful compilation, the complete controller with the created program, as previously described in the modules for hardware configuration, can be downloaded



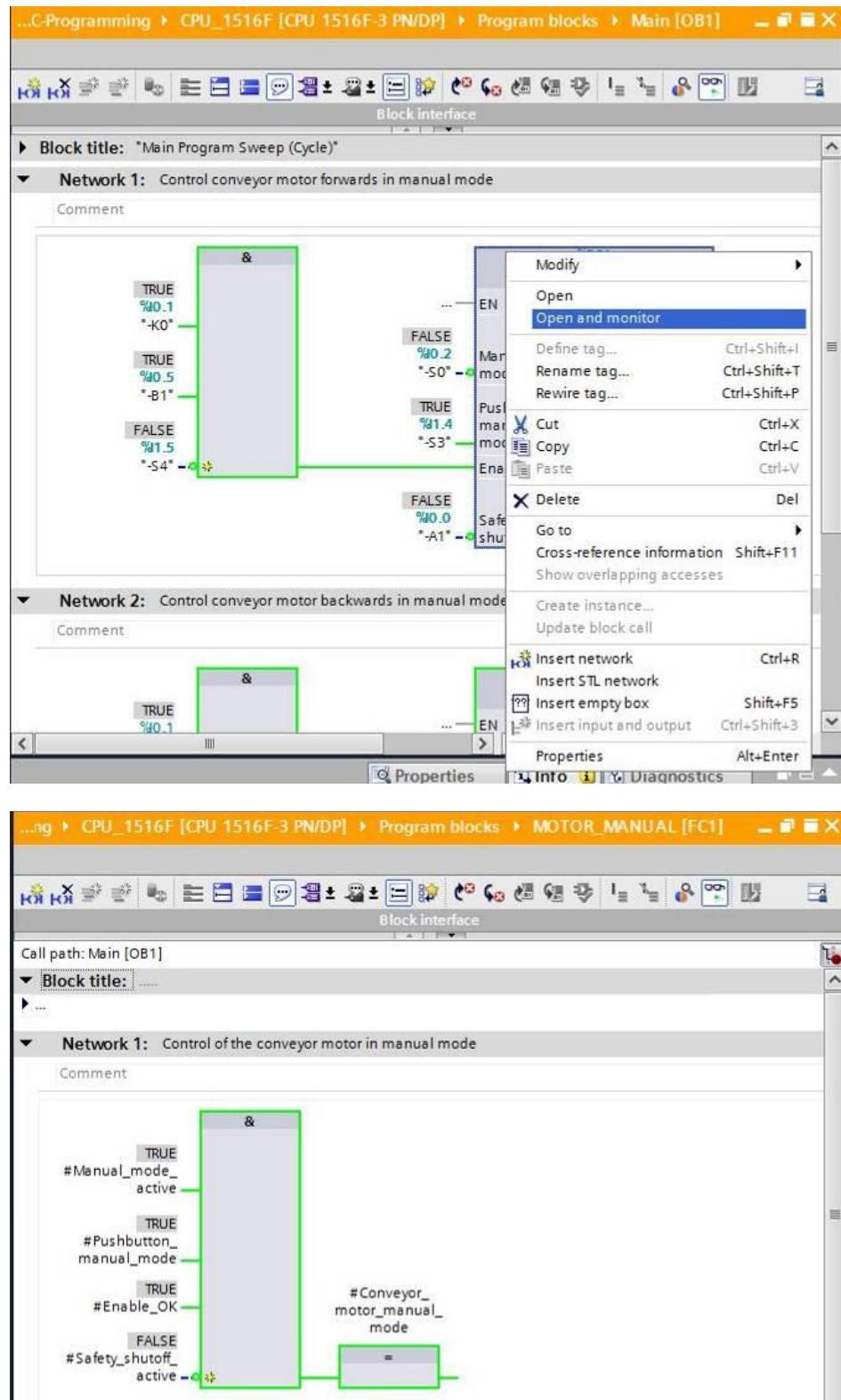
7.12 Monitor program blocks

- The desired block must be open for monitoring the downloaded program. The monitoring can be activated/deactivated by clicking the icon. (→ Main [OB1] →



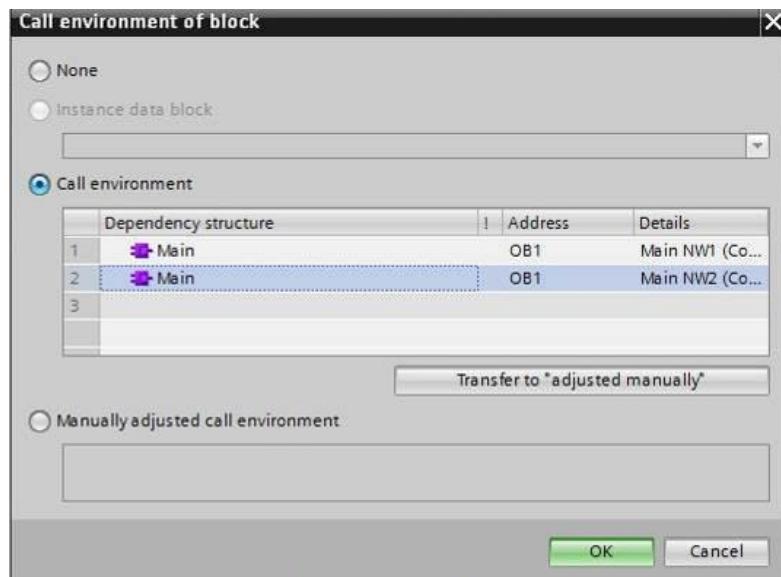
Note: The monitoring here is signal-related and controller-dependent. The signal states at the terminals are indicated with TRUE or FALSE.

- The "MOTOR_MANUAL" [FC1] function called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking (→ "MOTOR_MANUAL" [FC1] → Open and monitor).



Note: The monitoring here is function-related and controller-independent. The actuation of sensors and the station status are shown here with TRUE or FALSE.

- If a particular point of use of the "MOTOR_MANUAL" [FC1] function is to be monitored, the call environment can be selected using the  icon (→  → Call environment → OK)



8 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Activate conveyor manual mode conveyor forward (-S3 = 1) then conveyor motor forwards fixed speed (-Q1 = 1)	
4	Same as 3 but activate EMERGENCY OFF (-A1 = 0) → -Q1 = 0	
5	Same as 3 but AUTO mode (-S0 = 1) → -Q1 = 0	
6	Same as 3 but switch off station (-K0 = 0) → -Q1 = 0	
7	Same as 3 but cylinder not retracted (-B1 = 0) → -Q1 = 0	
8	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Activate conveyor manual mode reverse (-S4 = 1) then conveyor motor backwards fixed speed (-Q2 = 1)	
9	Same as 8 but activate EMERGENCY OFF (-A1 = 0) → -Q2 = 0	
10	Same as 8 but AUTO mode (-S0 = 1) → -Q2 = 0	
11	Same as 8 but switch off station (-K0 = 0) → -Q2 = 0	
12	Same as 8 but cylinder not retracted (-B1 = 0) → -Q2 = 0	
13	Same as 8 but also activate manual mode conveyor forwards (-S3 = 1) → -Q1 = 0 and -Q2 = 0	
14	Project successfully archived	



Training Curriculum

TIA Portal Module 005
Basics of FB Programming
with SIMATIC S7-1500

Table of contents

1	Goal.....	176
2	Prerequisite.....	176
3	Required hardware and software.....	176
4	Theory	177
4.1	Operating system and application program.....	177
4.2	Organization blocks.....	177
4.3	Process image and cyclic program processing	179
4.4	Functions	180
4.5	Function blocks and instance data blocks.....	180
4.6	Global data blocks.....	181
4.7	Library-compatible code blocks	182
4.8	Programming languages	183
5	Task.....	184
6	Planning.....	184
6.1	EMERGENCY STOP	184
6.2	Automatic mode - Conveyor motor	185
7	Structured step-by-step instructions.....	185
7.1	Retrieve an existing project	185
7.2	Create a new tag table	186
7.3	Create new tags within a tag table.....	188
7.4	Import "Tag_table_sorting_station"	189
7.5	Create function block FB1 "MOTOR_AUTO" for the conveyor motor in automatic mode	191
7.6	Define the interface of FB1 "MOTOR_AUTO"	192
7.7	Program FB1: MOTOR_AUTO	194
7.8	Program organization block OB1 – Control of the forward belt tracking in automatic mode.....	201
7.9	The result in the LAD (Ladder Logic) programming language has the following appearance. .	205
7.10	Save and compile the program.....	206
7.11	Download the program	206
7.12	Monitor program blocks.....	207
8	Checklist	210

BASICS OF FB PROGRAMMING

1 Goal

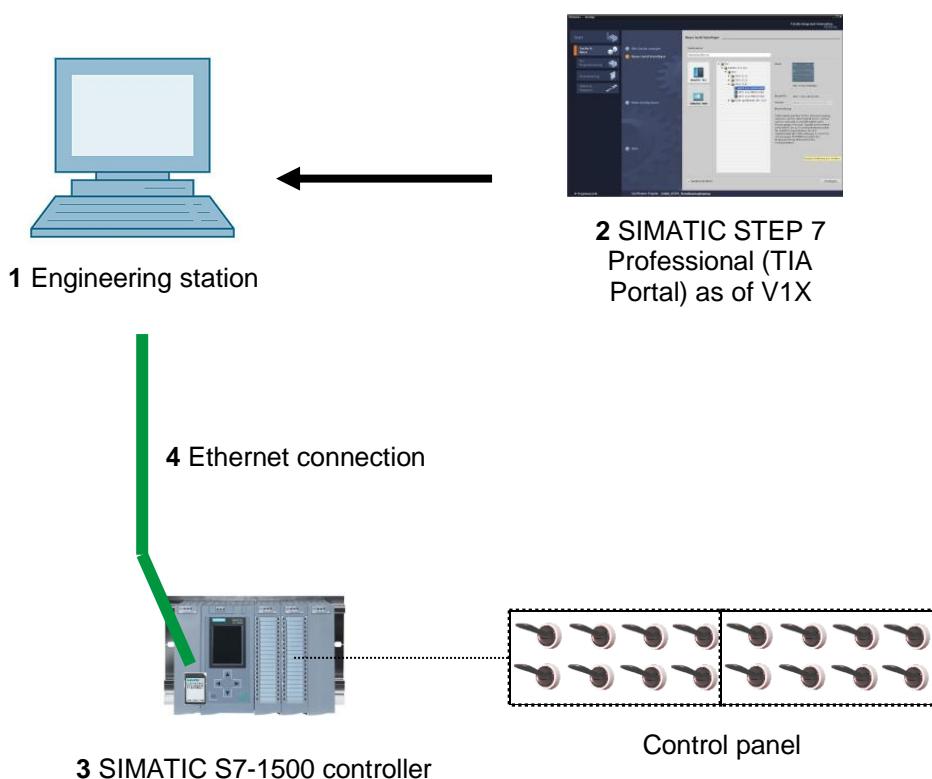
In this chapter, you will get to know the basic elements of a control program – the **organization blocks (OBs)**, **functions (FCs)**, **function blocks (FBs)** and **data blocks (DBs)**. In addition, you will be introduced to **library-compatible** function und function block programming. You will get to know the **Function Block Diagram (FBD)** programming language and use it to program a function block (FB1) and an organization block (OB1).

2 Prerequisite

This chapter builds on the hardware configuration of SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations that have digital input and output cards can be used.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 Operating system and application program

Every controller (CPU) contains an **operating system**, which organizes all functions and processes of the CPU that are not associated with a specific control task. The tasks of the operating system include the following:

- Performing a warm restart
- Updating the process image of the inputs and outputs
- Cyclically calling the user program
- Detecting interrupts and calling interrupt OBs
- Detecting and handling errors
- Managing memory areas

The operating system is an integral component of the CPU and comes pre-installed.

The **user program** contains all functions that are necessary for executing your specific automation task. The tasks of the user program include the following:

- Checking the basic requirements for a warm restart using startup OBs
- Processing of process data, i.e. activation of output signals as a function of the input signal states
- Reaction to interrupts and interrupt inputs
- Error handling during normal program execution

4.2 Organization blocks

Organization blocks (OBs) form the interface between the operating system of the controller (CPU) and the application program. They are called from the operating system and control the following operations:

- Cyclic program processing (e.g. OB1)
- Startup characteristics of the controller
- Interrupt-driven program processing
- Error handling

A project must have ***an organization block for cyclic program processing*** at a minimum. An OB is called by a **start event** as shown in Figure 1. In addition, the individual OBs have defined priorities so that, for example, an OB82 for error handling can interrupt the cyclic OB1.

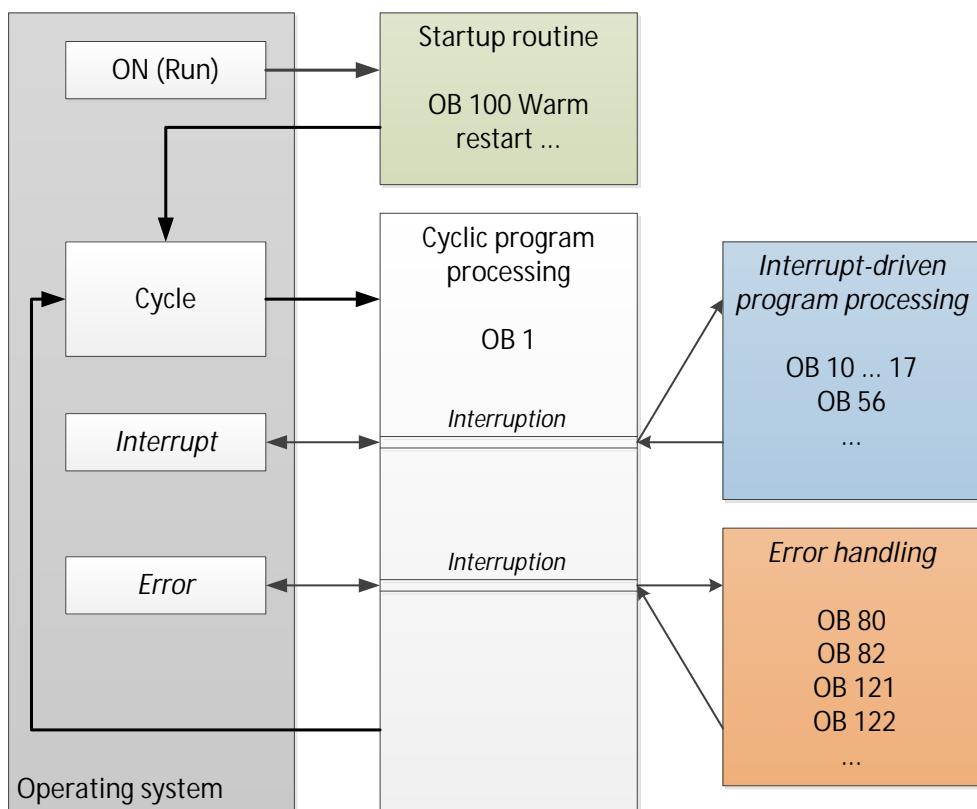


Figure 1:

Start events in the operating system and OB calls

When a start event occurs, the following reactions are possible:

- If an OB has been assigned to the event, this event triggers the execution of the assigned OB. If the priority of the assigned OB is greater than the priority of the OB that is currently being executed, it is executed immediately (interrupt). If not, the assigned OB waits until the higher-priority OB has been completely executed.
- If an OB is not assigned to the event, the default system reaction is performed.

Table 1 gives a couple of examples of start events for a SIMATIC S7-1500, their possible OB number(s) and the default system reaction in the event the organization block is not present in the controller.

Start event	Possible OB numbers	Default system reaction
Startup	100, ≥ 123	Ignore
Cyclic program	1, ≥ 123	Ignore
Time-of-day interrupt	10 to 17, ≥ 123	-
Update interrupt	56	Ignore
Scan cycle monitoring time exceeded once	80	STOP
Diagnostic interrupt	82	Ignore
Programming error	121	STOP
IO access error	122	Ignore

Table 1: OB numbers for various start events

4.3 Process image and cyclic program processing

When the cyclic user program addresses the inputs (I) and outputs (O), it does not query the signal states directly from the input/output modules. Instead, it accesses a memory area of the CPU. This memory area contains an image of the signal states and is called the **process image**.

The cyclic program processing sequence is as follows:

1. At the start of the cyclic program, a query is sent to determine whether or not the individual inputs are energized. This status of the inputs is stored in the **process image of the inputs (PII)**. In doing so, the information 1 or "High" is stored for energized inputs and the information 0 or "Low" for de-energized inputs.
2. The CPU now executes the program stored in the cyclic organization block. For the required input information, the CPU accesses the previously read **process image of the inputs (PII)** and the results of logic operation (RLOs) are written to a so-called **process image of the outputs (PIQ)**.
3. At the end of the cycle, the **process image of the outputs (PIQ)** is transferred as the signal state to the output modules and these are energized or de-energized. The sequence then continues again with Item 1.

1. Save status of inputs in PII.

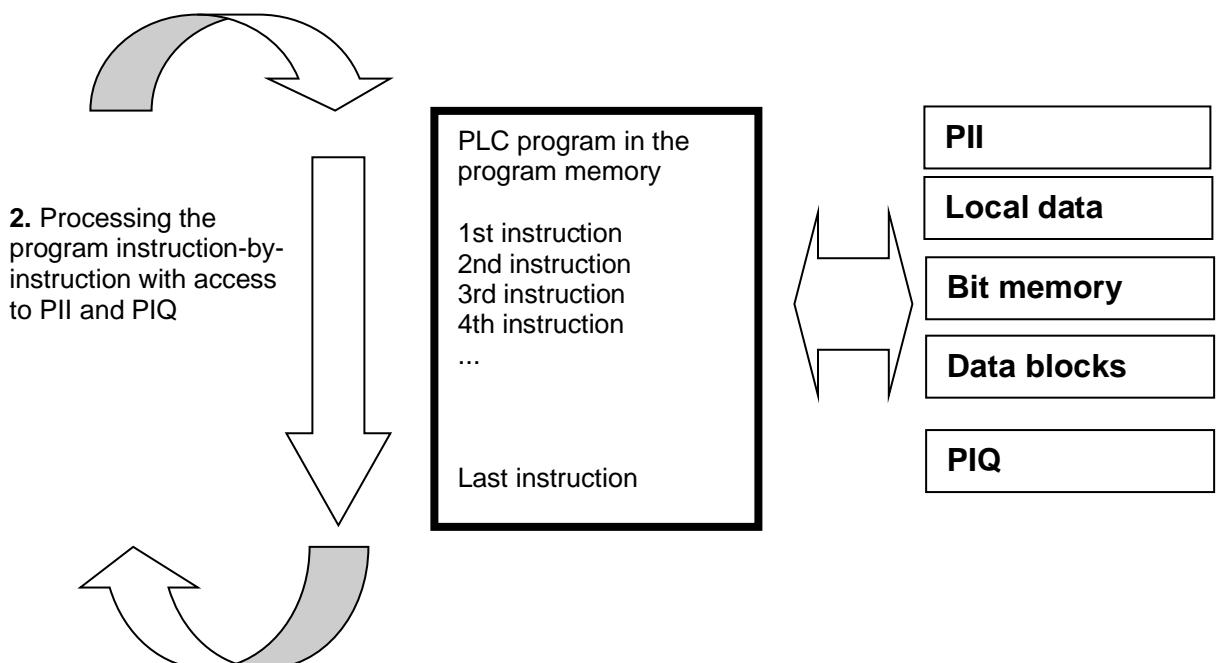


Figure 2: Cyclic program processing

Note: The time the CPU needs for this sequence is called *cycle time*. This depends, in turn, on the number and type of instructions and the processor performance of the controller.

4.4 Functions

Functions (FCs) are logic blocks without memory. They **have no data memory** in which values of block parameters can be stored. Therefore, all interface parameters must be connected when a function is called. To store data permanently, global data blocks must be created beforehand.

A function contains a program that is executed whenever the function is called from another code block.

Functions can be used, for example, for the following purposes:

- Math functions – that return a result dependent on input values.
- Technological functions – such as individual controls with binary logic operations.

A function can also be called several times at different points within a program.

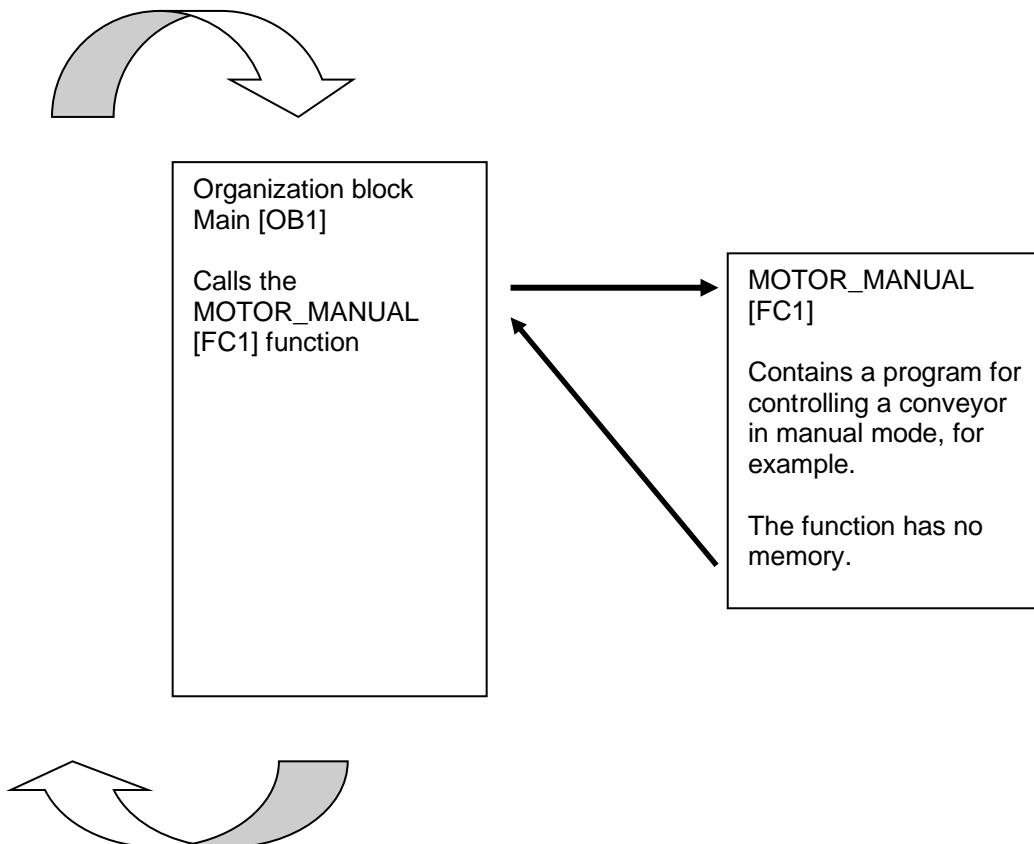


Figure 3: Function with call from organization block Main [OB1]

4.5 Function blocks and instance data blocks

Function blocks are code blocks that store their input, output and in-out tags as well as static tags permanently in instance data blocks, so that they **are available after the block has been executed**. For this reason, they are also referred to as blocks with "memory".

Function blocks can also operate with temporary tags. Temporary tags are not stored in the instance DB, however. Instead, they are only available for one cycle.

Function blocks are used for tasks that cannot be implemented with functions:

- Whenever timers and counters are required in the blocks.
- Whenever information must be saved in the program, such as pre-selection of the operating mode with a button.

Function blocks are always executed when called from another code block. A function block can also be called several times at different points within a program. This facilitates the programming of frequently recurring complex functions.

A call of a function block is referred to as an instance. Each instance of a function block is assigned a memory area that contains the data that the function block uses. This memory is made available by data blocks created automatically by the software.

It is also possible to provide memory for multiple instances in one data block in the form of a **multi-instance**. The maximum size of instance data blocks varies depending on the CPU. The tags declared in the function block determine the structure of the instance data block.

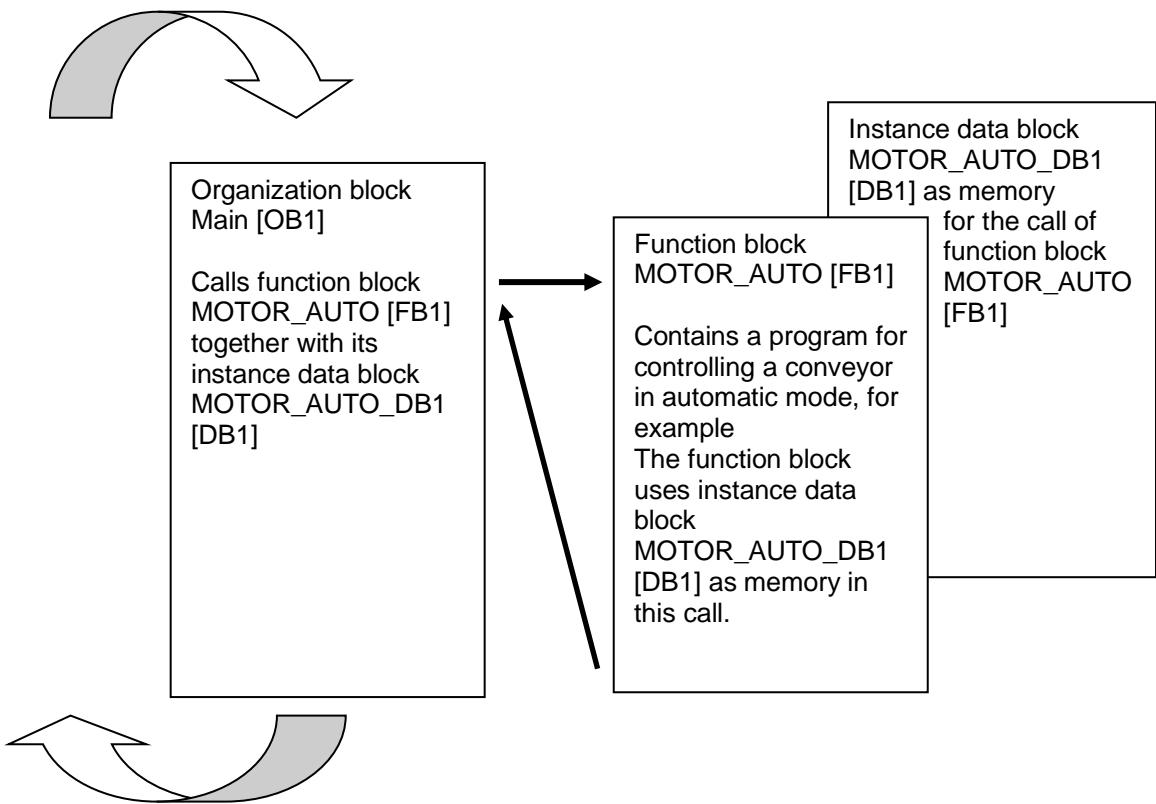


Figure 4: Function block and instance with call from organization block Main [OB1]

4.6 Global data blocks

In contrast to logic blocks, data blocks contain no instructions. Rather, they serve as memory for user data.

Data blocks thus contain variable data that is used by the user program. You can define the structure of global data blocks as required.

Global data blocks store data that can be used **by all other blocks** (see Figure 5). Only the associated function block should access instance data blocks. The maximum size of data blocks varies depending on the CPU.

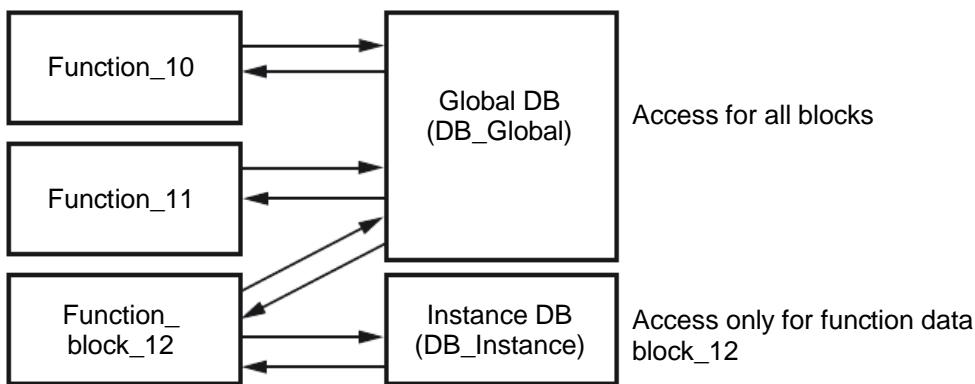


Figure 5: Difference between global DB and instance DB.

Application examples for **global data blocks** are:

- Saving of information about a storage system. "Which product is located where?"
- Saving of recipes for particular products.

4.7 Library-compatible code blocks

A user program can be created with linear or structured programming. **Linear programming** writes the entire user program in the cycle OB, but is only suitable for very simple programs for which other less expensive control systems, such as LOGO!, can now be used.

Structured programming is always recommended for more complex programs. Here, the overall automation task can be broken down into small sub-tasks in order to implement a solution for them in functions and function blocks.

In this case, library-compatible logic blocks should be created preferentially. This means that the input and output parameters of a function or function block are defined generally and only supplied with the current global tags (inputs/outputs) when the block is used.

The screenshot shows the TIA Portal interface for a function block named "MOTOR_AUTO [FB1]". The top part displays the block's properties and configuration. The "Inputs" section lists the following parameters:

Name	Data type	Default value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1 □ Input							
2 □ Automatic_mode_act...	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Automatic mode activated
3 □ Start	Bool	false	Non-retain	<input type="checkbox"/>	<input checked="" type="checkbox"/>		Pushbutton automatic start
4 □ Stop	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Pushbutton automatic stop
5 □ Enable_OK	Bool	false	Non-retain	<input type="checkbox"/>	<input type="checkbox"/>		All enable conditions OK
6 □ Safety_shutoff_active	Bool	false	Non-retain	<input type="checkbox"/>	<input type="checkbox"/>		Safety shutoff active e.g. emergency sto...
7 □ Output							
8 □ Conveyor_motor_aut...	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Control of the conveyor motor in autom...

The bottom part shows the ladder logic for "Network 1: Memory automatic_start_stop and control of the conveyor motor in automatic mode". The logic includes contacts for #Stop, #Safety_shutoff_active, and #Automatic_mode_active, and coil assignments for #Memory_automatic_start_stop, #Conveyor_motor_automatic_mode, and #Enable_OK.

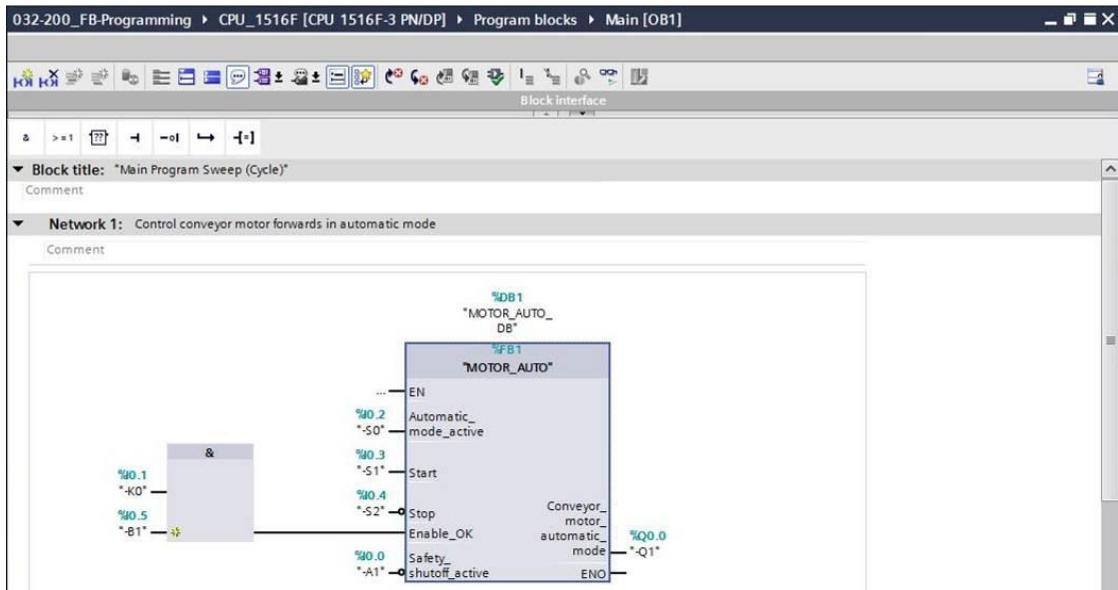


Figure 6: Library-compatible function with call in OB1

4.8 Programming languages

The available programming languages for programming functions are Function Block Diagram (FBD), Ladder Logic (LAD), Statement List (STL) and Structured Control Language (SCL). For function blocks, the GRAPH programming language is additionally available for programming graphical step sequences.

The **Function Block Diagram (FBD)** programming language will be presented in the following.

FBD is a graphical programming language. The representation is based on electronic switching systems. The program is mapped in networks. A network contains one or more logic operation paths. Binary and analog signals are linked by boxes. The graphical logic symbols known from Boolean algebra are used to represent the binary logic.

You can use binary functions to query binary operands and to logically combine their signal states. The following instructions are examples of binary functions: "AND operation", "OR operation" and "EXCLUSIVE OR operation". These are shown in Figure 7.



Figure 7: Binary functions in FBD and associated logic table

You can thus use simple instructions, for example, to control binary outputs, evaluate edges and execute jump functions in the program.

Program elements such as IEC timers and IEC counters provide complex instructions.

The empty box serves as a placeholder in which you can select the required instruction.

Enable input EN (enable)/ Enable output ENO (enable output) mechanism:

- An instruction without EN/ENO mechanism is executed independent of the signal state at the box inputs.
- Instructions with EN/ENO mechanism are only executed if enable input "EN input has signal state "1". When the box is processed correctly, enable output "ENO" has signal state "1". As soon as an error occurs during the processing, the "ENO" enable output is reset. If enable input EN is not connected, the box is always executed.

5 Task

The following functions of the sorting station process description will be planned, programmed and tested in this chapter:

- Automatic mode - Conveyor motor

6 Planning

The programming of all functions in OB1 is not recommended for reasons of clarity and reusability. The majority of the program code will therefore be moved into functions (FCs) and function blocks (FBs). The decision on which functions is to be moved to the FB and which is to run in OB 1 is planned below.

6.1 EMERGENCY STOP

The EMERGENCY STOP does not require a separate function. Just like the operating mode, the current state of the EMERGENCY STOP relay can be used directly at the blocks.

6.2 Automatic mode - Conveyor motor

Automatic mode of the conveyor motor is to be encapsulated in a function block (FB) "MOTOR_AUTO". On the one hand, this preserves the clarity of OB1. On the other hand, it enables reuse if another conveyor belt is added to the station. Table 2 lists the planned parameters.

Input	Data	Comment
Automatic_mode_active	BOOL	Automatic mode activated
Start	BOOL	Pushbutton automatic start
Stop	BOOL	Pushbutton automatic stop
Enable_OK	BOOL	All enable conditions OK
Safety_shutoff_active	BOOL	Safety shutoff active, e.g. emergency stop pressed
Output		
Conveyor_motor_automatic_mode	BOOL	Control of the conveyor motor in automatic mode
Static		
Memory_automatic_start_stop	BOOL	Memory used for start/stop automatic mode

Table 2: Parameters for FB "MOTOR_AUTO"

The Memory_automatic_start_stop is latched with Start but only if the reset conditions are not present.

The Memory_automatic_start_stop is reset if Stop is present or safety shutoff is active or automatic mode is not activated (manual mode).

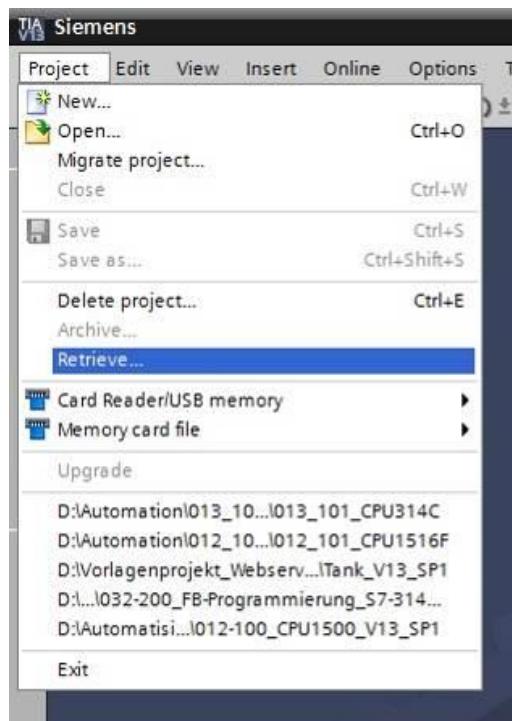
The Conveyor_motor_automatic_mode output is controlled when Memory_automatic_start_stop is set and the enable conditions are met.

7 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

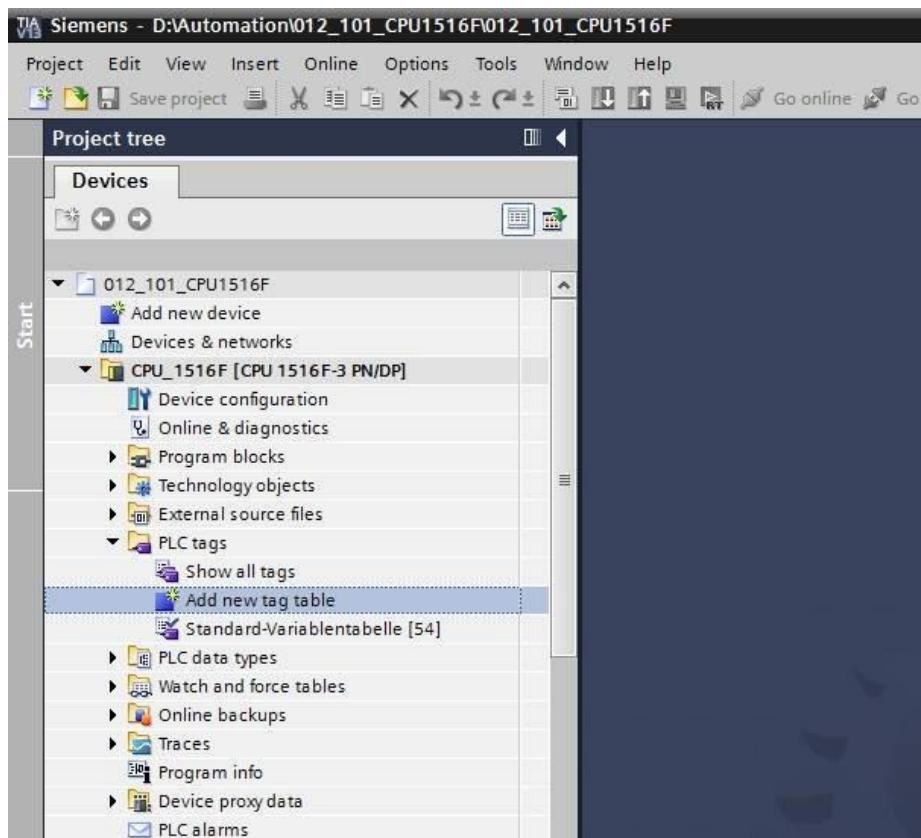
- Before we can start programming the function block (FB) "MOTOR_AUTO", we need a project with a hardware configuration (e.g. SCE_EN_012_101_Hardware_Configuration_S7-1516F_R1502.zap). To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open. (→ Project → Retrieve → Select a .zap archive → Open)



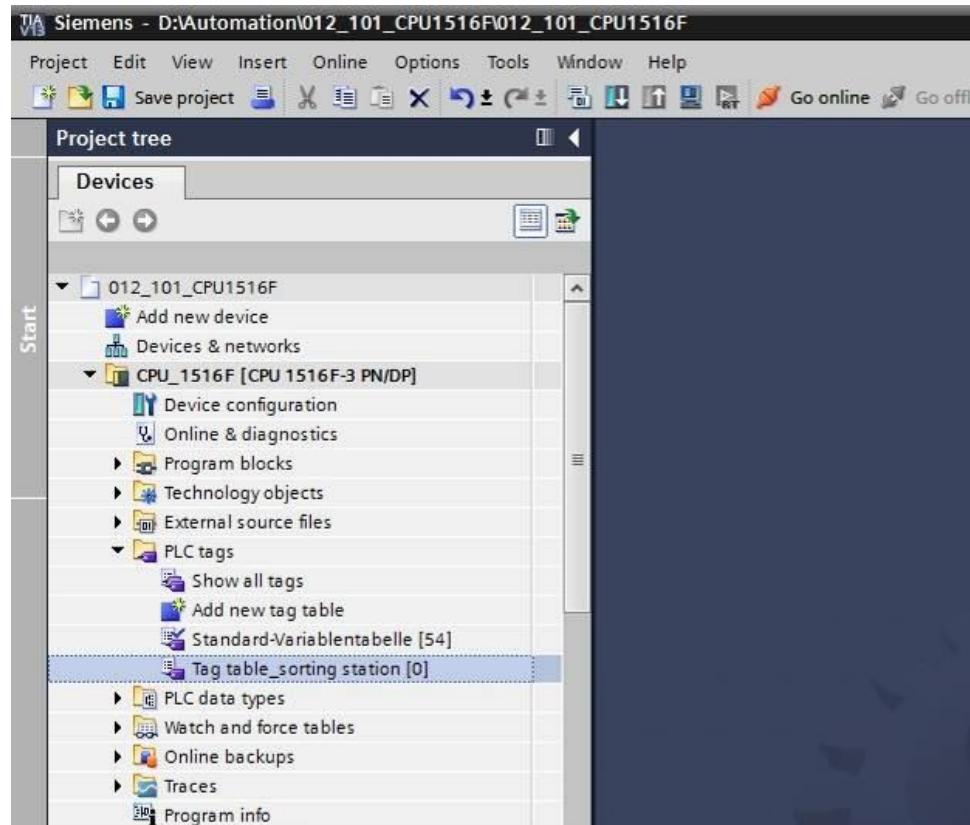
- The next step is to select the target directory where the retrieved project will be stored.
Confirm your selection with "OK". (→ Target directory → OK)

7.2 Create a new tag table

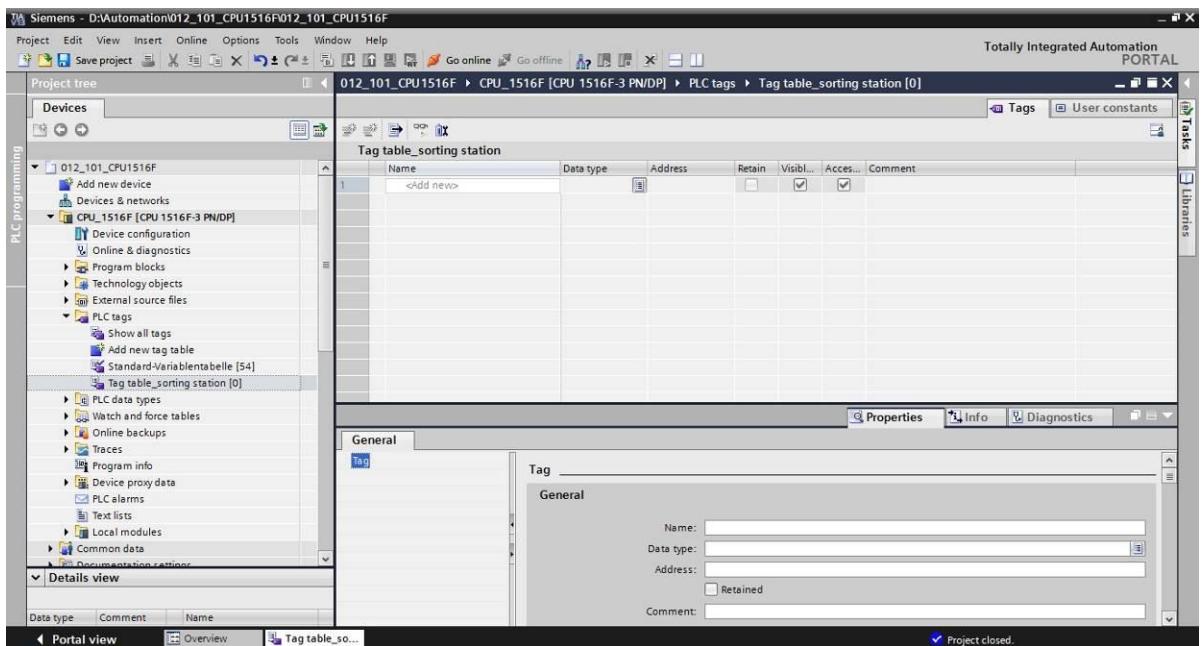
- In the project view, navigate to the → PLC tags of your controller and create a new tag table by double-clicking → Add new tag table.



- Rename the tag table you just created as "Tag_table_sorting_station" (→ right-click "Tag_table_1" → "Rename" → Tag_table_sorting_station).

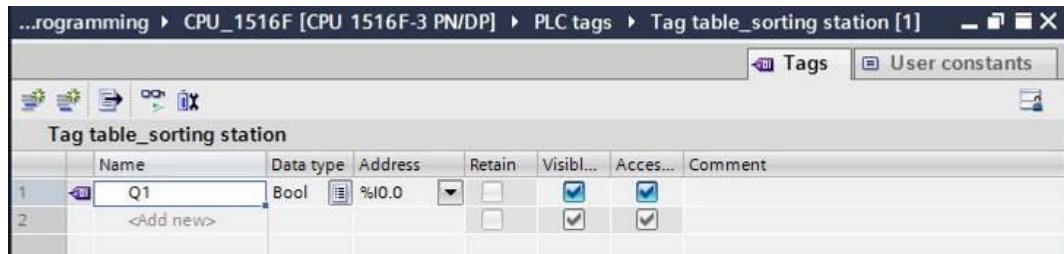


- Open this tag table with a double-click. (→ Tag_table_sorting_station)

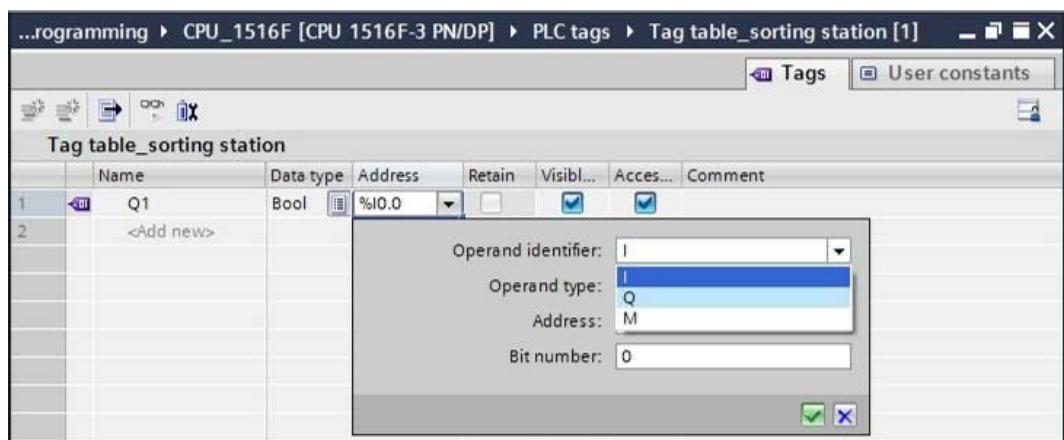


7.3 Create new tags within a tag table

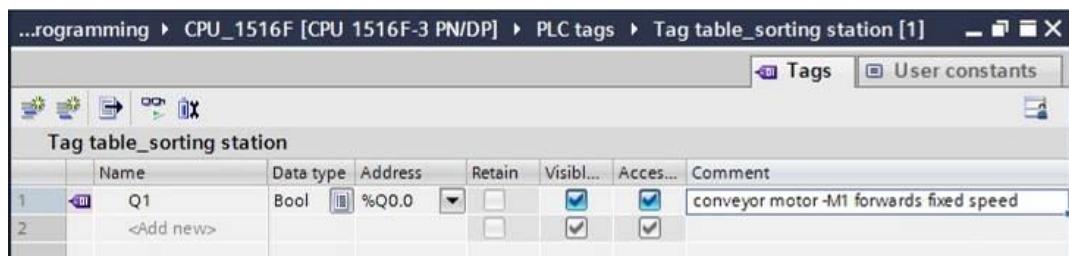
- Add the name Q1 and confirm the entry with the Enter key. If you have not yet created additional tags, TIA Portal now automatically assigns data type "Bool" and address %I0.0 (I 0.0) (→ <Add> → Q1 → Enter).



- Change the address to %Q0.0 (Q 0.0) by entering this directly or by clicking the drop-down arrow to open the Addressing menu, changing the operand identifier to Q and confirming with Enter or by clicking the check mark. (→ %I0.0 → Operand identifier → Q →)



- Enter the "Conveyor motor M1 forwards fixed speed" comment for the tag.



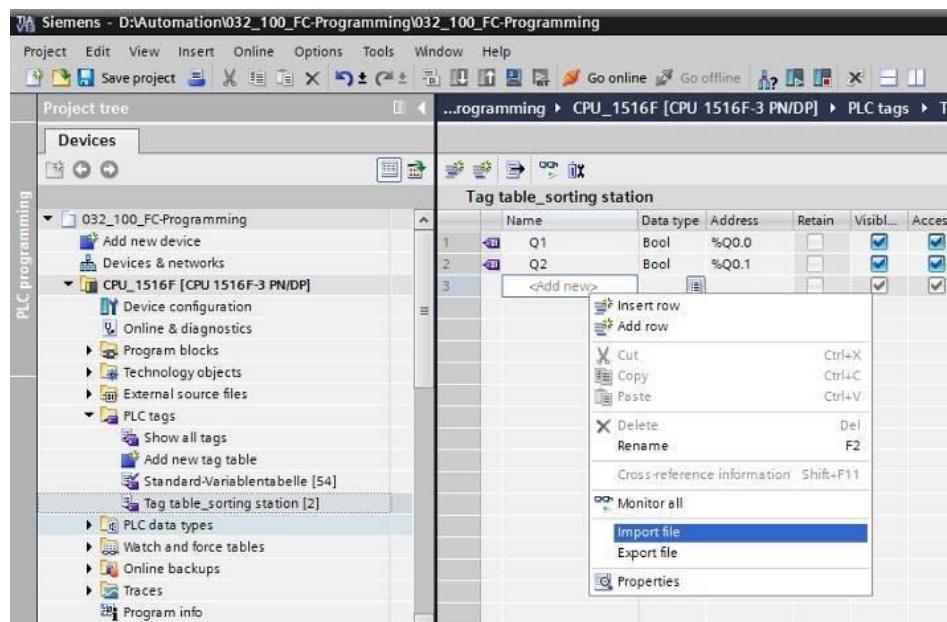
- Add a new Q2 tag in line 2. TIA Portal has automatically assigned the same data type as in line 1 and has incremented the address by 1 to %Q0.1 (Q0.1). Enter the comment "Conveyor motor M1 backwards fixed speed".
(→ <Add> → Q2 → Enter → Comment → Conveyor motor M1 backwards fixed speed)

	Name	Data type	Address	Retain	Visible	Access	Comment
1	Q1	Bool	%Q0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
2	Q2	Bool	%Q0.1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
3	<Add new>			<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

7.4 Import "Tag_table_sorting_station"

→ To insert an existing symbol table, right-click on an empty field of the created "Tag_table_sorting_station". Select "Import file" in the shortcut menu.

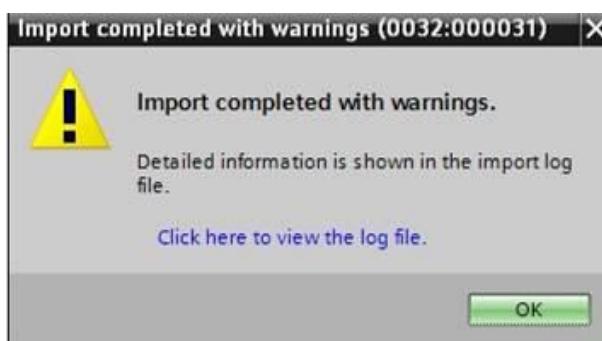
(→ Right-click in an empty field of the tag table → Import file)



→ Select the desired symbol table (e.g. in .xlsx format) and confirm the selection with "Open".

(→ SCE_EN_020-100_Tag_table_sorting_station... → Open)

→ When the import is finished, you will see a confirmation window and have an opportunity to view the log file for the import. Click → OK.



- You can see that some addresses have been highlighted in orange. These are duplicate addresses and the names of the associated tags have been numbered automatically to avoid confusion.
 - Delete the duplicate tags by selecting the lines and pressing the Del key on your keyboard or by selecting "Delete" in the shortcut menu.
- (→ Right-click on selected tags → Delete)

Name	Data type	Address	Retain	Visible...	Access...	Comment
1 -Q1	Bool	%Q0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
2 -Q2	Bool	%Q0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
3 -S4	Bool	%I1.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	return signal emergency stop ok (nc)
4 -S5	Bool	%I1.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	main switch „ON“ (no)
5 -S6	Bool	%I1.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	mode selector manual(0) / automatic(1)
6 -Q1	Bool	%Q0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton automatic start (no)
7 -Q2	Bool	%Q0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton automatic stop (nc)
8 -Q3	Bool	%Q0.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor cylinder -M4 retracted (no)
9 -M2	Bool	%Q0.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor cylinder -M4 extended (nc)
10 -M3	Bool	%Q0.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor motor -M1 active (pulse signal for positioning) (no)
11 -P1	Bool	%Q0.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor part at slide (no)
12 -P2	Bool	%Q0.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor metal part (no)
13 -P3	Bool	%Q0.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor part in front of cylinder -M4 (no)
14 -S1	Bool	%I0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor part at end of conveyor (no)
15 -S2	Bool	%I0.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode conveyor -M1 forwards (no)
16 -S3	Bool	%I0.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode conveyor -M1 backwards (no)
17 -S4	Bool	%I0.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder -M4 retract (no)
18 -S5	Bool	%I0.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder -M4 extend (no)
19 -Q1	Bool	%Q0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
20 -Q2	Bool	%Q0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
21 -Q3	Bool	%Q0.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 variable speed
22 -M2	Bool	%Q0.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder -M4 retract
23 -M3	Bool	%Q0.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder -M4 extend
24 -P1	Bool	%Q0.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „main switch on“
25 -P2	Bool	%Q0.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „manual mode“
26 -P3	Bool	%Q0.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „automatic mode“

- You now have a complete symbol table of the digital inputs and outputs in front of you.
- Save your project under the name 032-100_FCProgramming.

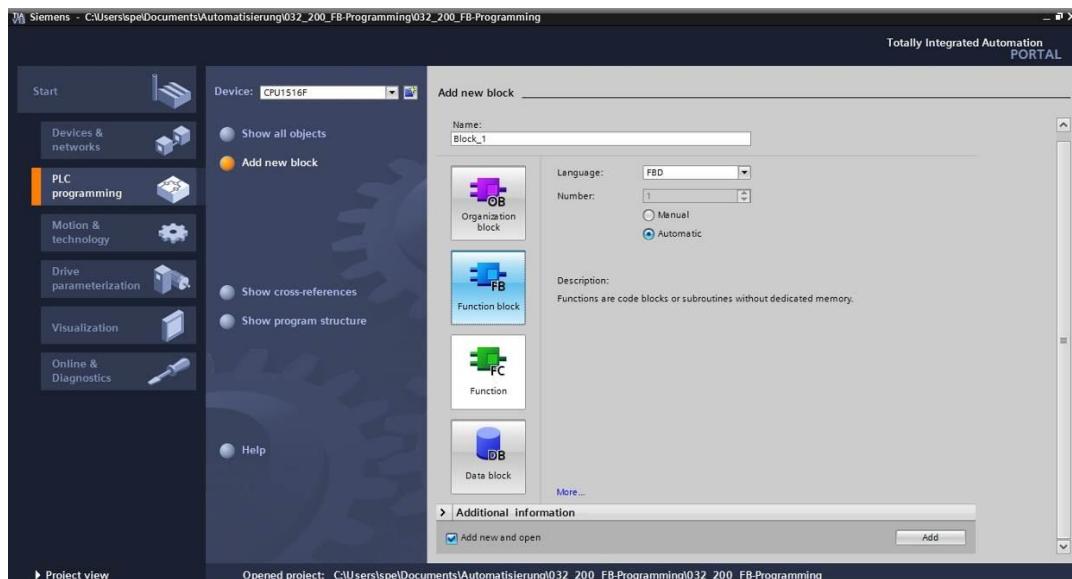
(→ Project → Save as ... → 032-200_FBFProgramming → Save)

Name	Data type	Address	Retain	Visible...	Access...	Comment
1 -A1	Bool	%I0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	return signal emergency stop ok (nc)
2 -K0	Bool	%I0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	main switch „ON“ (no)
3 -S0	Bool	%I0.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	mode selector manual(0) / automatic(1)
4 -S1	Bool	%I0.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton automatic start (no)
5 -S2	Bool	%I0.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton automatic stop (nc)
6 -B1	Bool	%I0.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor cylinder -M4 retracted (no)
7 -B2	Bool	%I0.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor cylinder -M4 extended (nc)
8 -B3	Bool	%I0.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor motor -M1 active (pulse signal for ...)
9 -B4	Bool	%I1.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor part at slide (no)
10 -B5	Bool	%I1.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor metal part (no)
11 -B6	Bool	%I1.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor part in front of cylinder -M4 (no)
12 -B7	Bool	%I1.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor part at end of conveyor (no)
13 -B8	Bool	%I1.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode conveyor -M1...
14 -B9	Bool	%I1.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode conveyor -M1...
15 -S5	Bool	%I1.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder -M4 re...
16 -S6	Bool	%I1.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder -M4 ex...
17 -Q1	Bool	%Q0.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 forwards fixed speed
18 -Q2	Bool	%Q0.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 backwards fixed speed
19 -Q3	Bool	%Q0.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor -M1 variable speed
20 -M2	Bool	%Q0.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder -M4 retract
21 -M3	Bool	%Q0.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder -M4 extend
22 -P1	Bool	%Q0.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „main switch on“
23 -P2	Bool	%Q0.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „manual mode“
24 -P3	Bool	%Q0.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „automatic mode“
25 -P4	Bool	%Q1.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „emergency stop activated“
26 -P5	Bool	%Q1.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display „automatic mode started“

7.5 Create function block FB1 "MOTOR_AUTO" for the conveyor motor in automatic mode

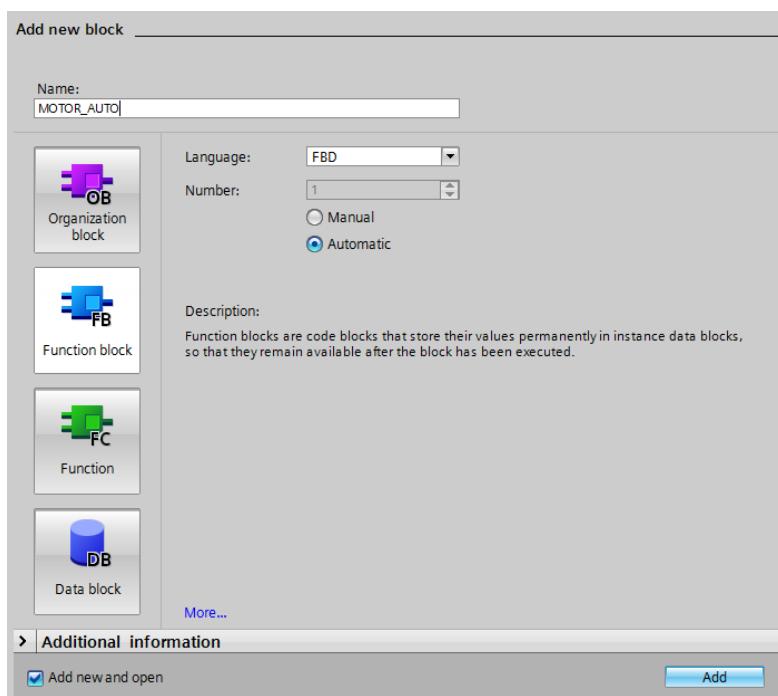
- In the PLC programming section of the portal view, click "Add new block" to create a new function block.

(→ PLC programming → Add new block → )



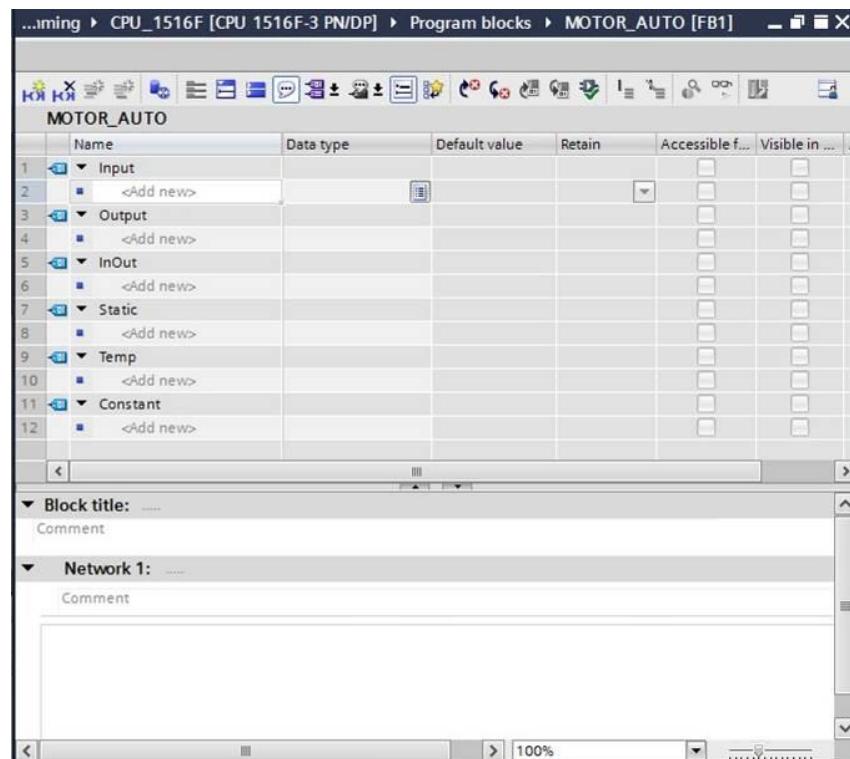
- Rename your new block to: "MOTOR_AUTO", set the language to FBD and keep automatic assignment of the number. Select the "Add new and open" check box. You are then taken automatically to your created function block in the project view. Click "Add".

(→ Name: MOTOR_AUTO → Language: FBD → Number: Automatic → Add new and open → Add)

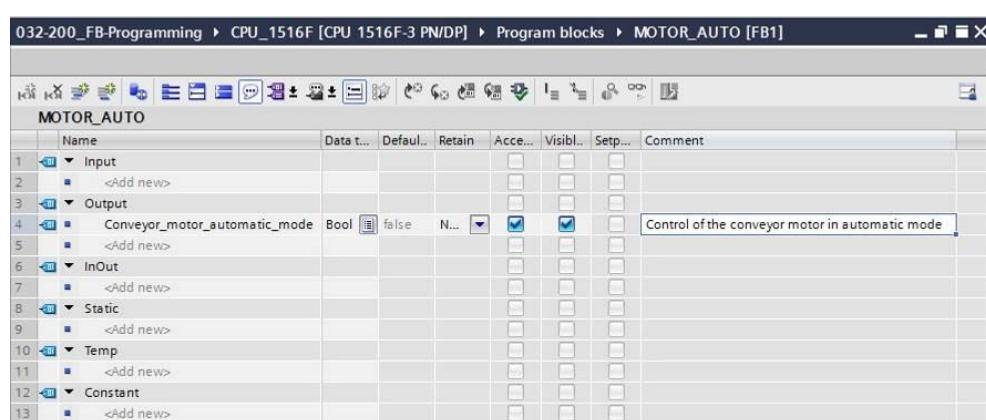


7.6 Define the interface of FB1 "MOTOR_AUTO"

- If you selected "Add new and open", the project view opens with a window for creating the block you just added.
- You can find the interface description of your function block in the upper section of your programming view.



- A binary output signal is needed for controlling the conveyor motor. For this reason, we first create local output tag #Conveyor_motor_automatic_mode of the "Bool" type. Enter the comment "Control of the conveyor motor in automatic mode" for the parameter.
- (→ Output: Conveyor_motor_automatic_mode → Bool → Control of the conveyor motor in automatic mode)



- Add parameter #Automatic_mode_active as the input interface under Input and confirm the entry with the Enter key or by exiting the entry field. Data type "Bool" is assigned

automatically. This will be retained. Next, enter the associated comment "Automatic mode activated".

(→ Automatic_mode_active → Bool → Automatic mode activated)

- Add parameters #Start, #Stop, #Enable_OK and #Safety_shutoff_active as additional binary input parameters under Input and check their data types. Add descriptive comments.

MOTOR_AUTO								
	Name	Data t...	Default...	Retain	Access...	Visible...	Setup...	Comment
1	Input							
2	Automatic_mode_active	Bool	false	Non-r...	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Automatic mode activated
3	Start	Bool	false	Non-r...	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Pushbutton automatic start
4	Stop	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Pushbutton automatic stop
5	Enable_OK	Bool	false	Non-r...	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	All enable conditions OK
6	Safety_shutoff_active	Bool	<input type="checkbox"/>	false	N...	<input type="checkbox"/>	<input type="checkbox"/>	Safety shutoff active e.g. emergency stop operated
7	<Add new>							
8	Output							
9	Conveyor_motor_automatic_mode	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Control of the conveyor motor in automatic mode
10	<Add new>							
11	InOut							
12	<Add new>							
13	Static							
14	<Add new>							
15	Temp							
16	<Add new>							

The conveyor is started and stopped with pushbuttons. We therefore need a "Static" tag as a memory. Under Static, add tag #Memory_automatic_start_stop and confirm the entry with the Enter key or by exiting the entry field. Data type "Bool" is assigned automatically. This will be retained. Enter the associated comment "Memory used for start/stop automatic mode". (→ Memory_automatic_start_stop → Bool → Memory used for start/stop automatic mode)

MOTOR_AUTO								
	Name	Data t...	Default...	Retain	Access...	Visible...	Setup...	Comment
1	Input							
2	Automatic_mode_active	Bool	false	Non-r...	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Automatic mode activated
3	Start	Bool	false	Non-r...	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Pushbutton automatic start
4	Stop	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Pushbutton automatic stop
5	Enable_OK	Bool	false	Non-r...	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	All enable conditions OK
6	Safety_shutoff_active	Bool	<input type="checkbox"/>	false	N...	<input type="checkbox"/>	<input type="checkbox"/>	Safety shutoff active e.g. emergency stop operated
7	<Add new>							
8	Output							
9	Conveyor_motor_automatic_mode	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Control of the conveyor motor in automatic mode
10	<Add new>							
11	InOut							
12	<Add new>							
13	Static							
14	Memory_automatic_start_stop	Bool	<input type="checkbox"/>	false	N...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Memory used for start/stop automatic mode

- For purposes of program documentation, assign the block title, a block comment and a helpful network title for Network 1.

(→ Block title: Motor control in automatic mode → Network 1:

Memory_automatic_start_stop and control of the conveyor motor in automatic mode)

MOTOR_AUTO

Name	Data type	Default value	Retain	Access...	Visible...	Setp...	Comment
1 Input							
2 Automatic_mode_active	Bool	false	Non-retain				Automatic mode activated
3 Start	Bool	false	Non-retain				Pushbutton automatic start
4 Stop	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Pushbutton automatic stop
5 Enable_OK	Bool	false	Non-retain				All enable conditions OK
6 Safety_shutoff_active	Bool	false	Non-retain				Safety shutoff active e.g. emergency stop
7 Output							
8 Conveyor_motor_automatic_mode	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Control of the conveyor motor in automatic mode
9 InOut							
10 <Add new>							
11 Static							
12 Memory_automatic_start_stop	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Memory used for start/stop automation

Block title: Motor control in automatic mode

Conveyor motor in automatic mode:
The bit Memory_automatic_start_stop is set with the input Start, but only if the reset conditions are not fulfilled.
The bit Memory_automatic_start_stop is reset with the input Stop or if the safety shutoff is activated or if the automatic mode is not activated (manual mode).
If Memory_automatic_start_stop is set, the enable conditions are granted and Memory_conveyor_start_stop is set the output Conveyor_motor_automatic_mode is activated. For reasons of energy efficiency the conveyor motor should only run if a part is present. Therefore Memory_conveyor_start_stop is set if there is a part detected in front of Sensor_slide and reset with a negative edge at Sensor_end_of_conveyor or if the safety shutoff is activated or if the automatic mode is not activated (manual mode).

Network 1: Memory automatic_start_stop and control of the conveyor motor in automatic mode

7.7 Program FB1: MOTOR_AUTO

- Below the interface description, you see a toolbar in the programming window with various logic functions and below that an area with networks. We have already specified the block title and the title for the first network there. Programming is performed within the networks using individual logic blocks. Distribution among multiple networks helps to preserve the clarity of the program. In the following, you will get to know the various ways you can insert logic blocks.



- You can see a list of instructions you can use in the program on the right side of your programming window. Under → Basic instructions → Bit logic operations, find function (Assignment) and use a drag-and-drop operation to move it to Network 1 (green line appears, mouse pointer with + symbol).

(→ Instructions → Basic instructions → Bit logic operations →)

MOTOR_AUTO

Name	Data type	Default value	Retain	Access...	Visible...	Setp...	Comment
1 Input							
2 Automatic_mode_active	Bool	false	Non-retain				Automatic mode activated
3 Start	Bool	false	Non-retain				Pushbutton automatic start
4 Stop	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Pushbutton automatic stop
5 Enable_OK	Bool	false	Non-retain				All enable conditions OK
6 Safety_shutoff_active	Bool	false	Non-retain				Safety shutoff active e.g. emergency stop
7 Output							
8 Conveyor_motor_automatic_mode	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Control of the conveyor motor in automatic mode
9 InOut							

Network 1: Memory automatic_start_stop and control of the conveyor motor in automatic mode

Instructions

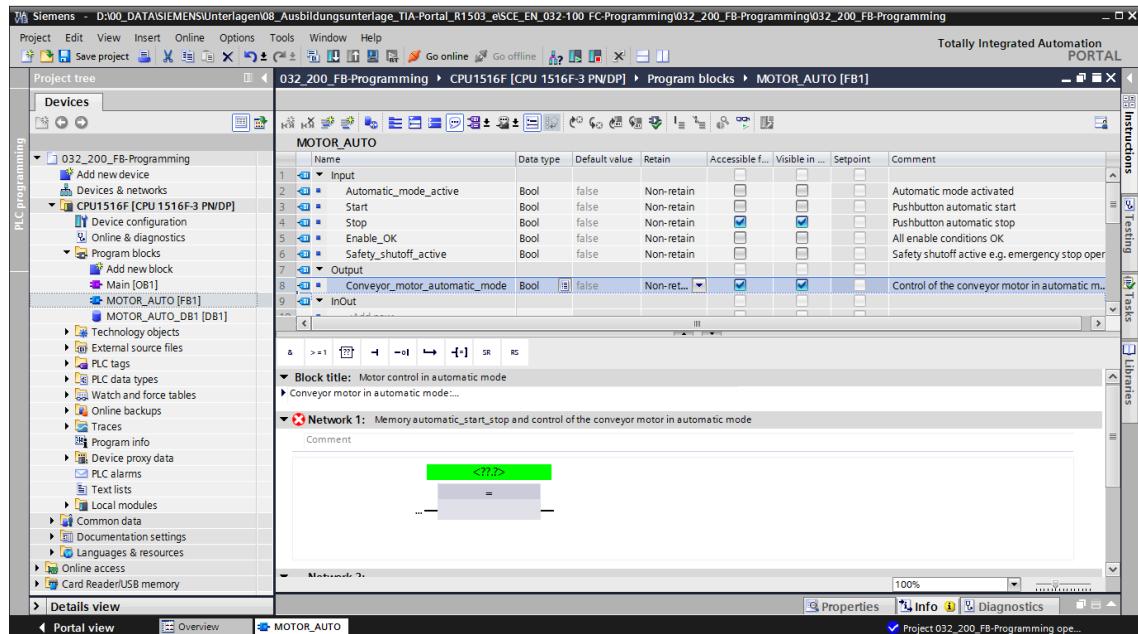
Favorites

Basic instructions

- General
 - AND logic...
 - OR logic...
 - XOR...
 - NOT...
 - NOT-OR...
 - AND-OR...
 - OR-AND...
 - EXCLUSIVE-OR...
 - Assignment...
- Bit logic opera...
 - &
 - >=1
 - x
 - [=]
 - [!=]
 - [R]
 - [S]
 - SET_BF
 - RESET_BF
 - SR
 - RS
 - [P]

- Now use drag-and-drop to move your output parameter #Conveyor_motor_automatic_mode onto <??> above the block you just inserted. The best way to select a parameter in the interface description is by "grabbing" it at the blue symbol .

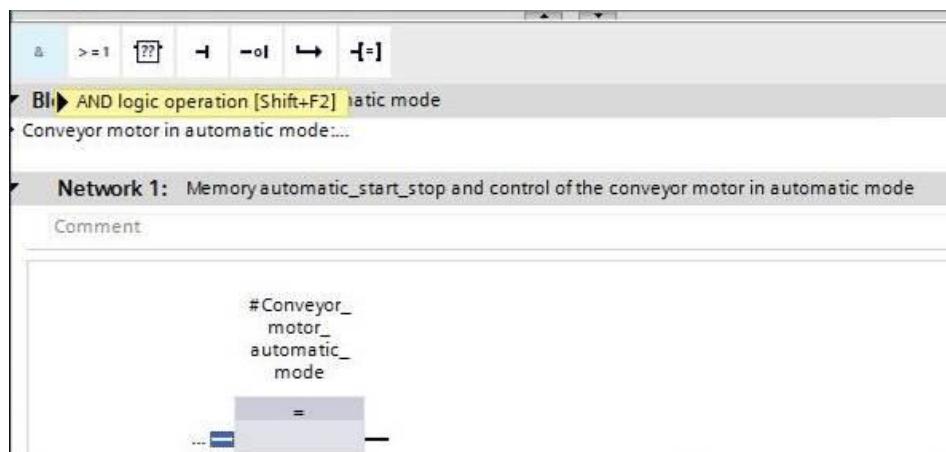
(→  Conveyor_motor_automatic_mode)



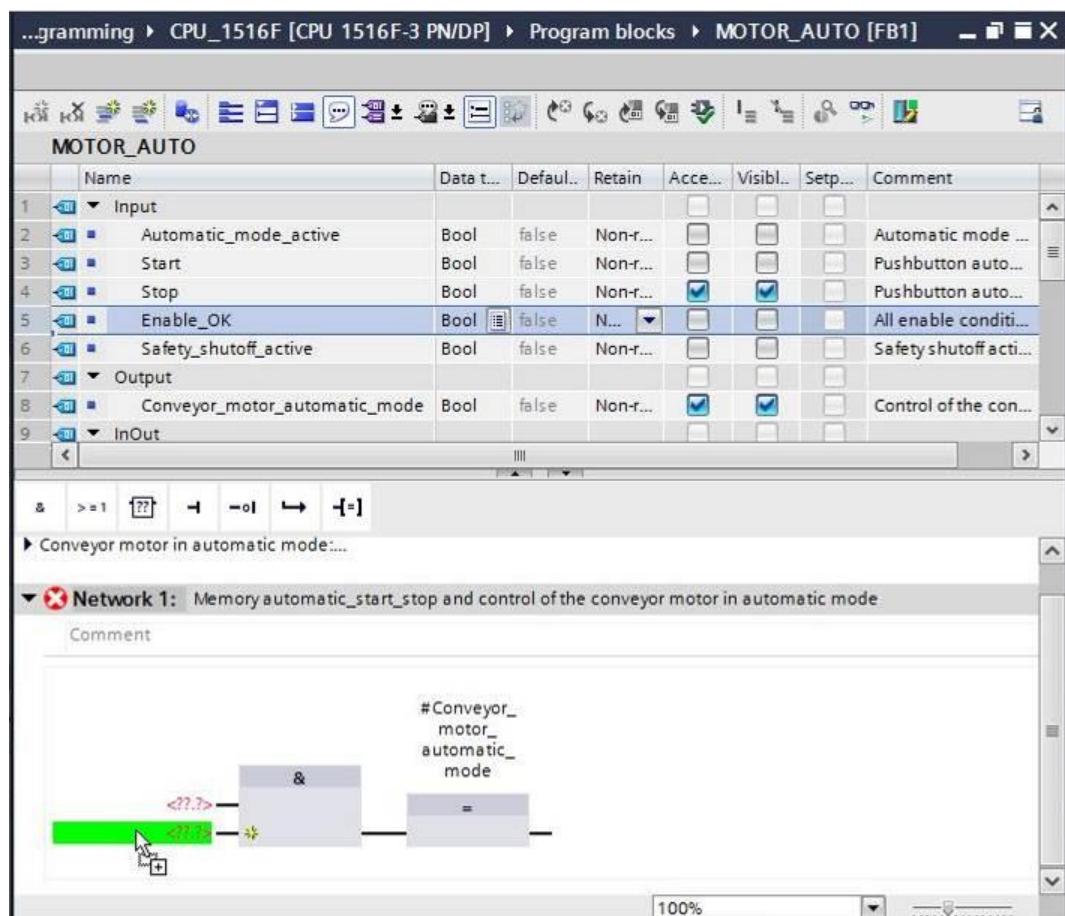
- This determines that the #Conveyor_motor_automatic_mode parameter is written by this block. Still missing, however, are the input conditions so that this actually happens. An SR flip-flop and #Enable_OK parameter are logically combined with an AND logic operation at the input of the assignment block. To do this, first click the input of the block so that the input line has a blue background.



- Click the  icon in your logic toolbar to insert an AND logic operation before your assignment block.

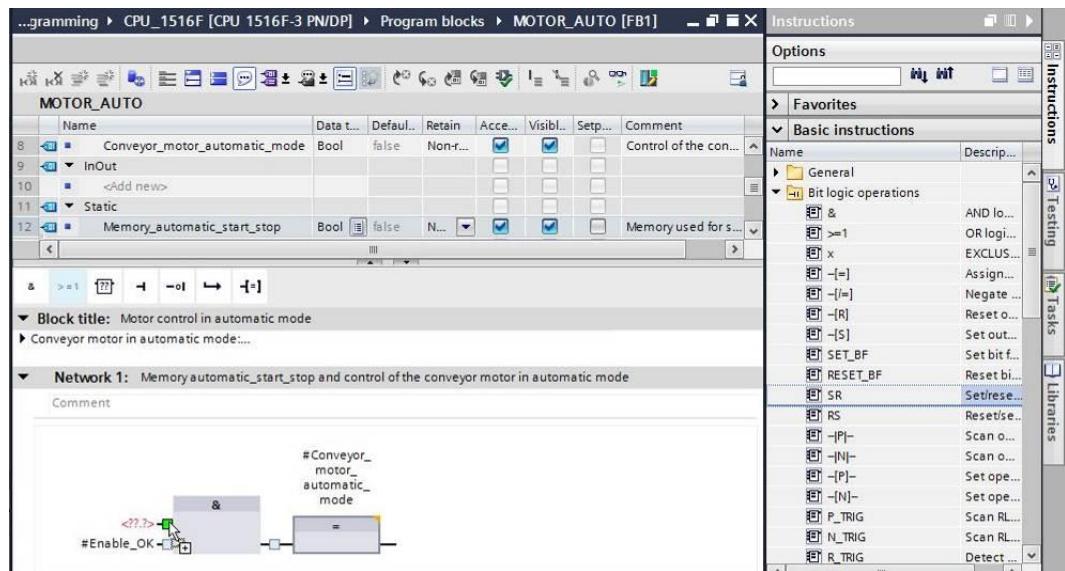


- Use drag-and-drop to move input parameter #Enable_OK onto the second input of the & logic operation <???.>. (→ Enable_OK)

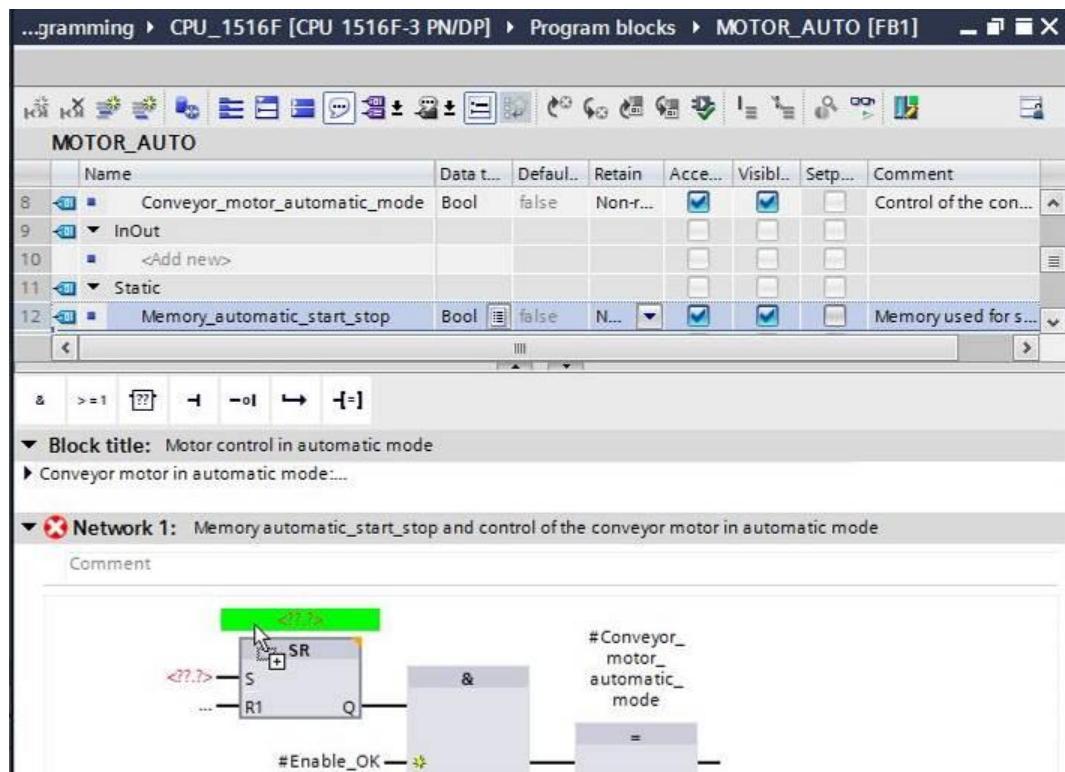


- Use drag-and-drop to move the Set/reset flip-flop function from the list of instructions under → Basic instructions → Bit logic operations onto the first input of the & operation .

(→ Instructions → Basic instructions → Bit logic operations → →)



- The SR flip-flop requires a memory tag. For this, use drag-and-drop to move static parameter #Memory_automatic_start_stop onto the <???.?> above the SR flip-flop. (→ Memory_automatic_start_stop)



- The #Memory_automatic_start_stop will be set with input tag #Start. Click twice on the S input of the SR flip-flop <???.?> and enter "Start" in the field that appears in order to see a list of available tags starting with "Start". Click the #Start tag and apply with → Enter.
(→ SR flip-flop → <???.?> → Start → #Start → Enter)

...gramming > CPU_1516F [CPU 1516F-3 PN/DP] > Program blocks > MOTOR_AUTO [FB1]

Name	Data t...	Defaul...	Retain	Acce...	Visibl...	Setp...	Comment
8 Conveyor_motor_automatic_mode	Bool	false	Non-r...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Control of the con...
9 InOut				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
10 <Add new>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
11 Static				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
12 Memory_automatic_start_stop	Bool	false	N...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Memory used for s...

& >=1 ?? -o! -o+ [=]

▼ Block title: Motor control in automatic mode
► Conveyor motor in automatic mode...

▼ Network 1: Memory automatic_start_stop and control of the conveyor motor in automatic mode

Comment

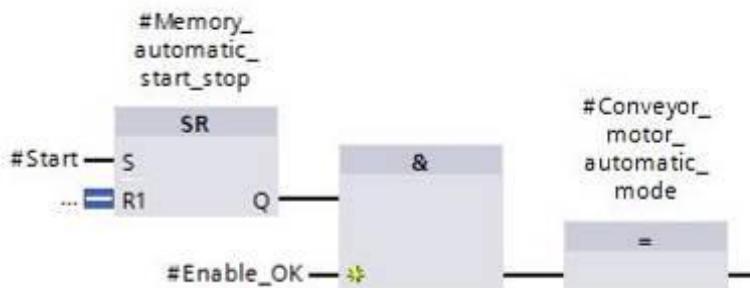
```

#Memory_
automatic_
start_stop
SR
start S
#Start Bool
#Conveyor_
motor_
automatic_
mode
Pushbutton aut...
&
#Enable_OK *

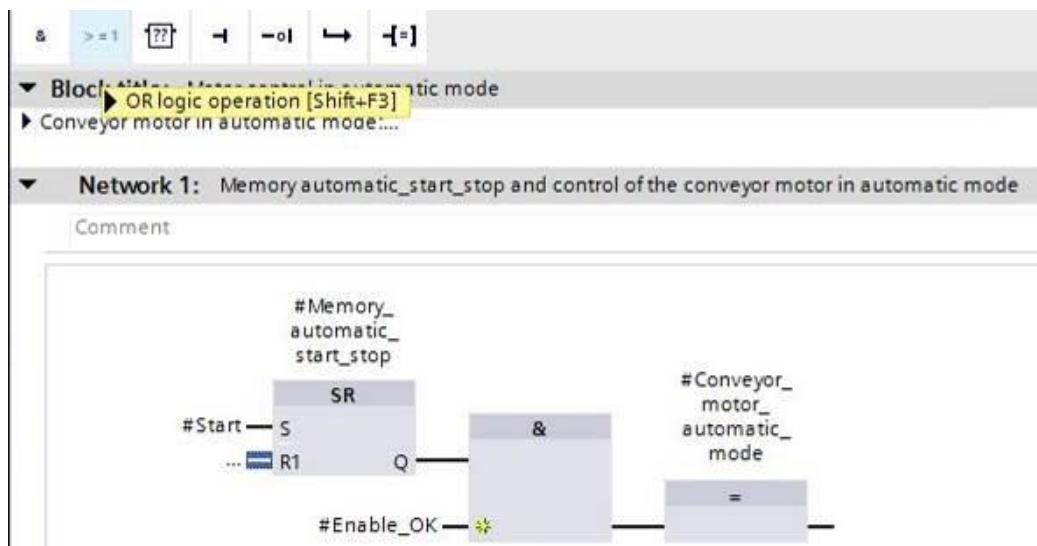
```

Note: When assigning tags in this way, there is a risk of a mix-up with the global tags from the tag table. The previously presented procedure using drag and drop from the interface description should therefore be used preferentially.

- Multiple conditions are to be able to stop the conveyor. An OR block is therefore needed at the R1 input of the SR flip-flop. First, click the R1 input of the SR flip-flop so that the input line has a blue background.



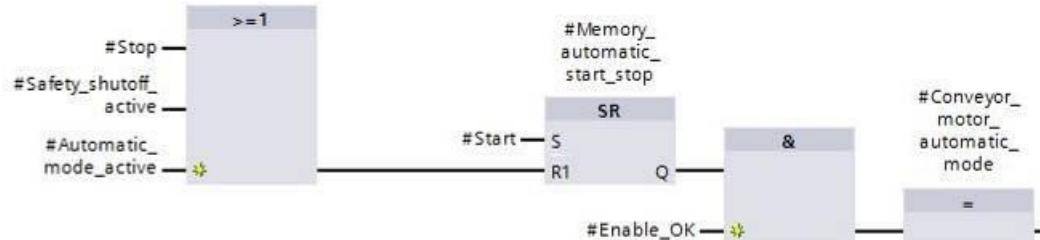
- Click the >=1 icon in your logic toolbar to insert an OR logic operation.



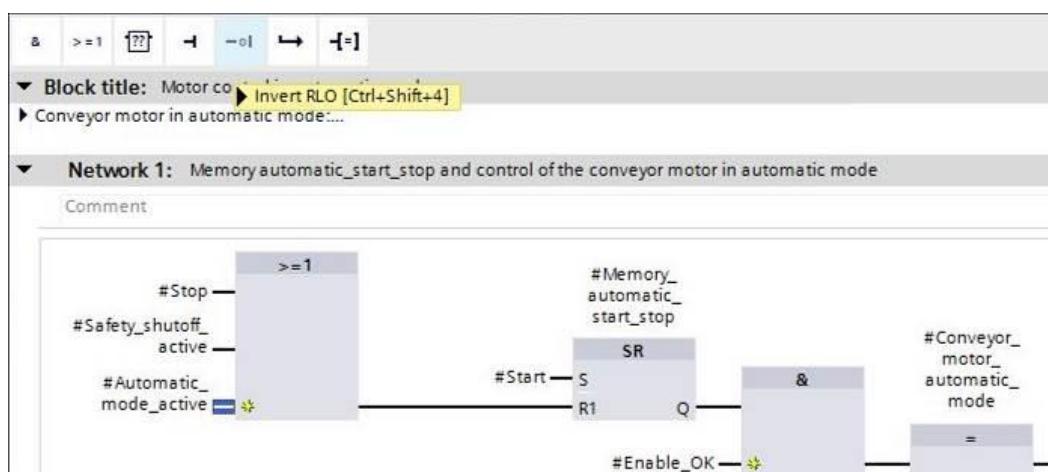
- The OR block has 2 inputs initially. In order to logically combine an additional input tag, click the yellow star of the OR block.



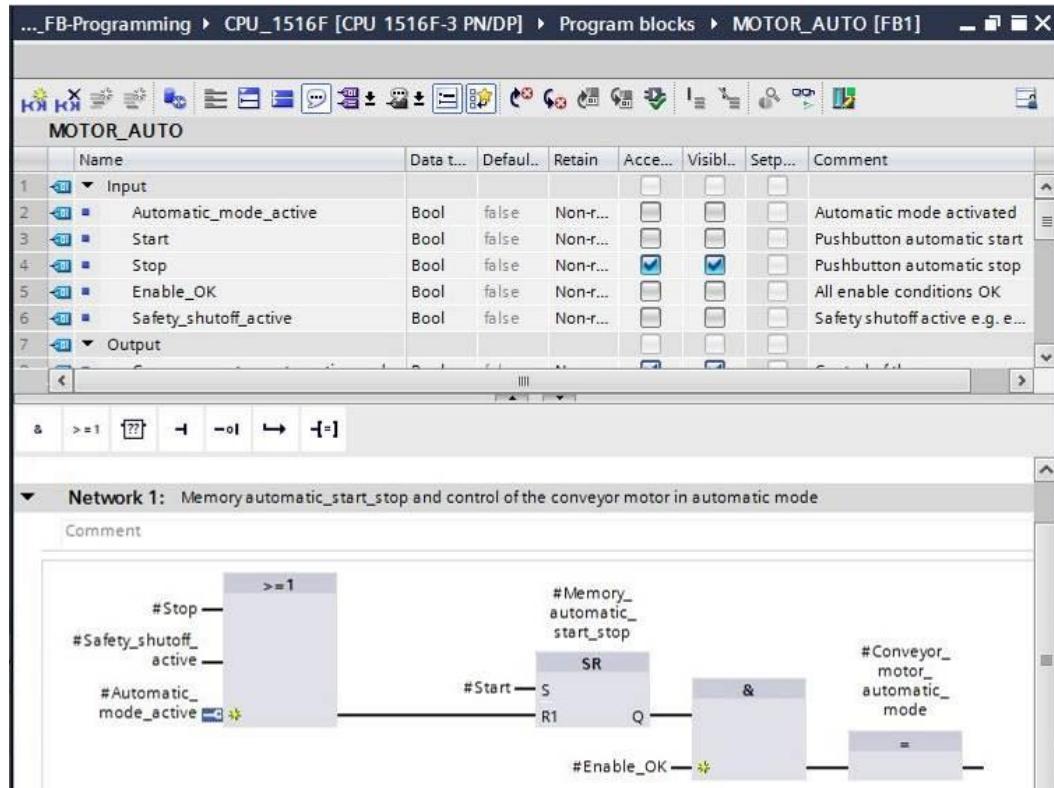
- Add input tags #Stop, #Safety_shutoff_active and #Automatic_mode_active to the 3 inputs of the OR block.



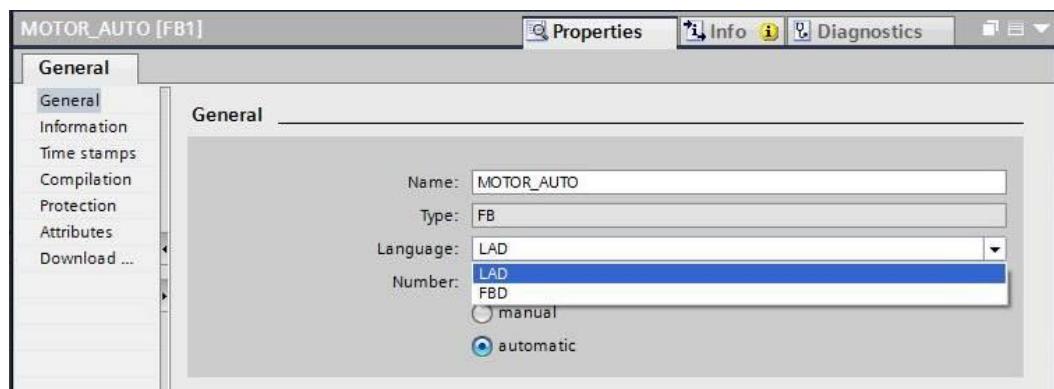
- Negate the input connected to parameter #Automatic_mode_active by selecting it and clicking .



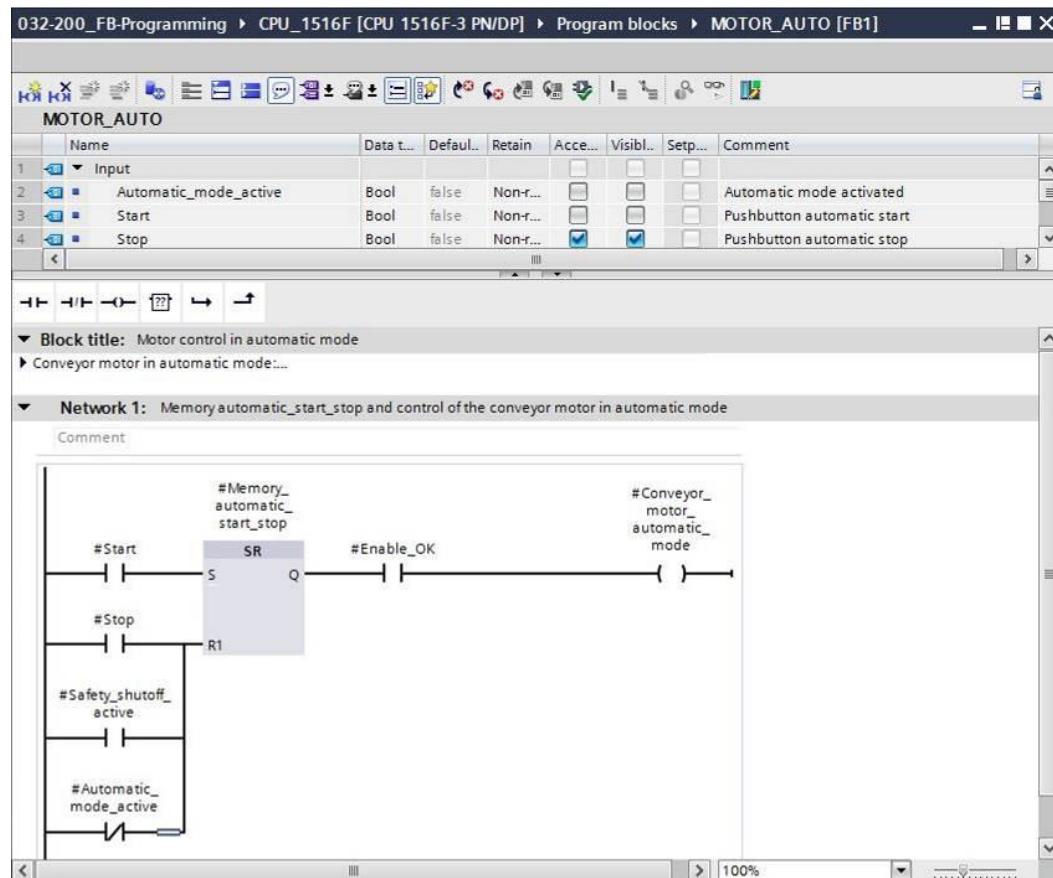
- Do not forget to click Save project. The finished function block "MOTOR_AUTO" [FB1] in FBD is shown below.



- Under "General" in the properties of the block, you can change the "Language" to LAD (Ladder Logic) (→Properties → General → Language: LAD)



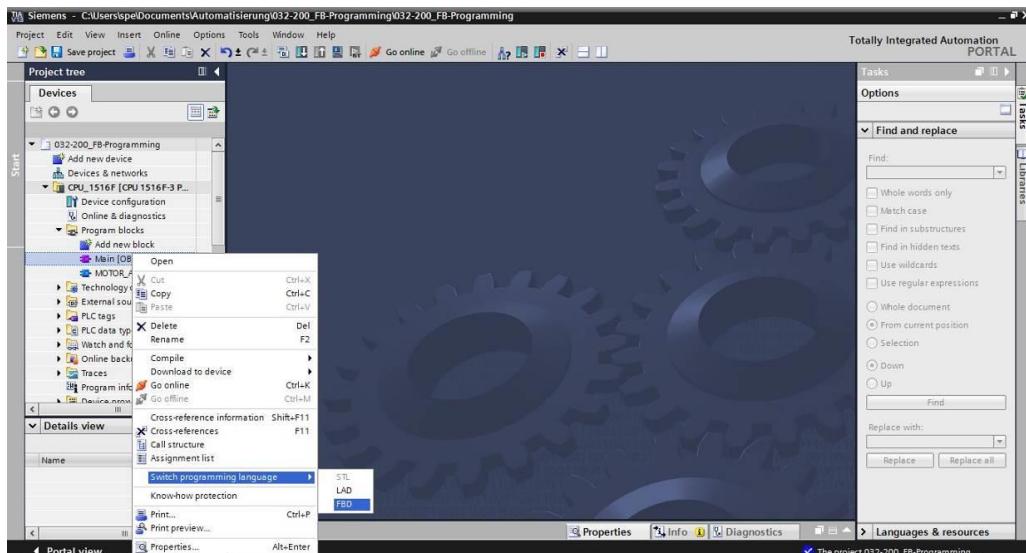
- The program has the following appearance in LAD.



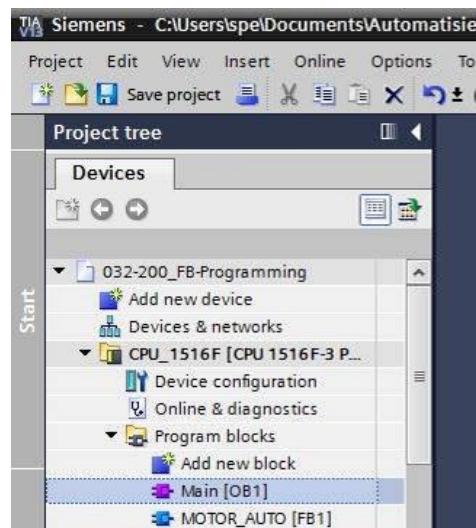
7.8 Program organization block OB1 – Control of the forward belt tracking in automatic mode

- Before programming organization block "Main [OB1]", we switch the programming language to FBD (Function Block Diagram). To do so, first click on "Main [OB1]" in the "Program blocks" folder.

(→ CPU_1516F[CPU 1516F-3 PN/DP → Program blocks → Main [OB1] → Switch programming language → FBD)

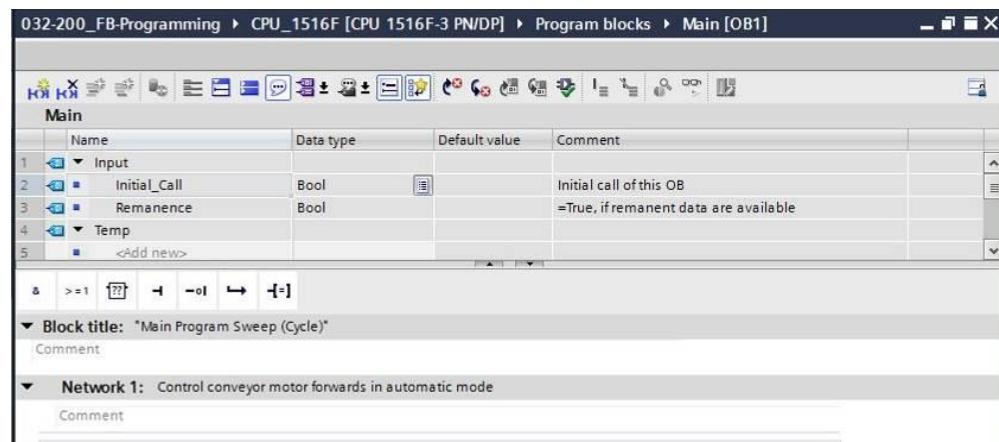


- Open the "Main [OB1]" organization block with a double-click.

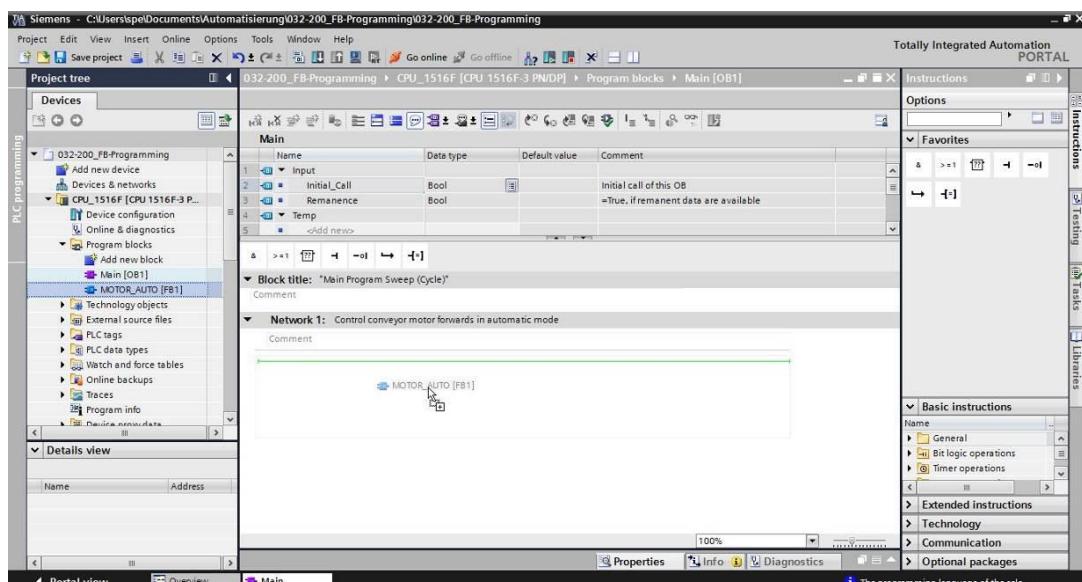


→ Assign Network 1 the name "Control conveyor tracking forward in automatic mode"

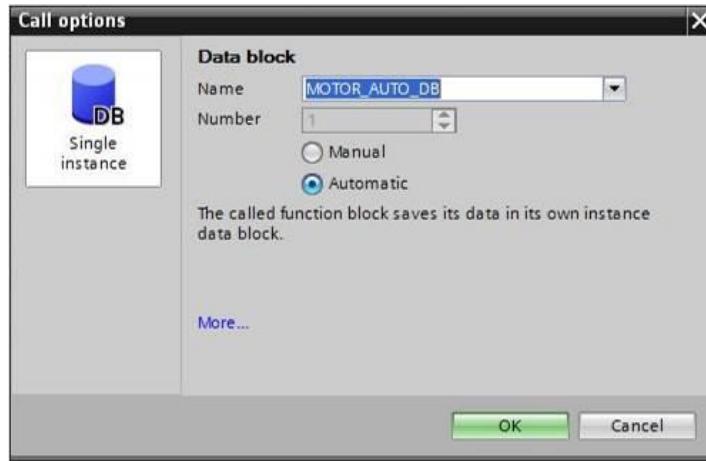
(→ Network 1:... → Control conveyor motor forwards in automatic mode)



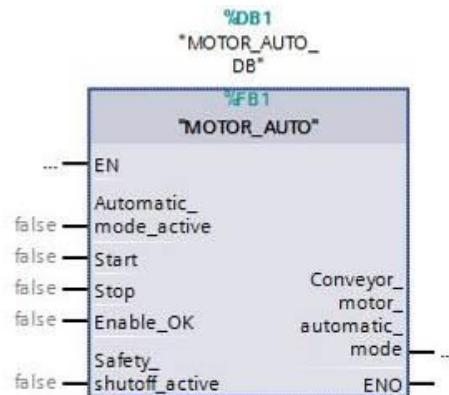
→ Use drag-and-drop to move your "MOTOR_AUTO [FB1]" function block onto the green line in Network 1.



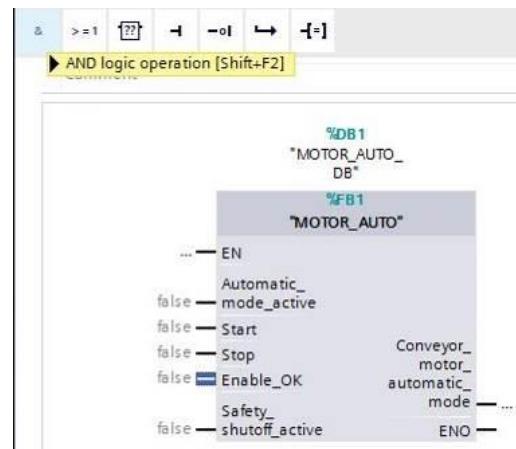
- The instance data block for this call of FB1 is created automatically. Assign a name and apply it with OK. (→ MOTOR_AUTO_DB1 → OK)



- A block with the interface you defined, the instance data block and connections EN and ENO are inserted in Network 1.

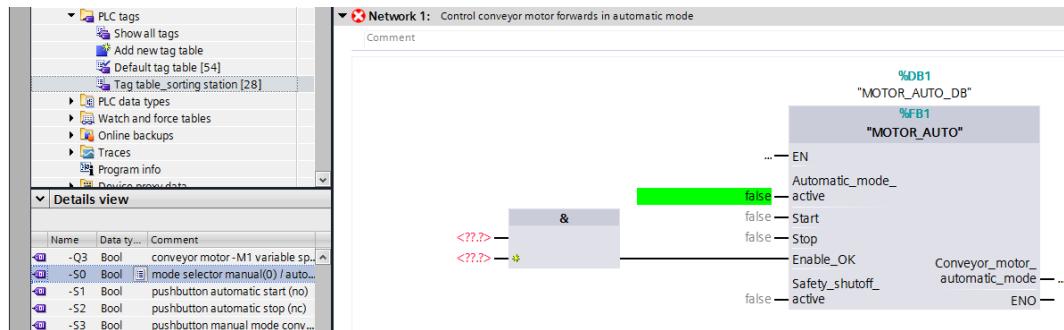


- To insert an AND before input parameter "Enable_OK", select this input and insert the AND by clicking the icon in your logic toolbar (→).

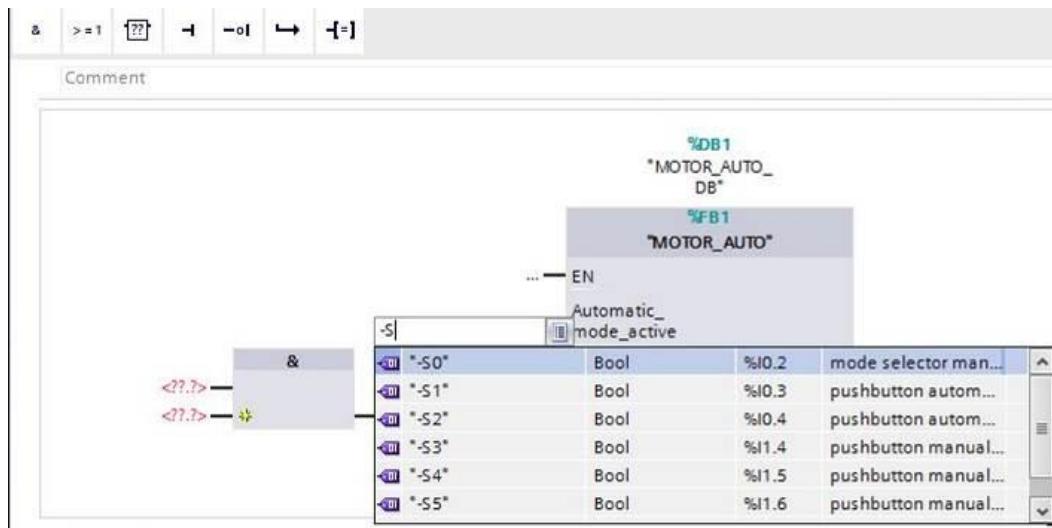


- To connect the block to the global tags from "Tag_table_sorting_station", we have two options:

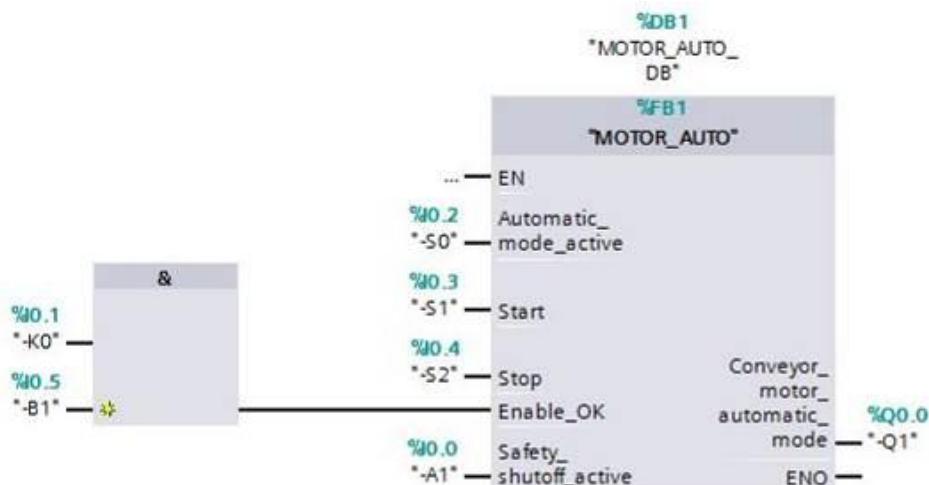
- Either select the "Tag_table_sorting_station" in the project tree and use drag-and-drop to move the desired global tag from the Details view to the interface of FC1 (→ Tag_table_sorting_station → Details view. → S0 → Automatic_mode_active)



- Or, enter the starting letters (e.g. "S") "-S" of the desired global tag for <???.?> and select the global input tag "-S0" (%I0.2) from the displayed list. (→ Automatic_mode_active → S → -S0)

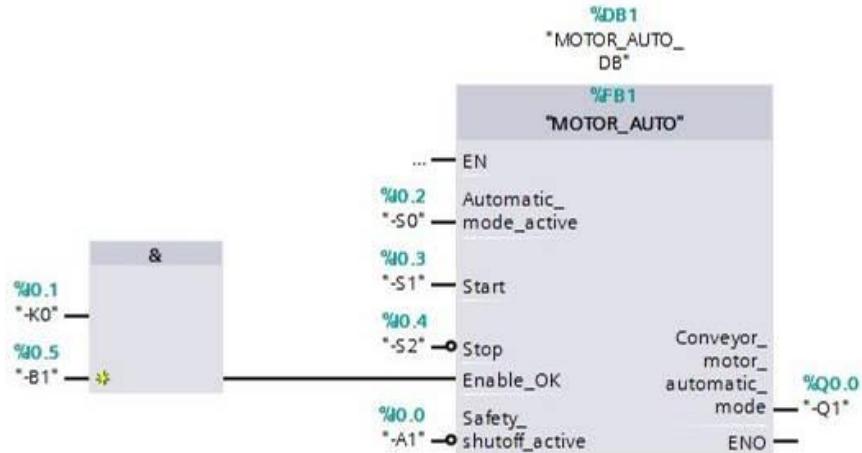


- Insert the other input tags "-S1", "-S2", "-K0", "-B1" and "-A1" and insert output tag "-Q1" (%Q0.0) at output "Conveyor_motor_automatic_mode".

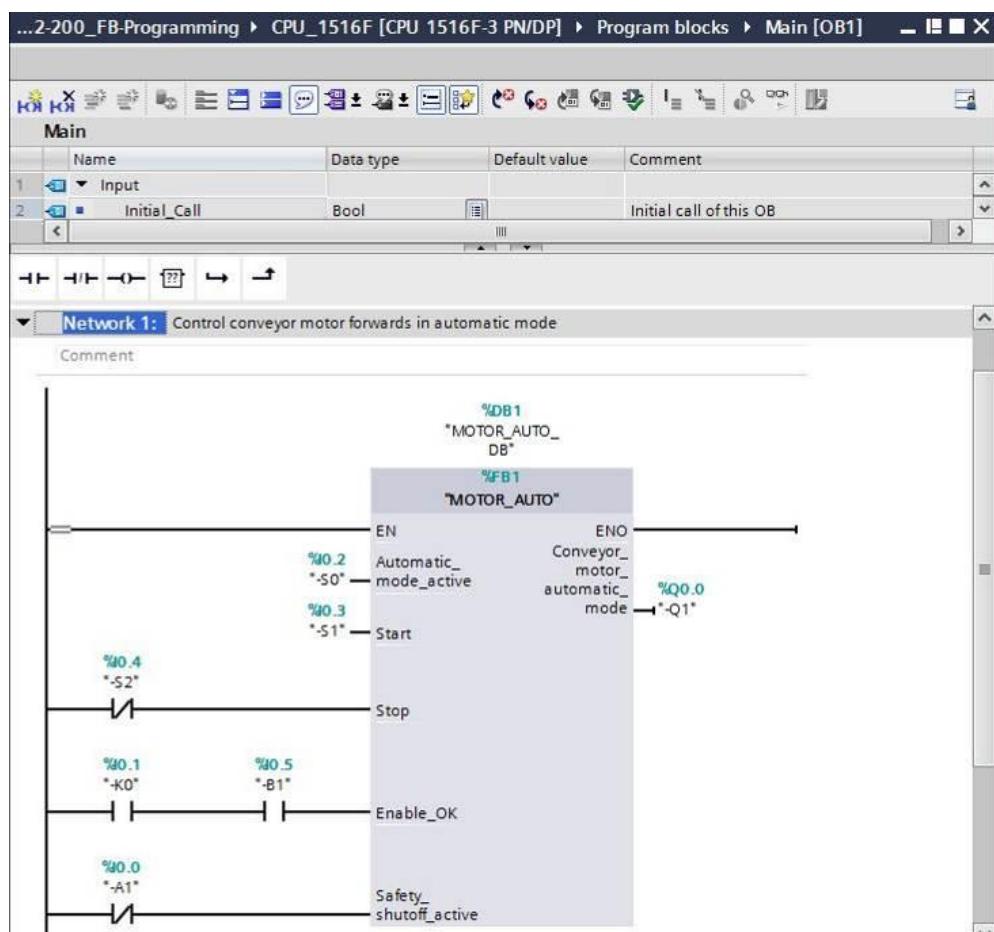


→ Negate the querying of input tags "-S2" and "-A1" by selecting them and clicking .

($\rightarrow -S2 \rightarrow \neg \square \rightarrow -A1 \rightarrow \neg \square)$

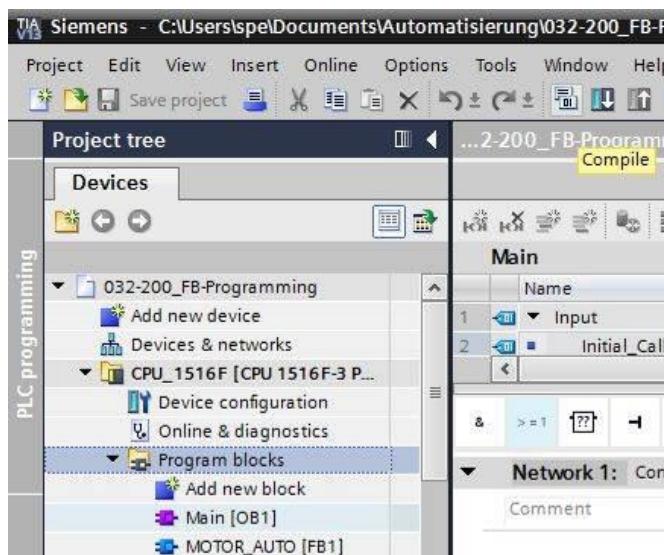


7.9 The result in the LAD (Ladder Logic) programming language has the following appearance.

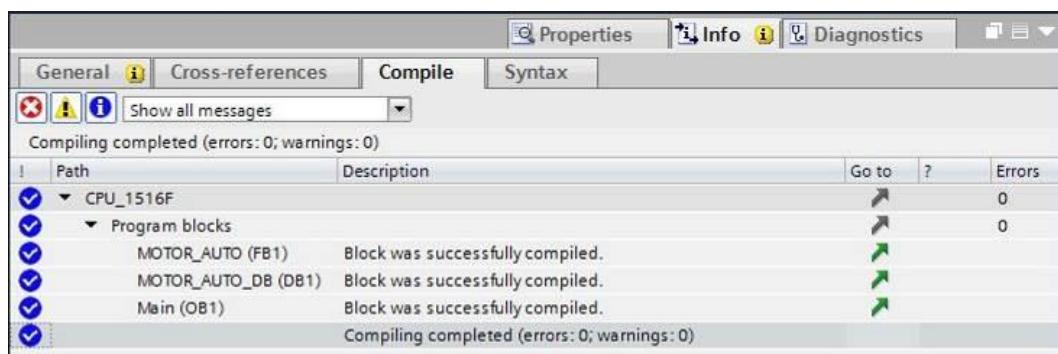


7.10 Save and compile the program

- To save your project, select the **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the icon for compiling in the menu
(→ → Program blocks → .

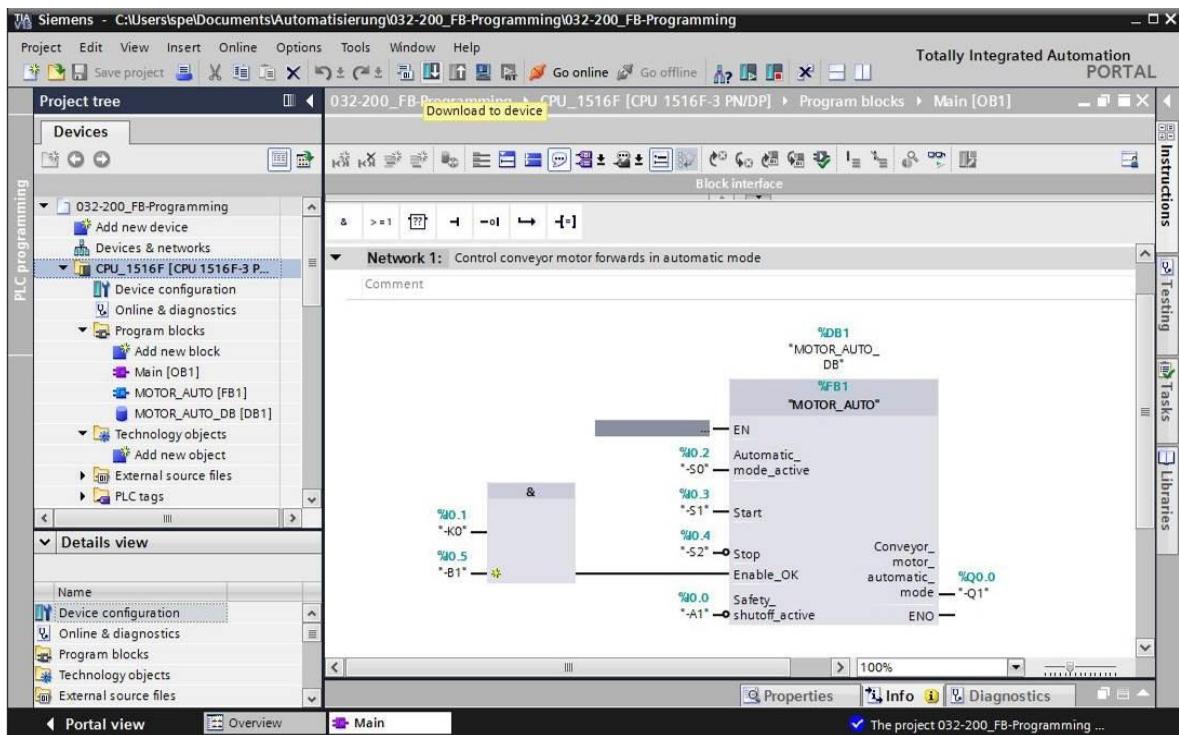


- The "Info", "Compile" area shows which blocks were successfully compiled.



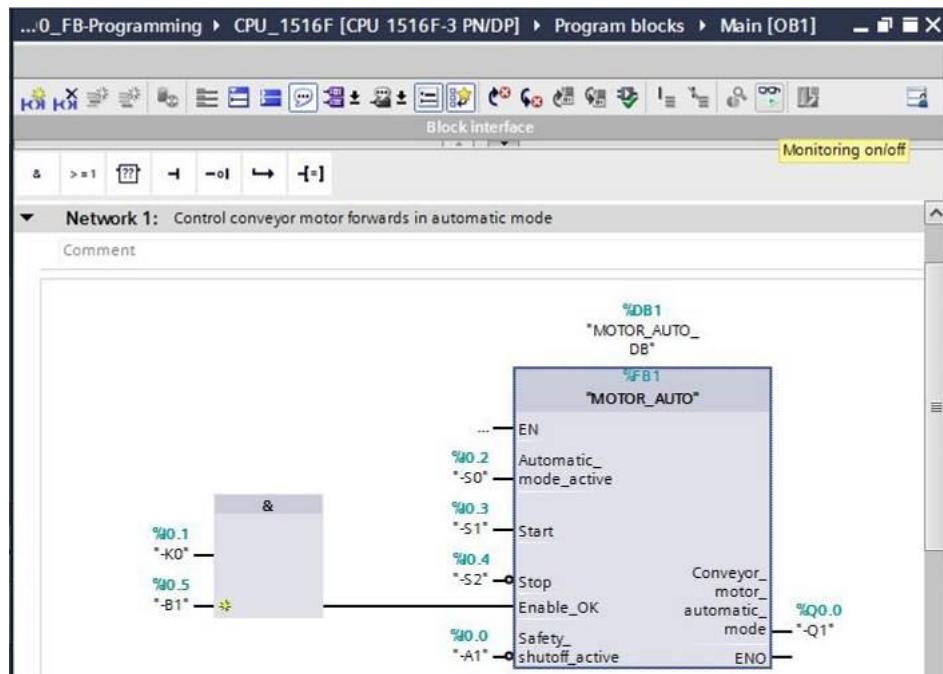
7.11 Download the program

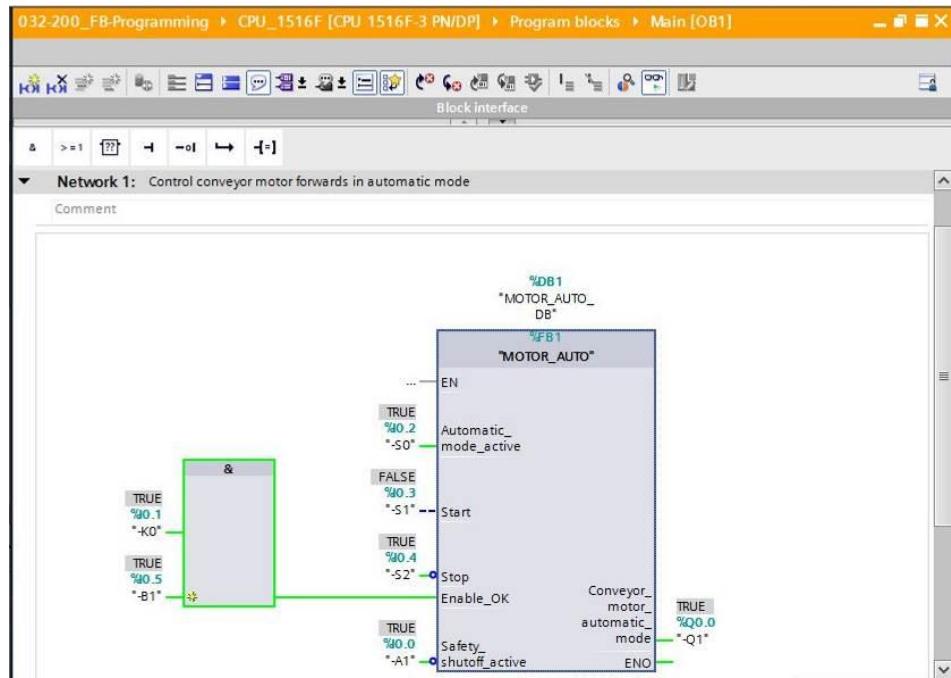
- After successful compilation, the complete controller with the created program, as previously described in the modules for hardware configuration, can be downloaded
(→ .



7.12 Monitor program blocks

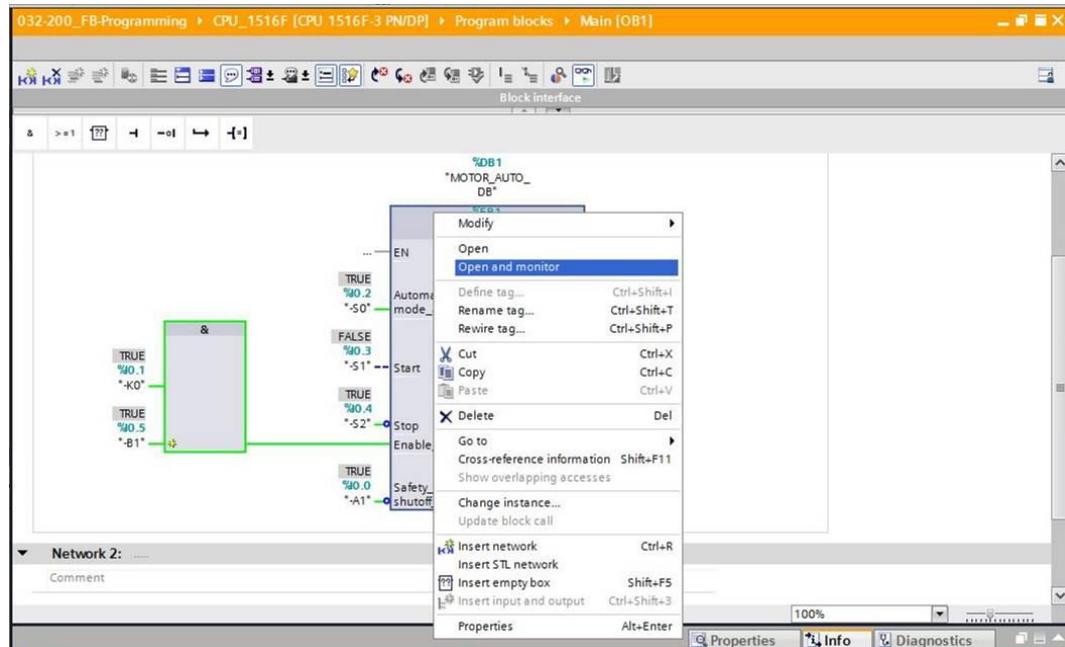
- The desired block must be open for monitoring the downloaded program. The monitoring can be activated/deactivated by clicking the icon. (→ Main [OB1] →

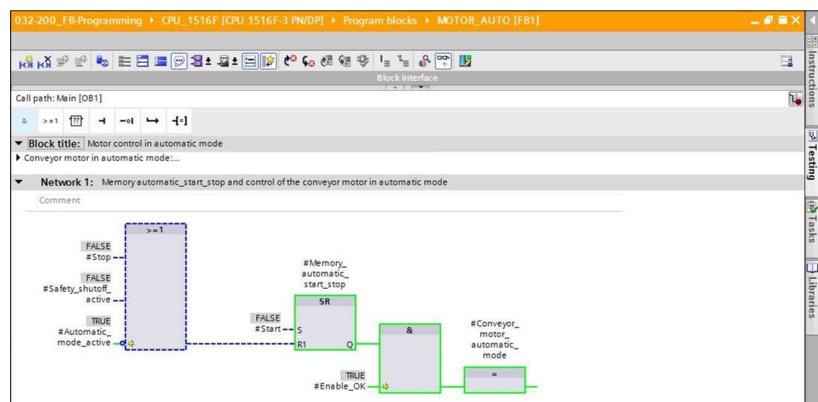




Note: The monitoring here is signal-related and controller-dependent. The signal states at the terminals are indicated with TRUE or FALSE.

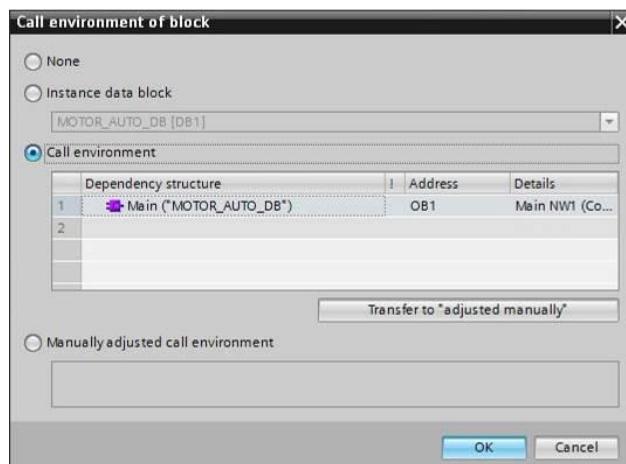
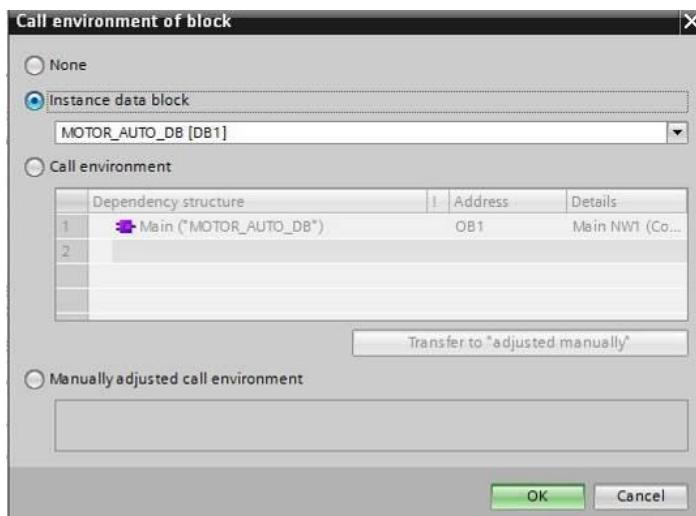
- The "MOTOR_AUTO" [FB1] function block called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking (→ "MOTOR_AUTO" [FB1] → Open and monitor).





Note: The monitoring here is function-related and controller-independent. The actuation of sensors and the station status are shown here with TRUE or FALSE.

- If a particular point of use of a "MOTOR_AUTO" [FB1] function block that is called multiple times is to be monitored, this can be performed using the icon. There are two alternatives available for specifying the call environment: using the call environment or the instance data block. (→ → Instance data block → MOTOR_AUTO_DB1 [DB1] → Call environment → Address: OB1 → Details: Main NW1 → OK).



8 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) then conveyor motor forwards fixed speed (-Q1 = 1) switches on and stays on.	
4	Briefly press the automatic stop pushbutton (-S2 = 0) → Q1 = 0	
5	Activate EMERGENCY OFF (-A1 = 0) → Q1 = 0	
6	Manual mode (-S0 = 0) → Q1 = 0	
7	Switch off station (-K0 = 0) → Q1 = 0	
8	Cylinder not retracted (-B1 = 0) → Q1 = 0	
9	Project successfully archived	



Training Curriculum

TIA Portal Module 006
IEC Timers and IEC Counters
Multi-instances for SIMATIC S7-1500

Table of contents

1	Goal.....	213
2	Prerequisite.....	213
3	Required hardware and software.....	213
4	Theory	214
4.1	Instances and multi-instances in SIMATIC S7-1500	214
4.1.1	Instance data blocks/single instances.....	214
4.1.2	Multi-instances.....	215
5	Planning.....	216
5.1	Automatic mode - Conveyor motor with time function.....	216
6	Structured step-by-step instructions.....	216
6.1	Retrieve an existing project.....	216
6.2	Addition of an IEC timer TP to FB1 "MOTOR_AUTO"	218
6.3	Update the block call in the organization block	223
6.4	Save and compile the program	224
6.5	Download the program.....	225
6.6	Monitor program blocks	225
7	Checklist	227

IEC TIMERS AND IEC COUNTERS MULTI-INSTANCES FOR SIMATIC S7-1500

1 Goal

In this chapter, you will become acquainted with the use of single instances and multi-instances for programming of the SIMATIC S7-1500 with the TIA Portal programming tool.

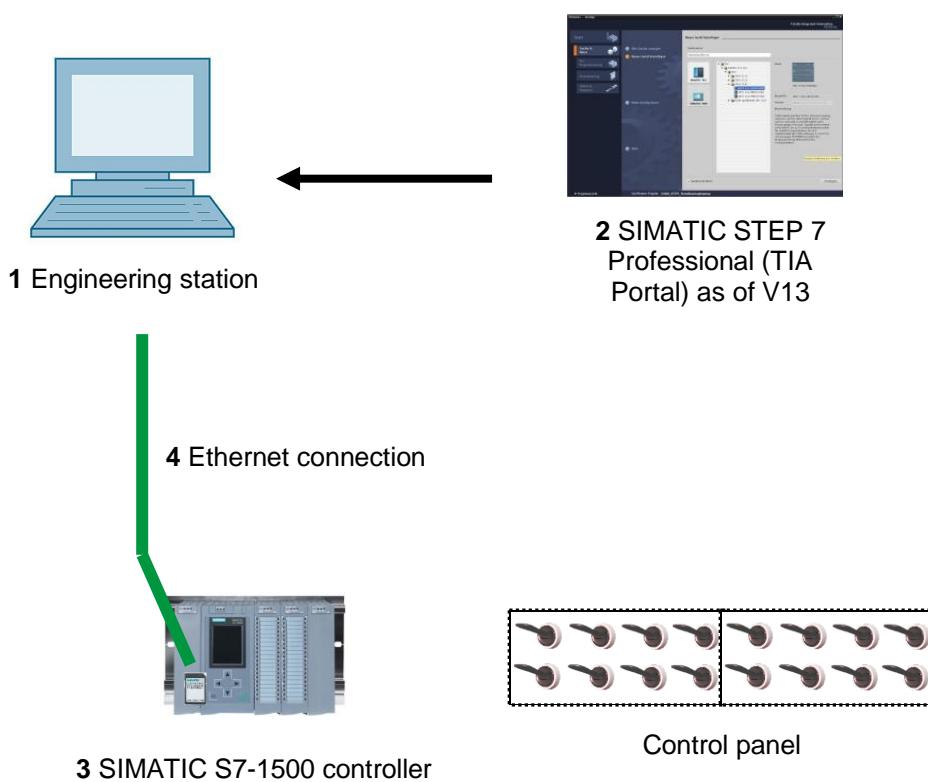
The module explains the various types of instance data blocks and shows step-by-step how to add IEC timers and IEC counters to a program block.

2 Prerequisite

This chapter builds on the FB programming with the SIMATIC S7 CPU1516F-3 PN/DP. You can use the following project for this chapter.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V13
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 Instances and multi-instances in SIMATIC S7-1500

The call of a function block is referred to as an **instance**. An **instance** is assigned to every call of a function block and serves as a data memory. It stores the actual parameters and the static data of the function block.

The tags declared in the function block determine the structure of the instance data block.

Use of single instances and multi-instances

You can assign instances as follows:

Call as a **single instance**:

- A separate instance data block for each instance of a function block

Call as a **multi-instance**:

- One instance data block for several instances of one or more function blocks

4.1.1 Instance data blocks/single instances

The call of a function block that is assigned its own instance data block is called a **single instance**.

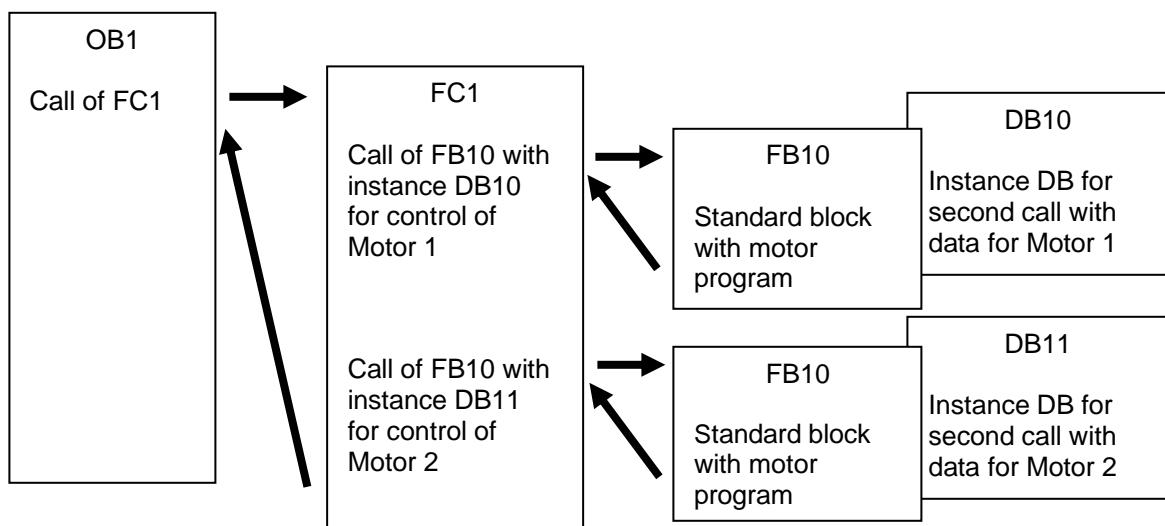
If the function block was created according to the rules for library-compatible standard blocks, it can also be called multiple times.

However, you must assign another instance data block for each call as a single instance.

Example of single instances:

The following figure shows the control of two motors using one function block FB10 and two different data blocks:

The different data for the individual motors, such as speed, acceleration time and total operating time, are saved in the instance data blocks DB10 and DB11.



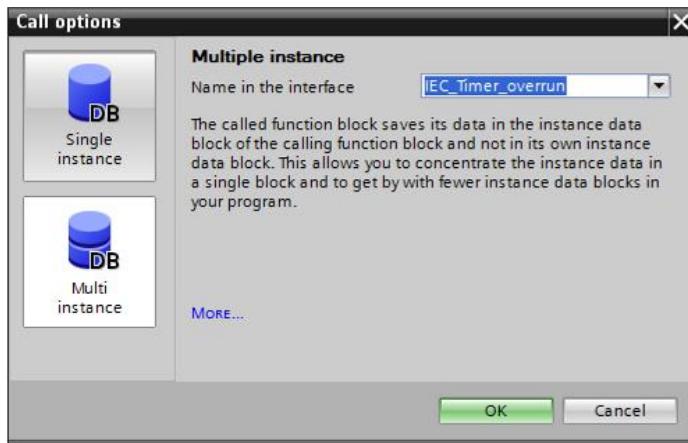
Note: Some commands, such as timers and counters, react like function blocks. When these are called, they also require an assigned memory area, e.g. in the form of an instance data block.

4.1.2 Multi-instances

You may want to limit the number of data blocks used for instances or this may be necessary due to lack of memory in the utilized CPU.

If other function blocks, timers, counters, etc. that already exist will be called in a function block in your user program, you can call these other function blocks without separate (i.e., additional) instance DBs.

Simply select ‘Multi-instance’ for the call options:



Notes: Multi-instances enable a called function block to store its data in the instance data block of the calling function block.

In this case, the calling block must always be a function block.

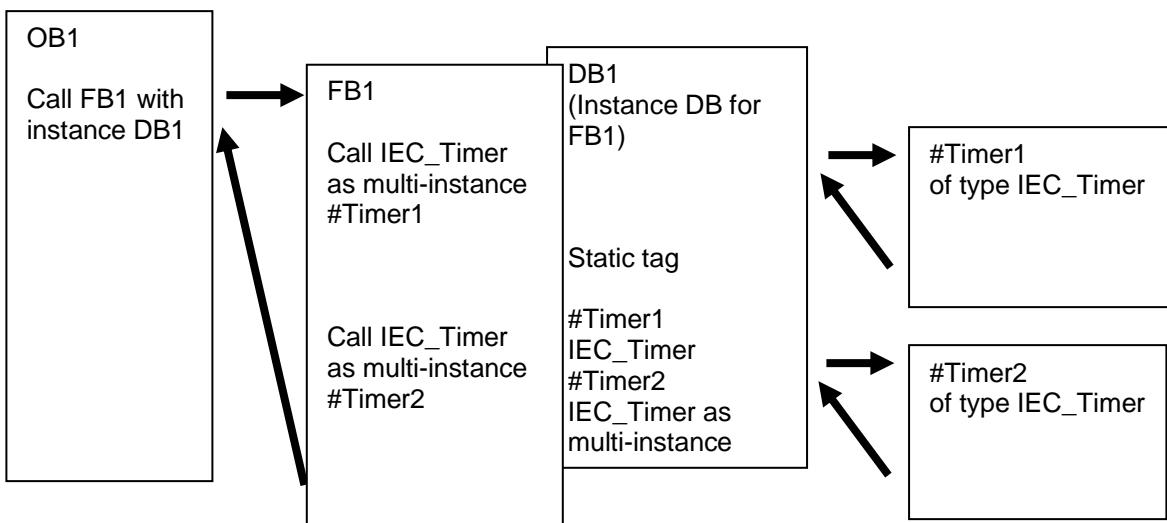
This allows you to concentrate the instance data in one instance data block and thus make better use of the number of DBs available.

Incidentally, this is always required when the calling block is to remain available for reuse as a standard block.

Example of multi-instances:

The following figure shows two calls of an IEC_Timer of type TP (pulse) within a function block.

The different data for the two counters is stored as different **multi-instances** in the instance data block DB1 of the calling function block FB1.



5 Planning

The IEC timer is programmed as an addition to the MOTOR_AUTO [FB1] function block from the "032-200_FBProgramming.zap13" project. This project must be retrieved in order to add the IEC timer TP (latching pulse). A multi-instance will be created as a memory for the timer.

5.1 Automatic mode - Conveyor motor with time function

The Memory_automatic_start_stop is latched with Start but only if the reset conditions are not present.

The Memory_automatic_start_stop is reset if Stop is present or safety shutoff is active or automatic mode is not activated (manual mode).

The Conveyor_motor_automatic_mode output is activated when Memory_automatic_start_stop is set, the enable conditions are met and Memory_conveyor_start_stop is set.

To save energy, the conveyor should only run when a part is present.

For this reason, the Memory_conveyor_start_stop is set when Sensor_chute_occupied signals a part and reset when Sensor_end_of_conveyor produces a negative edge or safety shutoff is active or automatic mode is not activated (manual mode).

Addition of time function:

Because the Sensor_end_of_conveyor is not able to be mounted directly at the end of the conveyor, the Sensor_end_of_conveyor signal must be stretched.

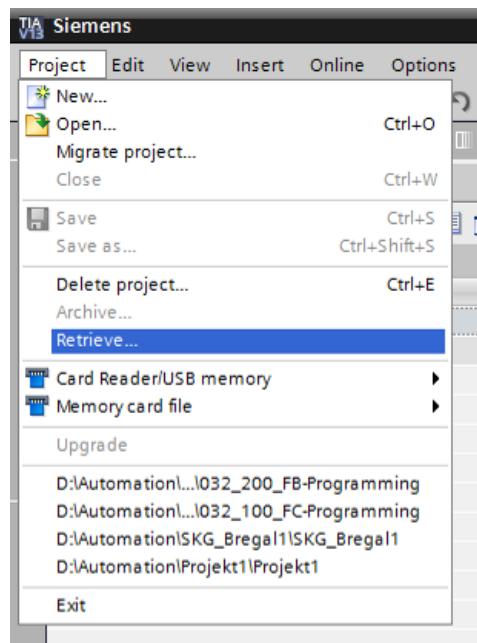
To achieve this, a latching pulse will be inserted between Sensor_end_of_conveyor and the negative edge detection.

6 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

6.1 Retrieve an existing project

Before we can expand the "MOTOR_AUTO [FB1]" function block, we must retrieve the "032-200_FBProgramming.zap13" project from chapter "SCE_EN_032-200 FBProgramming". To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open. (→ Project → Retrieve → Select a .zap archive → Open)

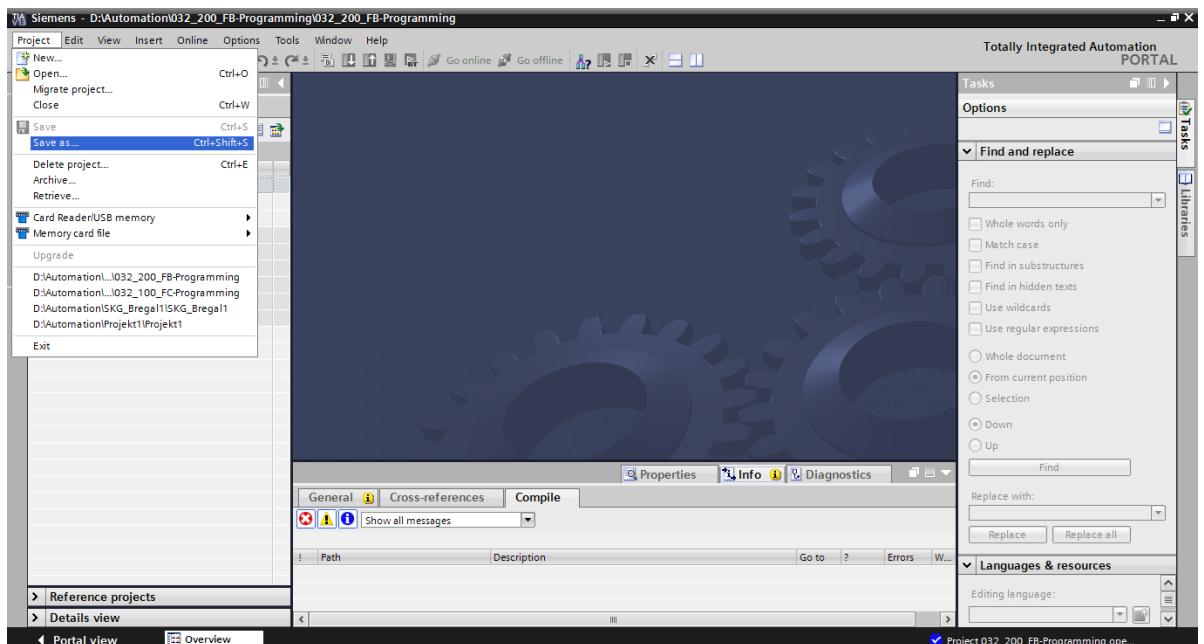


The next step is to select the target directory where the retrieved project will be stored.

Confirm your selection with "OK". (→ Target directory → OK)

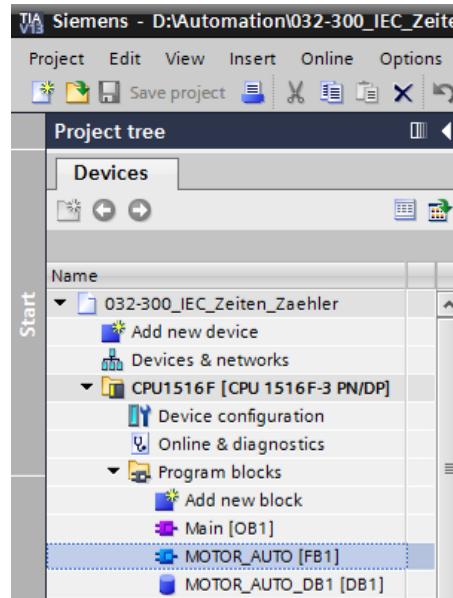
Save the opened project under the name 032-300_IEC_Timers_Counters.

→ Project → Save as ... → 032-300-IEC_Timers_Counters → Save)



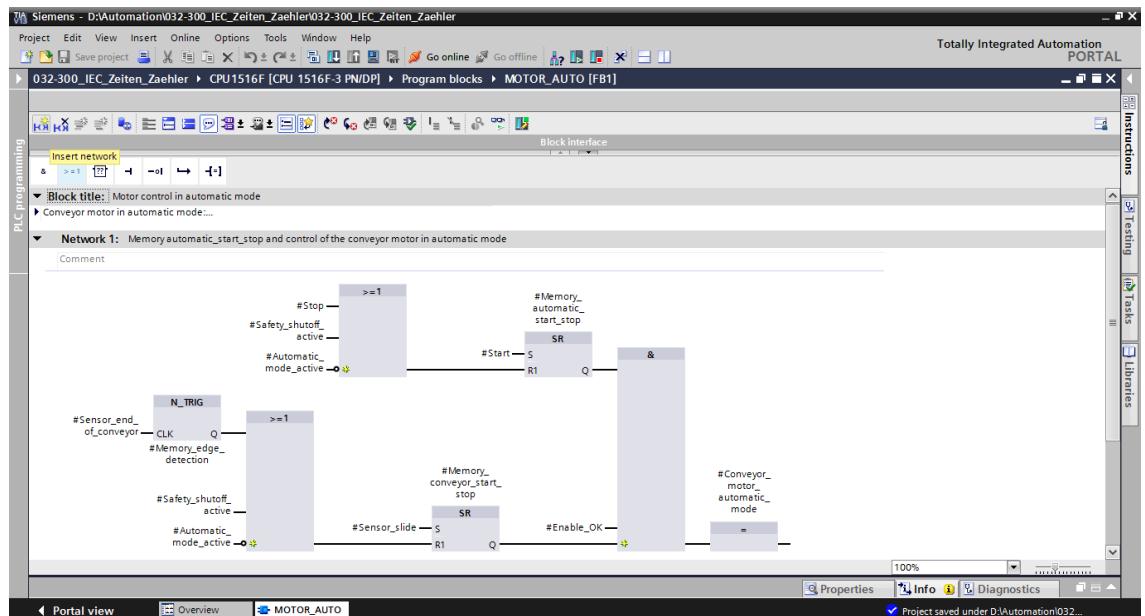
6.2 Addition of an IEC timer TP to FB1 "MOTOR_AUTO"

First, open the "MOTOR_AUTO [FB1]" function block with a double-click.



Insert another network at the beginning of the "MOTOR_AUTO [FB1]" function block by

selecting the → "block title" and then clicking the → icon for "Insert network".



Add helpful information to the block comment and the network title of "Network 1:".

Block title: Motor control in automatic mode

Conveyor motor in automatic mode:

The bit Memory_automatic_start_stop is set with the input Start, but only if the reset conditions are not fulfilled. The bit Memory_automatic_start_stop is reset with the input Stop or if the safety shutoff is activated or if the automatic mode is not activated (manual mode). If Memory_automatic_start_stop is set, the enable conditions are granted and Memory_conveyor_start_stop is set the output Conveyor_motor_automatic_mode is activated. For reasons of energy efficiency the conveyor motor should only run if a part is present. Therefore Memory_conveyor_start_stop is set if there is a part detected in front of Sensor_slide and reset with a negative edge at Sensor_end_of_conveyor or if the safety shutoff is activated or if the automatic mode is not activated (manual mode).

As the Sensor_end_of_conveyor couldn't be assembled directly at the end of the conveyor we need an additional time until we can stop the motor. For that purpose we use an extended pulse timer in between the Sensor_end_of_conveyor and the negative edge detection.

Network 1: Overrun time end of conveyor pulse 2 seconds

On the right side of your programming window, you will see the timer functions in the list of instructions. Under → Basic instructions → Timer operations, find function TP (Generate pulse) and use a drag-and-drop operation to move it to Network 1 (green line appears, mouse pointer with + symbol).

(→ Instructions → Basic instructions → Timer operations → TP)

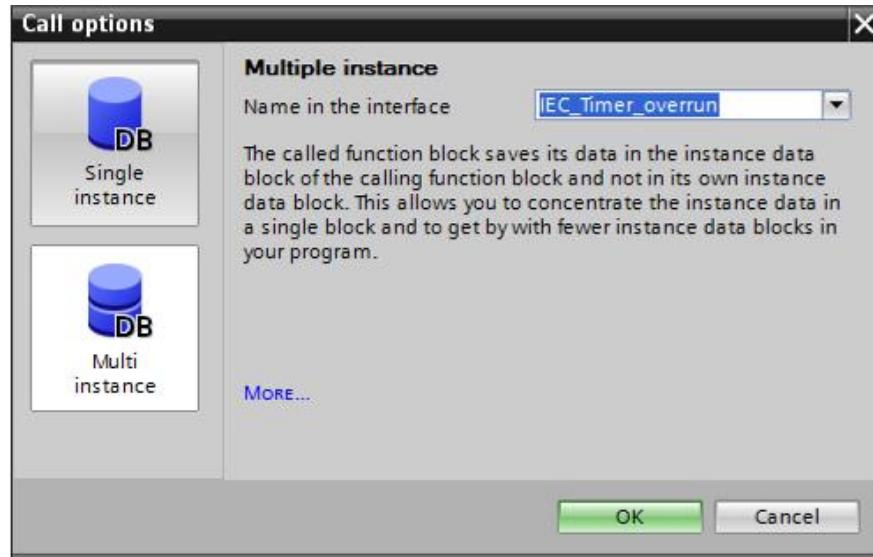
Name	Data type	Default value	Retain	Accessible f...	Visible in ...	Setpoint	C...
1	Input						
2	Automatic_mode_act...	Bool	false	Non-ret...			A...
3	Start	Bool	false	Non-retain			Pu...
4	Stop	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Pu...
5	Enable_OK	Bool	false	Non-retain			All...
6	Safety_shutoff_active	Bool	false	Non-retain			Sa...
7	Sensor_slide	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Se...
8	Sensor_end_of_conve...	Bool	false	Non-retain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Se...

Block title: Motor control in automatic mode

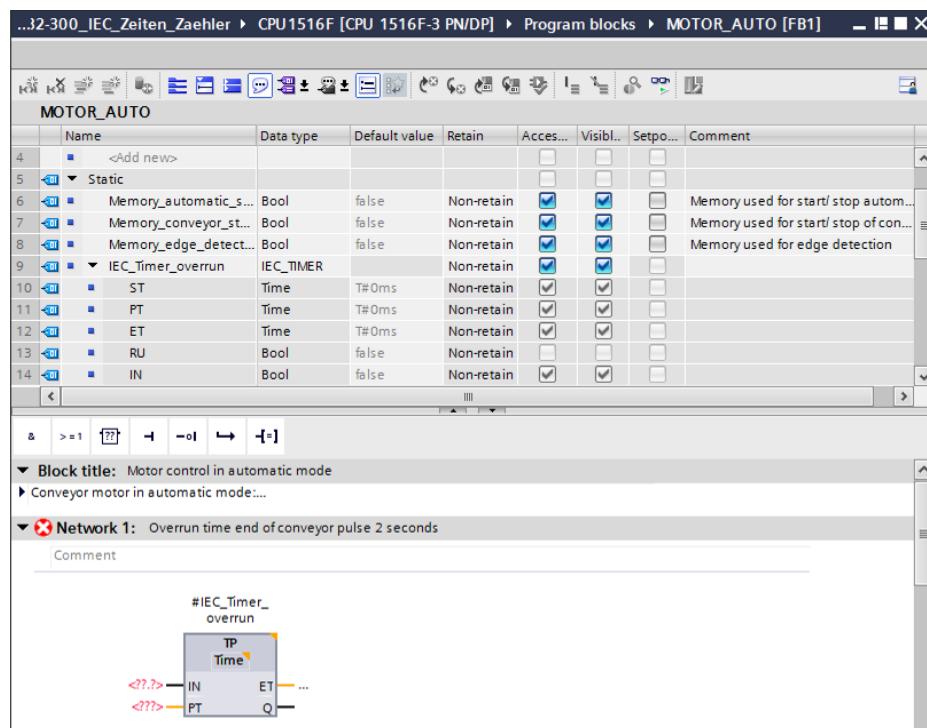
Conveyor motor in automatic mode...

Network 1: Overrun time end of conveyor pulse 2 seconds

The timer function requires a memory. Here, this memory is made available within the instance data block by the function block without the creation of a new instance data block. Select the →"Multi-instance" option for this. Enter a name for the multi-instance and confirm with → "OK". (→ Multi-instance → IEC_Timer_overrun → OK)

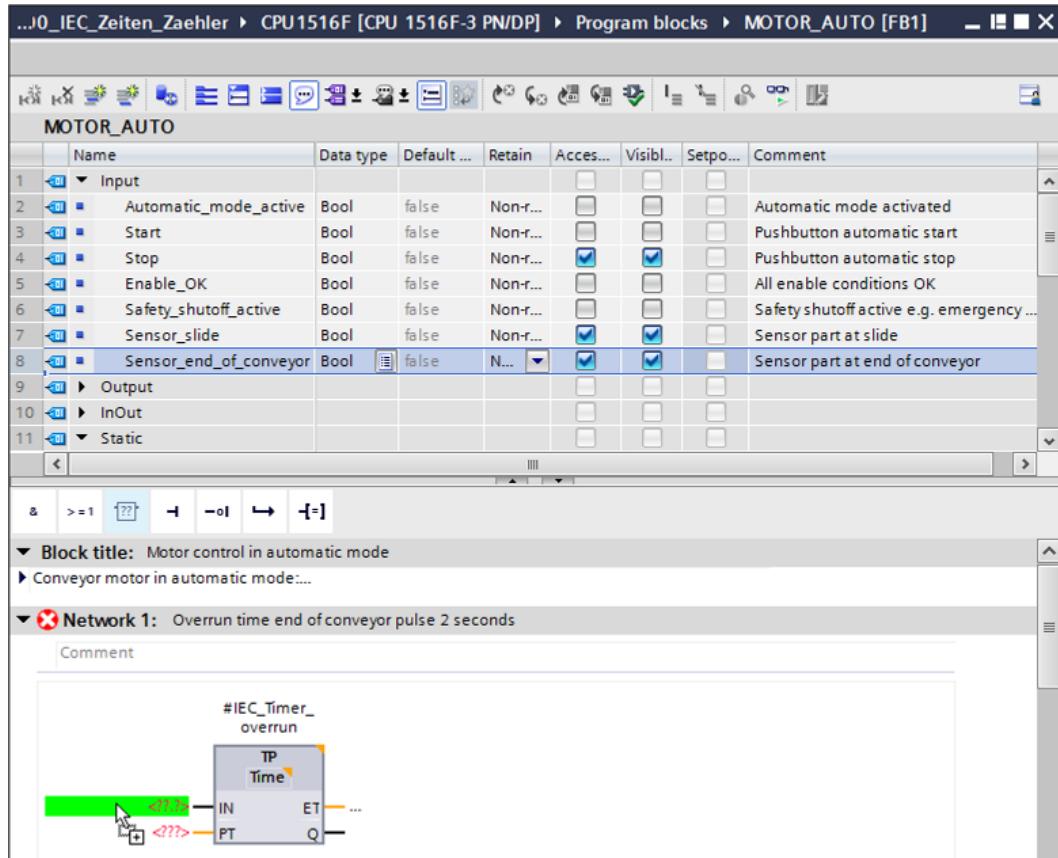


As a result, a tag structure of "Static" type suitable for TP Timer will be created in the interface description.

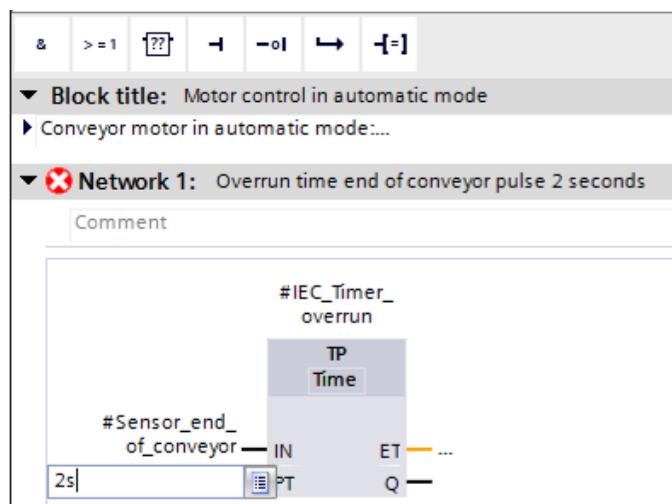


Note: A multi-instance can only be used for programming within a function block because static tags are only available there.

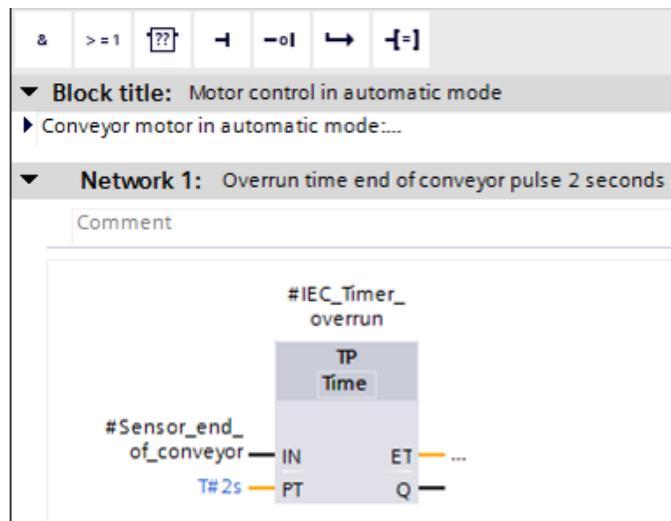
Use drag-and-drop to move input parameter #Sensor_end_of_conveyor to <??.> in front of parameter "IN" of TP Timer so that this will be started at a positive edge at input #Sensor_end_of_conveyor. The best way to select a parameter in the interface description is by "grabbing" it at the blue symbol. (→ Sensor_end_of_conveyor)



Enter the required pulse duration of 2 seconds in front of parameter "PT" (→ 2s)

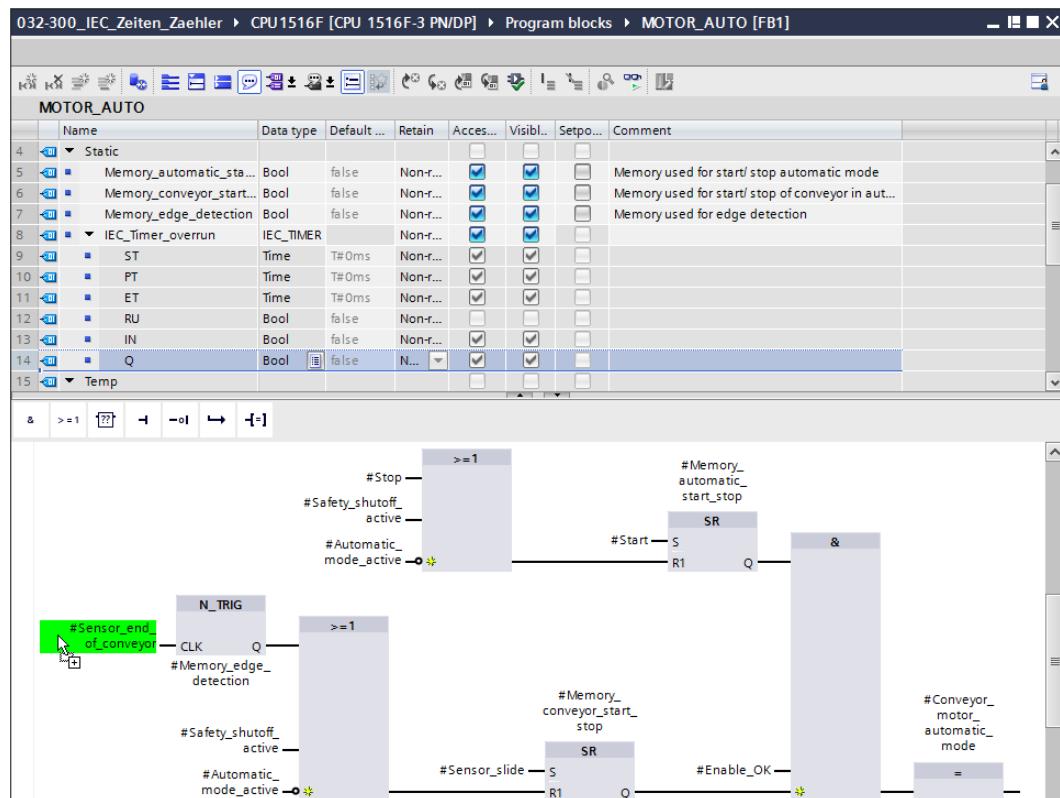


The entry of 2s is converted automatically to the IEC-Time format suitable for the IEC timer and is shown as constant "T#2s".

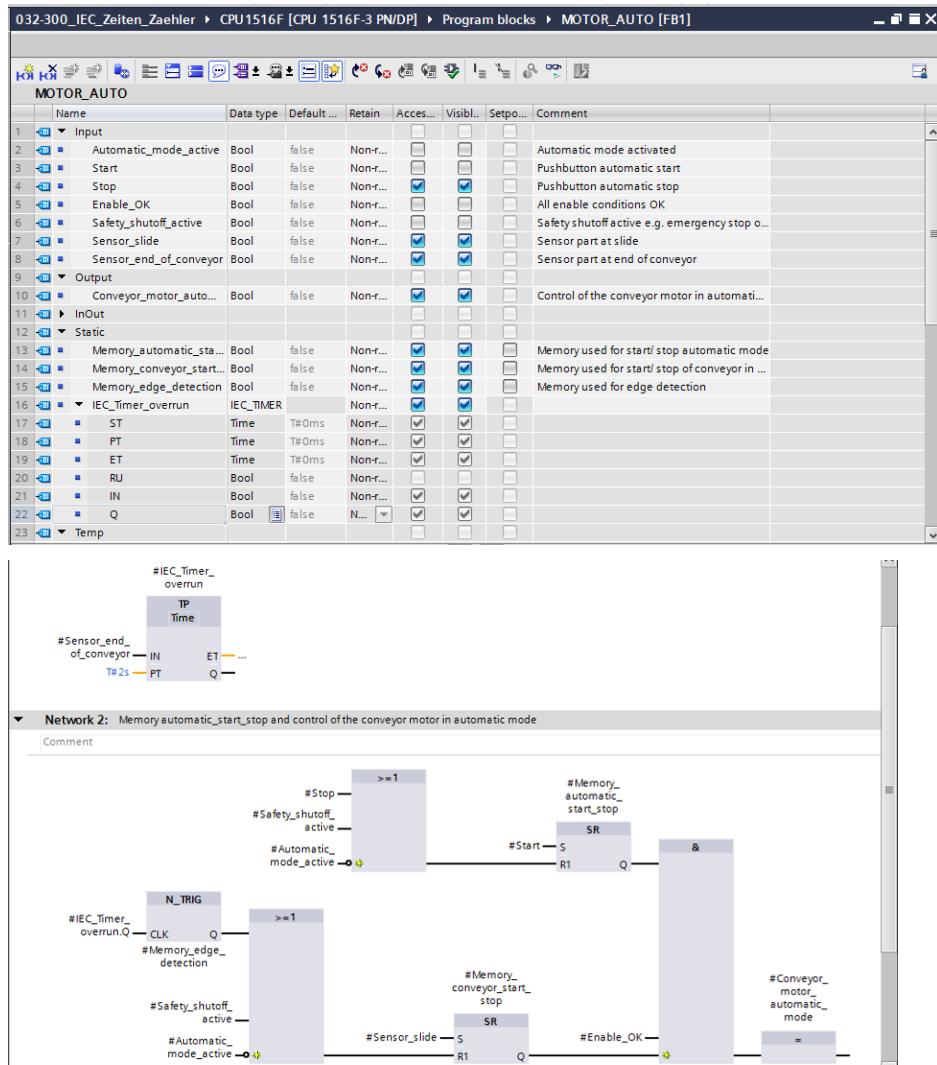


Now move output "Q" from tag structure "IEC_Timer_overrun" onto input "CLK" of negative edge "N_TRIG" in Network 2. This will replace the #Sensor_end_of_conveyor input tag previously entered there and the conveyor will be stopped by a negative edge of the IEC_Timer_overrun pulse.

(→ Network 2 → IEC_Timer_overrun → Q → #Sensor_end_of_conveyor)

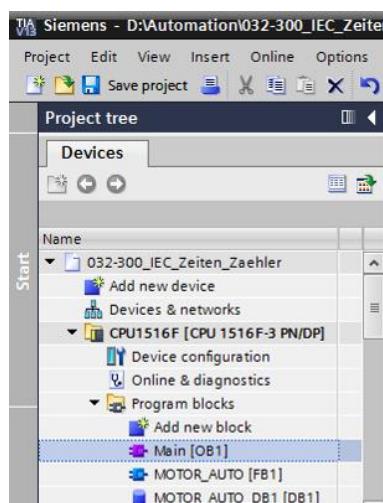


Do not forget to click Save project. The finished function block "MOTOR_AUTO" [FB1] with the timer is shown in FBD below.

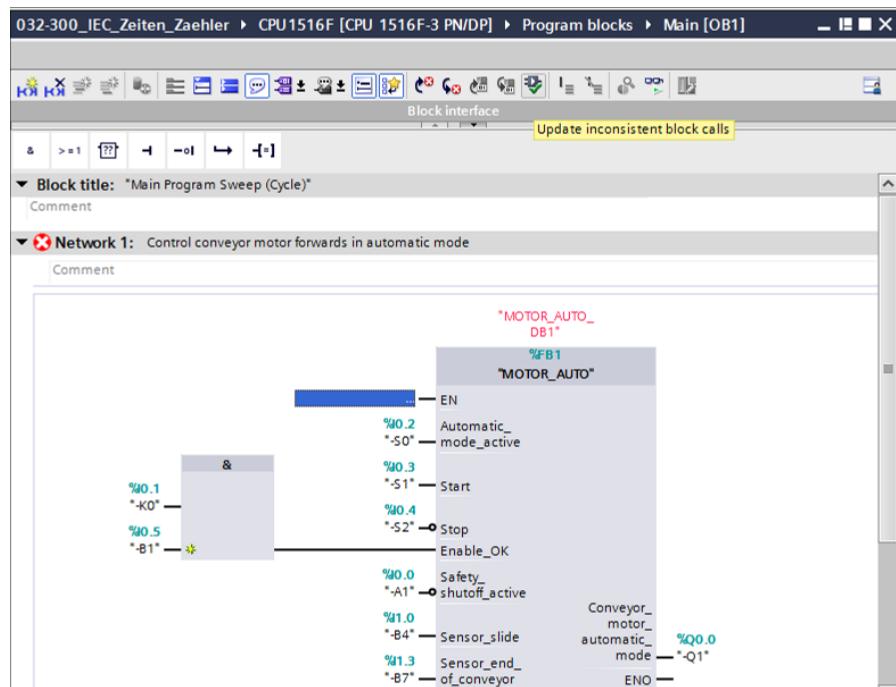


6.3 Update the block call in the organization block

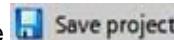
Open the "Main [OB1]" organization block with a double-click.

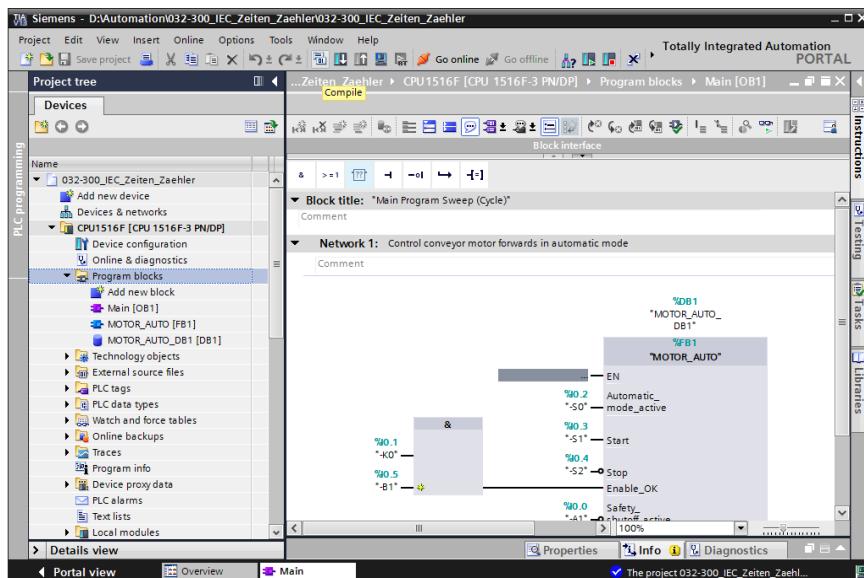


In Network 1 of the "Main [OB1]" organization block, instance data block "MOTOR_AUTO_DB1" for the "MOTOR_AUTO [FB1]" function block appears incorrect, because the additional memory for the TP Timer has not yet been added there. Click the → "  " icon for "Update inconsistent block calls". This will add the "MOTOR_AUTO_DB1" instance data block correctly again. (→ )

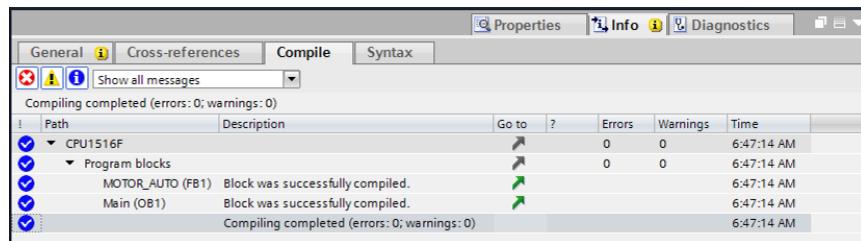


6.4 Save and compile the program

To save your project, select the  button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu (→  → Program blocks → ).

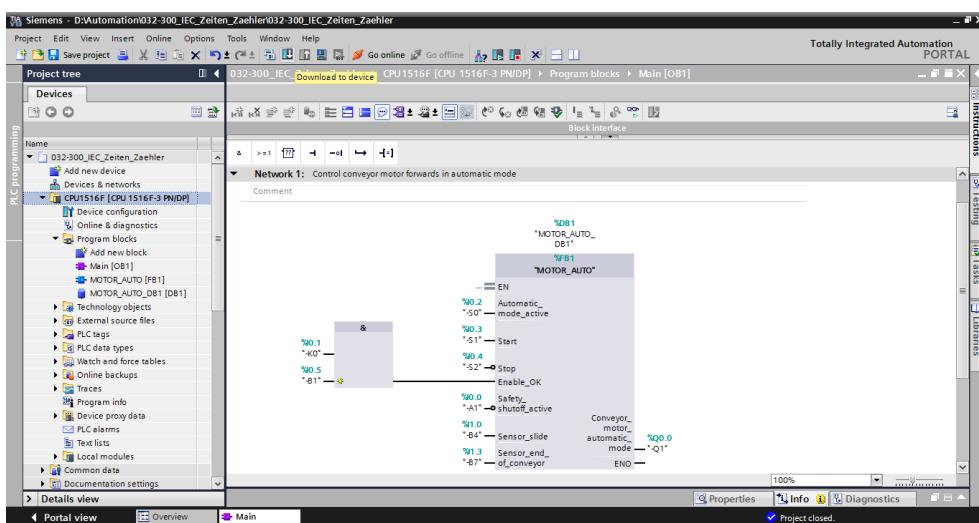


The "Info", "Compile" area shows which blocks were successfully compiled.



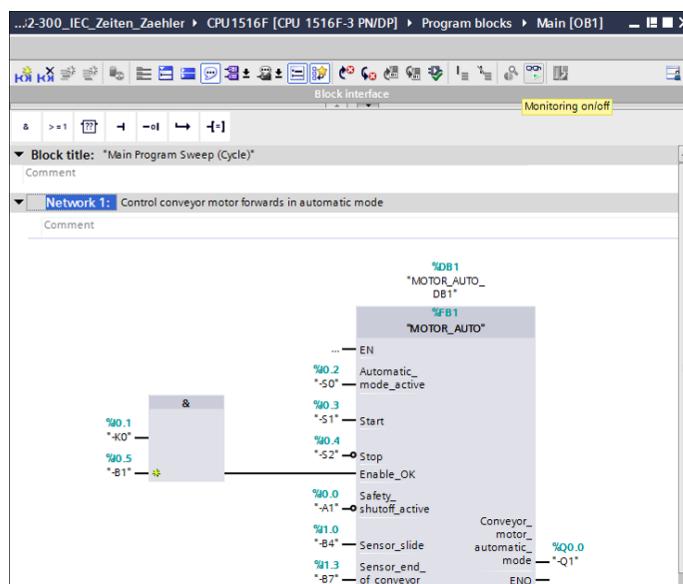
6.5 Download the program

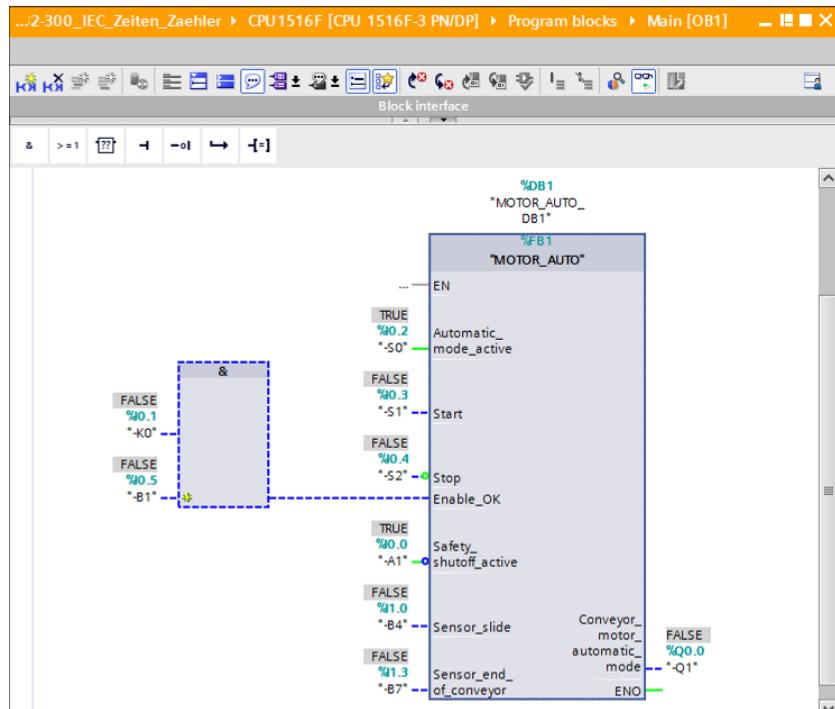
After successful compilation, the complete controller with the created program including the hardware configuration, as previously described in the modules, can be downloaded. (→



6.6 Monitor program blocks

The desired block must be open for monitoring the downloaded program. The monitoring can now be activated/deactivated by clicking the icon. (→ Main [OB1] →)

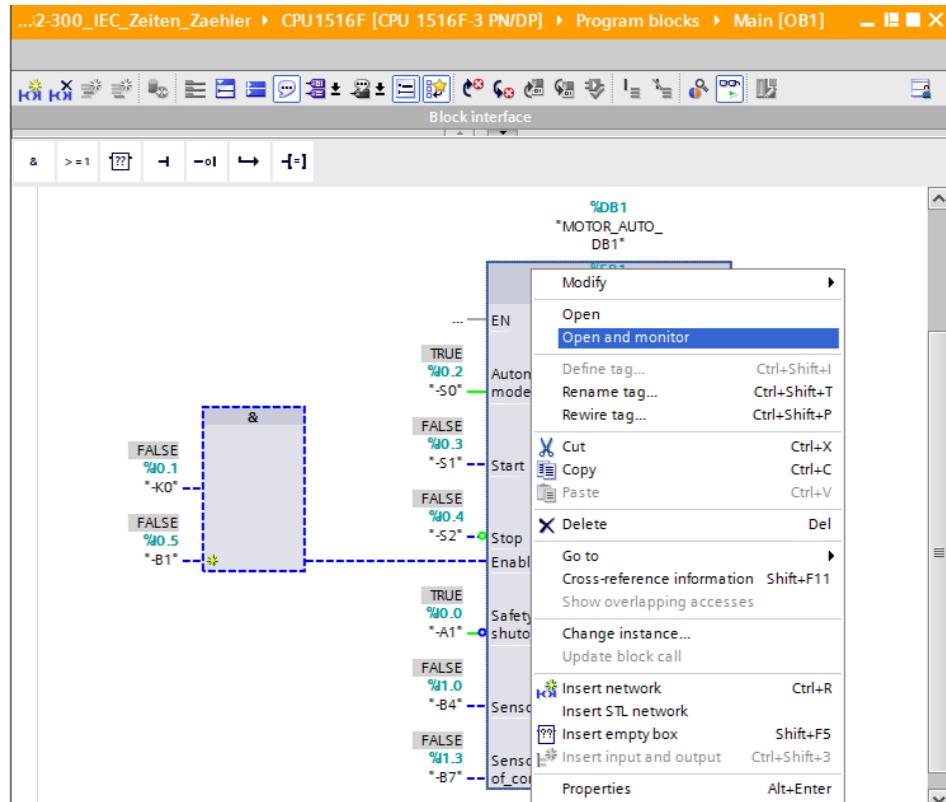


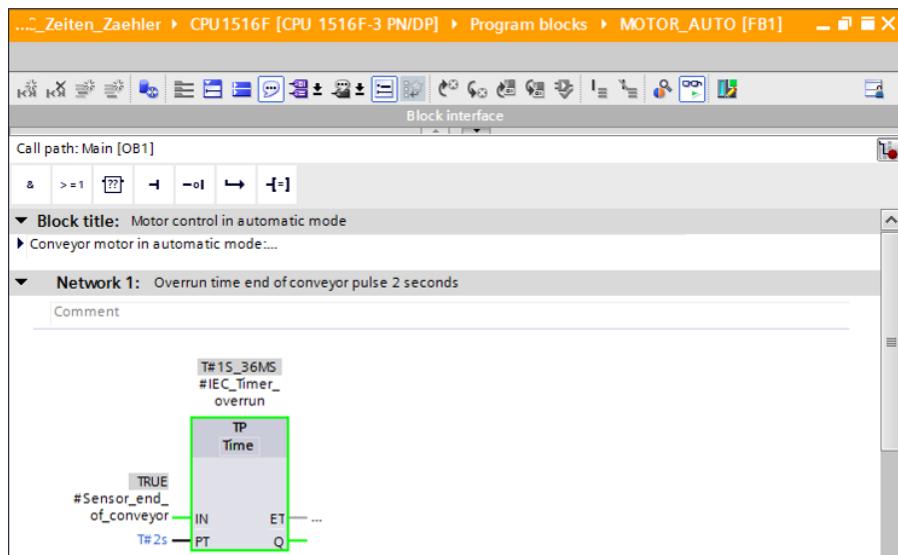


Note: The monitoring here is signal-related and controller-dependent. The signal states at the terminals are indicated with TRUE or FALSE.

The "MOTOR_AUTO" [FB1] function block called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking, thereby allowing the program code in the function block with the TP Timer to be monitored.

(→ "MOTOR_AUTO" [FB1] → Open and monitor)





Note: The monitoring here is function-related and controller-independent. The actuation of sensors and the station status are shown here with TRUE or FALSE.

7 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor at chute activated (-B4 = 1) then conveyor motor forwards fixed speed (-Q1 = 1) switches on and stays on.	
4	Sensor at end of conveyor activated (-B7 = 1) → -Q1 = 0 (after 2 seconds)	
5	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q1 = 0	
6	Activate EMERGENCY OFF (-A1 = 0) → -Q1 = 0	
7	Manual mode (-S0 = 0) → -Q1 = 0	
8	Switch off station (-K0 = 0) → -Q1 = 0	
9	Cylinder not retracted (-B1 = 0) → -Q1 = 0	
10	Project successfully archived	



Training Curriculum

TIA Portal Module 007

Basics of Diagnostics
with SIMATIC S7-1500

Table of contents

1	Goal.....	230
2	Prerequisite	230
3	Required hardware and software	230
4	Theory	231
4.1	Fault diagnostics and hardware faults	231
4.2	Hardware diagnostics	231
4.3	Diagnostics for program blocks.....	232
5	Task	233
6	Planning	233
6.1	Online interface	233
7	Structured step-by-step instructions.....	234
7.1	Retrieve an existing project.....	234
7.2	Download the program.....	234
7.3	Go online.....	236
7.4	Online & diagnostics of the SIMATIC S7 controller.....	239
7.5	Online/offline comparison	246
7.6	Monitor and modify tags.....	249
7.7	Force tags	252
7.8	Checklist.....	255

BASICS OF DIAGNOSTIC FUNCTIONS

1 Goal

In this module, the reader will become acquainted with the tools that support troubleshooting.

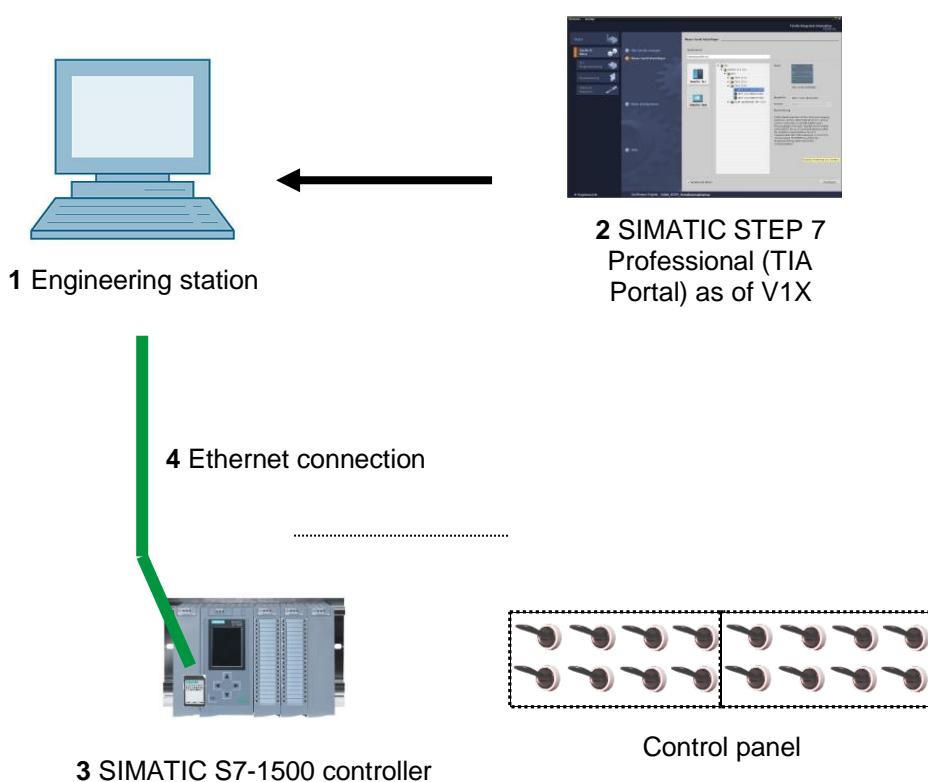
This module will present diagnostic functions that, for example, you can test with the TIA project.

2 Prerequisite

This chapter builds on the hardware configuration of SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations that have digital input and output cards can be used.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 Fault diagnostics and hardware faults

Faults can be caused by a variety of things.

Two error patterns can be distinguished for faults that occur after a transition to RUN.

1. The CPU goes to or stays in STOP mode. The yellow STOP LED lights up, and other indicator LEDs light up on the CPU, power supply unit, IO modules or bus modules.

A CPU fault has occurred in this case. For example, a module in the AS might be defective or have an incorrect parameter assignment, or a bus system fault might be present.

An interruption analysis will be performed in this instance by evaluating the hardware diagnostics and by reading the module status in the diagnostic buffer of the CPU.

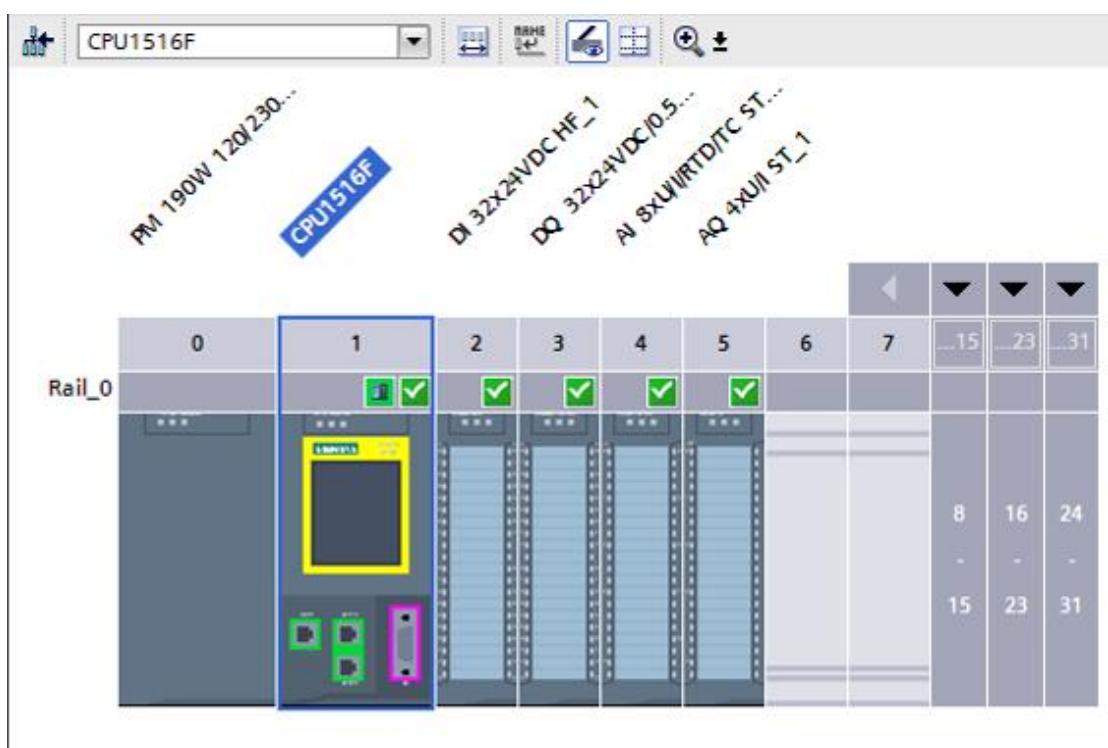
2. The CPU is in faulty RUN mode. The green RUN LED lights up and other indicator LEDs light up or flash on the CPU, power supply unit, IO modules or bus modules.

In this case, a fault may have occurred in the IO devices or power supply.

A visual check is performed initially to narrow down the fault area. The indicator LEDs on the CPU and IO devices are evaluated. The diagnostic data of the faulty IO and bus modules are read from the hardware diagnostics. In addition, a fault analysis can be performed using a watch table on the programming device.

4.2 Hardware diagnostics

The device view in online mode of the TIA Portal provides you a quick overview of the configuration and system status of the automation system.



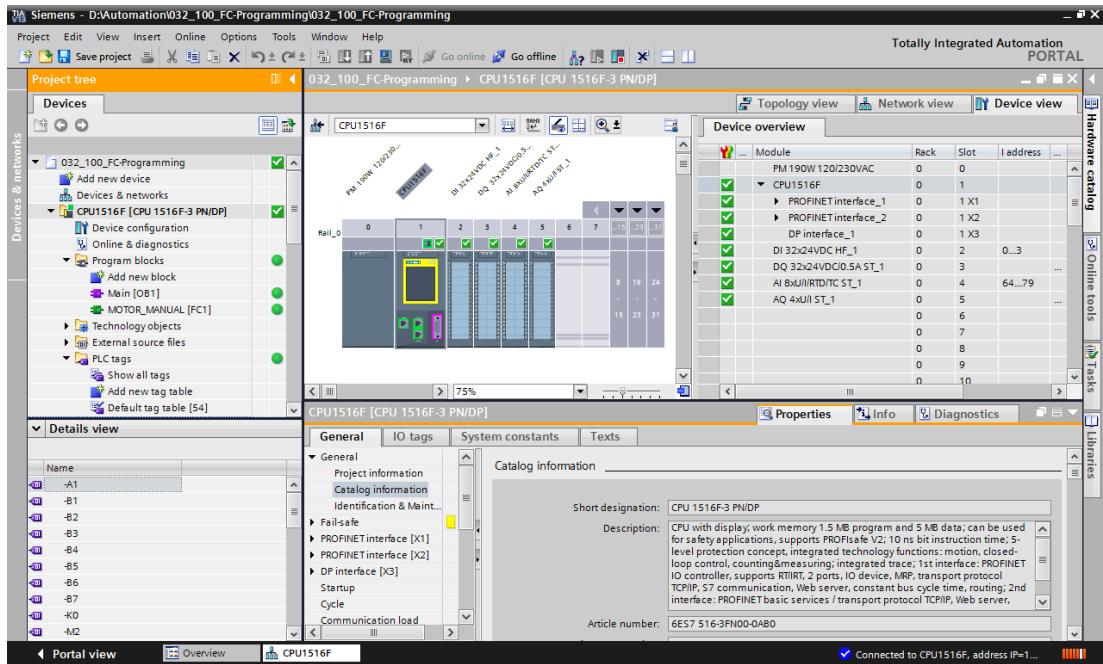


Figure 1: Online view of device configuration

4.3 Diagnostics for program blocks

The project tree window of the TIA Portal in online mode provides you an overview of the programmed blocks of the user program. With the help diagnostic symbols, a comparison of the program blocks used offline and online is displayed.

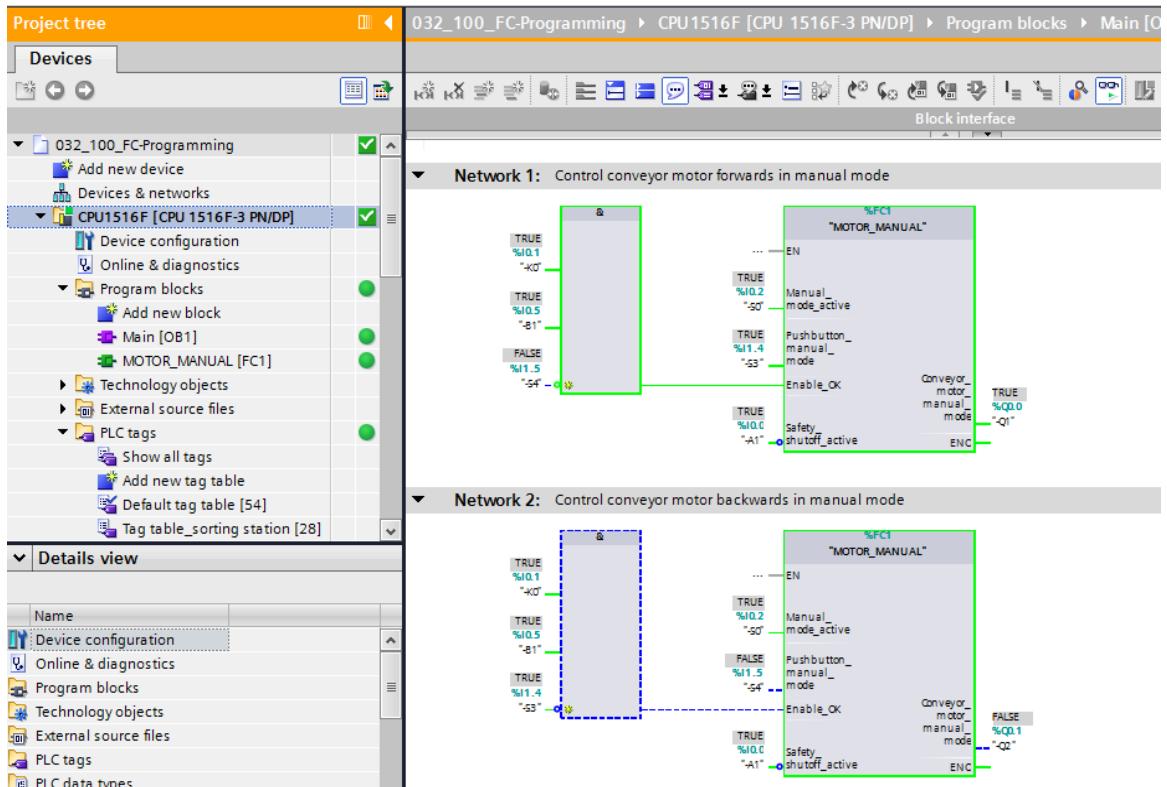


Figure 2: Online view of the Main [OB1] block

5 Task

The following diagnostic functions will be shown and tested in this chapter:

- Diagnostic symbols in the online view of the TIA Portal
- Device diagnostics with module status
- Offline/online comparison
- Monitoring and modifying tags
- Forcing tags

6 Planning

The diagnostic functions will be performed using a finished project as an example.

A project in the TIA Portal that was previously downloaded to the controller should be open for this.

In our case, after starting the TIA Portal, a previously created project will be retrieved from the archive and downloaded to the associated controller.

Afterwards, you can start implementing the diagnostic functions in the TIA Portal.

6.1 Online interface

Online diagnostics are only possible when the correct communication connection to the CPU has been set up beforehand. Here, we are connected via Ethernet/PROFINET.

When connecting online, you must therefore set the appropriate interfaces for your automation system.

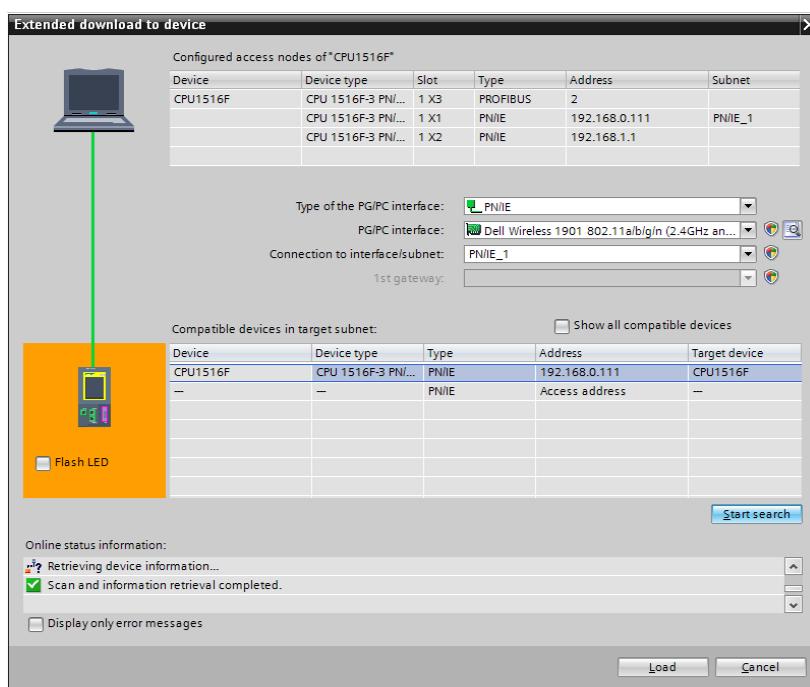


Figure 3: Go online

7 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

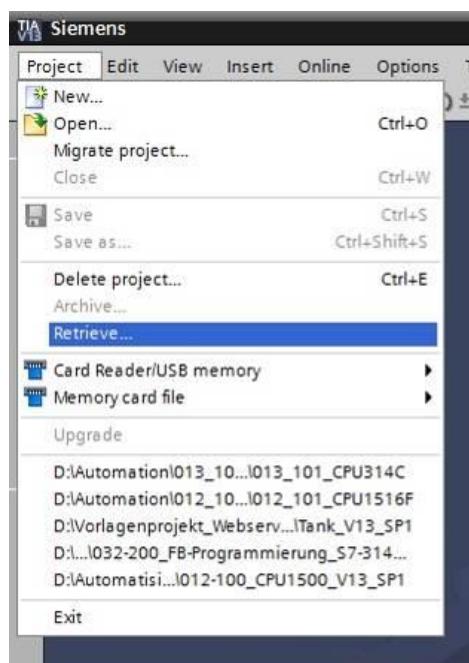
→ Before we can start the diagnostic functions, we need a project with programming and a hardware configuration.

(e.g., SCE_EN_032-100_FC-Programming....zap).

To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view.

Confirm your selection with "Open".

(→ Project → Retrieve → Select a .zap archive → Open)

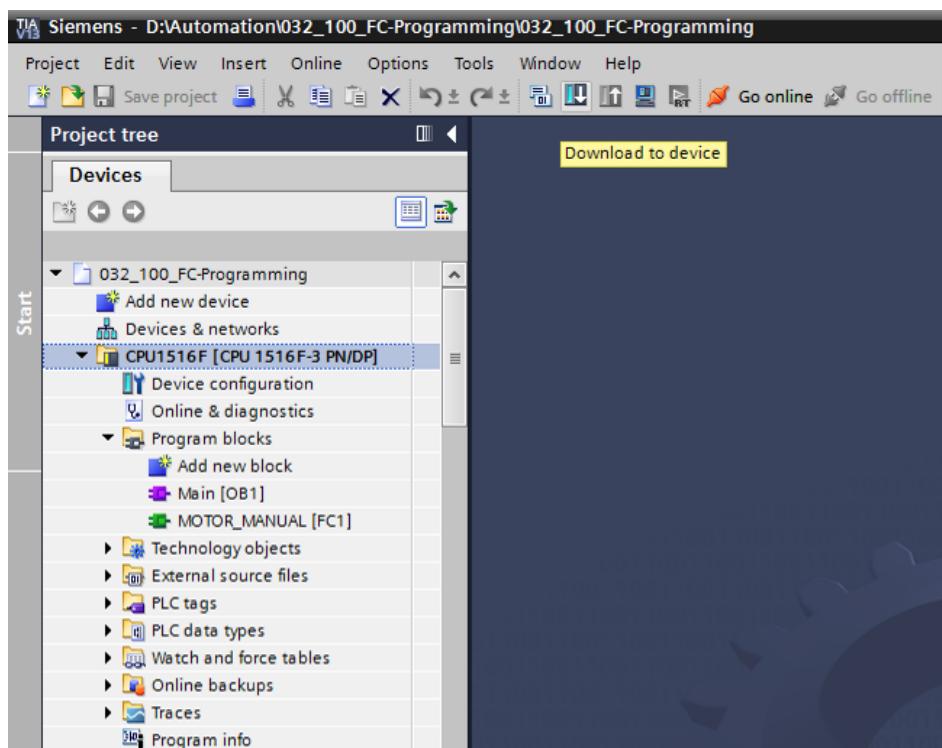


→ The next step is to select the target directory where the retrieved project will be stored.

Confirm your selection with "OK". (→ Target directory → OK)

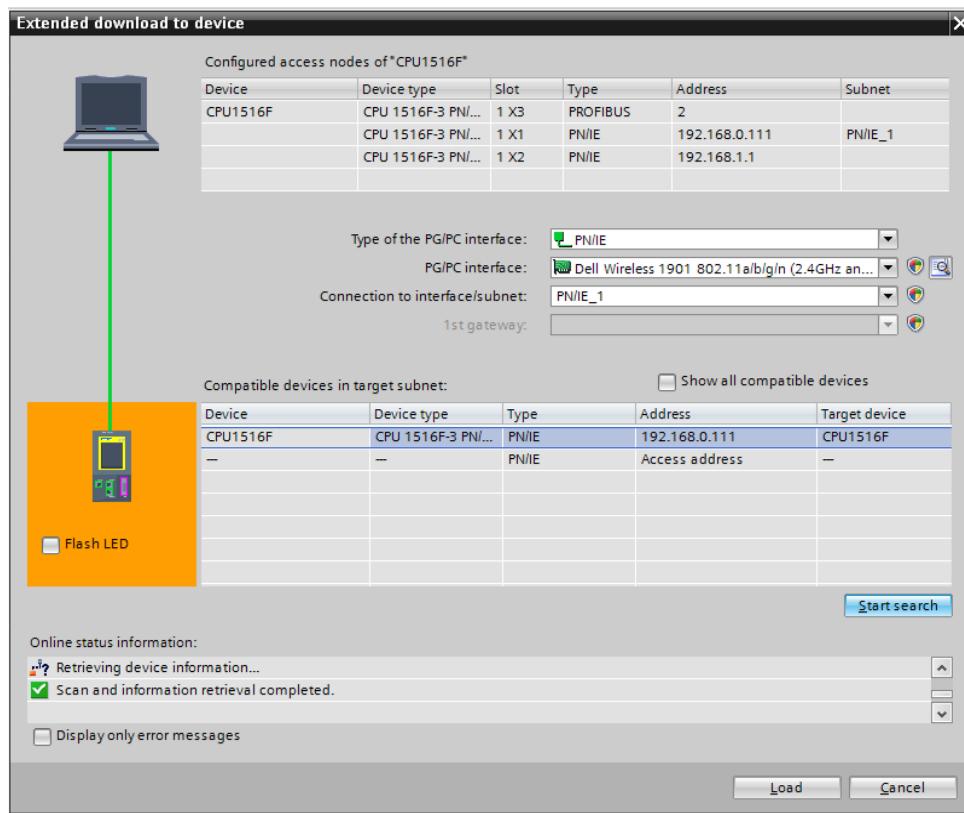
7.2 Download the program

→ After the project has been successfully retrieved, the controller can be selected and downloaded together with the created program. (→ 

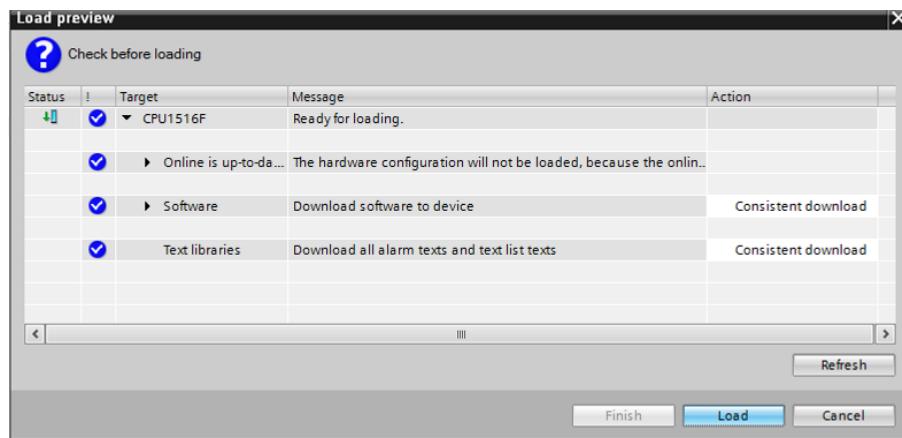


- Select the correct interfaces and click "Start search". (→ "PN/IE" → Selection of the network adapter of the PG/PC → Direct at slot '1 X1' → "Start search")

Once "Scan and information retrieval completed" appears, click "Load". (→ "Load")

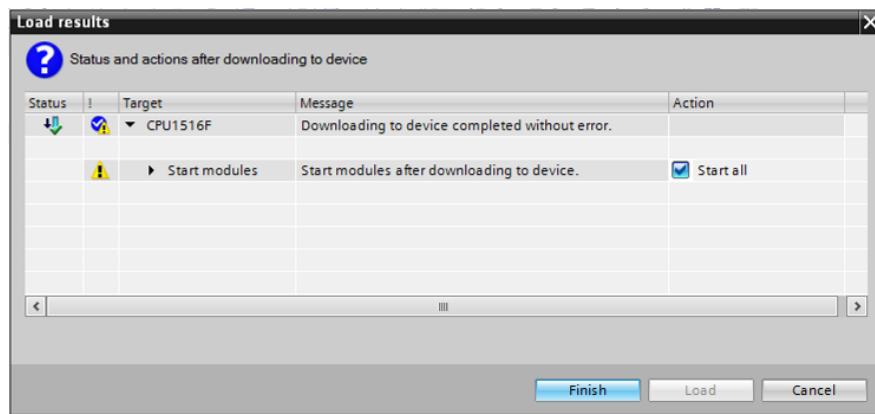


- Before downloading can be started, other actions may have to be set (highlighted in pink). Click "Load" again. (→ "Load").



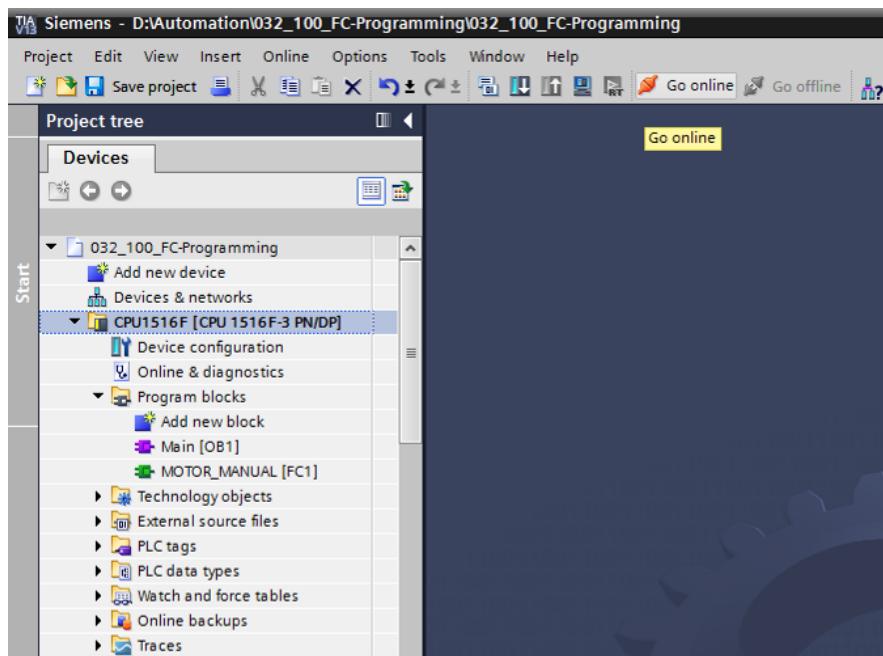
→ After loading, first select the "Start all" check box under Action.

Click "Finish". (→ select check box → "Finish")

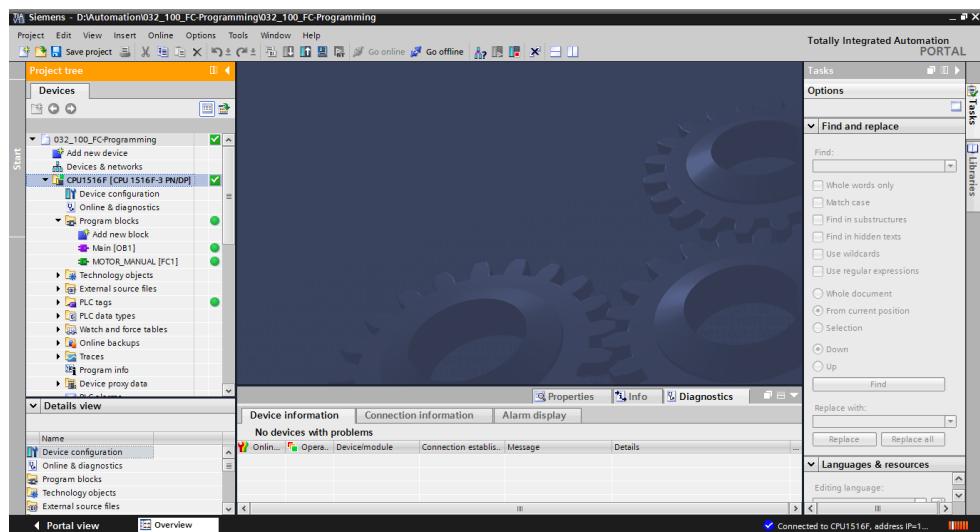


7.3 Go online

→ To get started with the diagnostic functions, we will select our controller ("PLC_1") and click "Go online". (→ PLC_1 → Go online)



- Once the online connection to the "PLC_1" controller is established, the CPU can be started or stopped with the following buttons . Diagnostic information in the form of symbols will already be available in the project tree and in the diagnostics window.

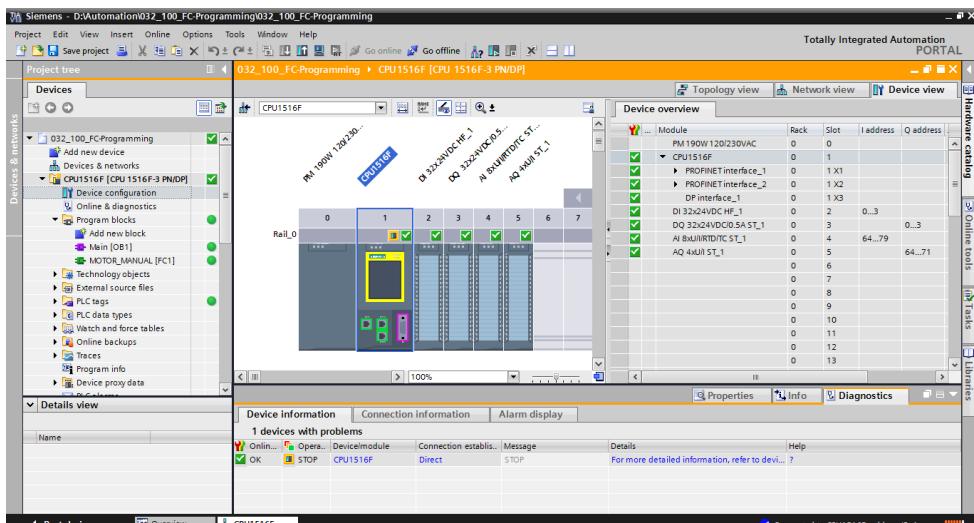


Symbols for the comparison status in the project tree

- The diagnostic symbols in the project tree show a comparison status representing the online/offline comparison of the project structure.

Symbol	Meaning
	Folder contains objects whose online and offline versions are different (only in the project tree)
	Online and offline versions of the object are different
	Object only exists online
	Object only exists offline
	Online and offline versions of the object are the same

- Double-click "Device configuration".



Operating state symbols for CPUs and CPs

- The graphical representation and device information window show the various operating states of the CPU or communication processors (CPs).

Symbol	Operating state
	RUN
	STOP
	STARTUP
	HOLD
	DEFECT
	Unknown operating state
	The configured module does not support display of the operating state.

Diagnostic symbols for modules and devices in the device overview

- The graphical representation and Device overview window display the operating states of the various modules, CPU, or CP using the following symbols.

Symbol	Meaning
	The connection to a CPU is currently being established.
	The CPU is not accessible at the set address.
	The type of CPU configured and type of CPU actually present are incompatible.
	During the establishment of the online connection to a protected CPU, the password dialog was terminated without input of the correct password.
	No fault
	Maintenance required
	Maintenance demanded
	Fault
	The module or device is deactivated.
	The module or device cannot be accessed from the CPU (valid for modules and devices below a CPU).
	Diagnostic data is not available because the current online configuration data differs from the offline configuration data.
	The configured module or device and the module or device actually present are incompatible (valid for modules or devices below a CPU).
	The configured module does not support display of the diagnostic status (valid for modules below a CPU).
	The connection has been established, but the state of the module is currently still being determined.
	The configured module does not support display of the diagnostic status.
	Error in lower-level component: A error is present in at least one lower-level hardware component.

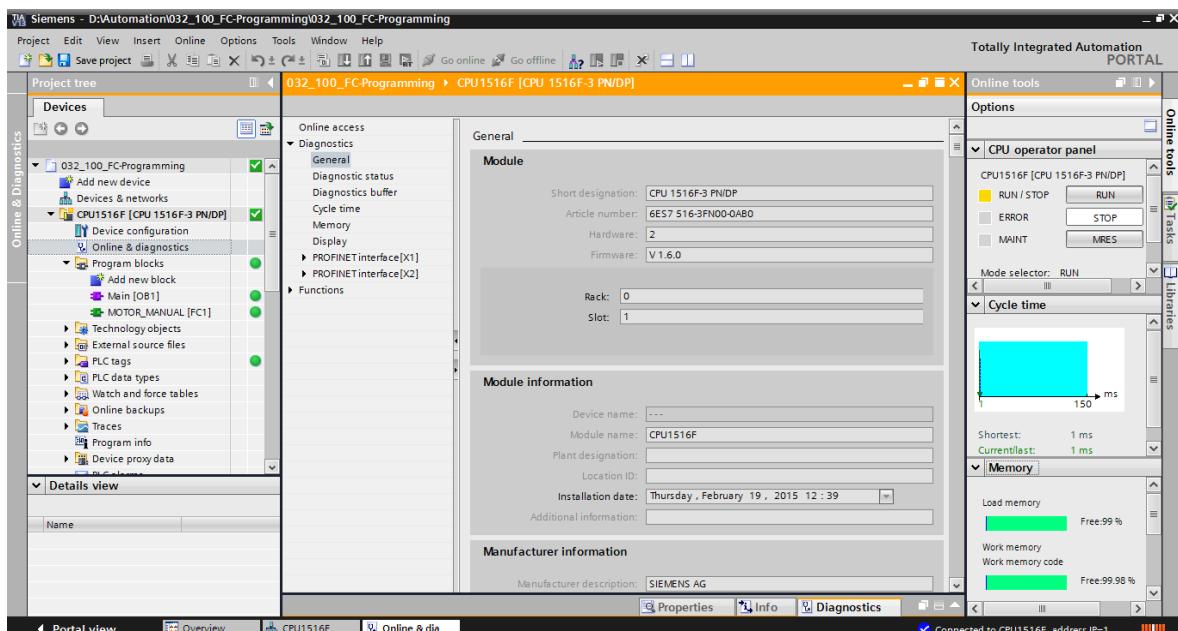
Color coding of ports and Ethernet cables

- The status of ports and Ethernet cables can be diagnosed in the Network view and Topology view.
- The following table shows the possible colors and their respective meaning.

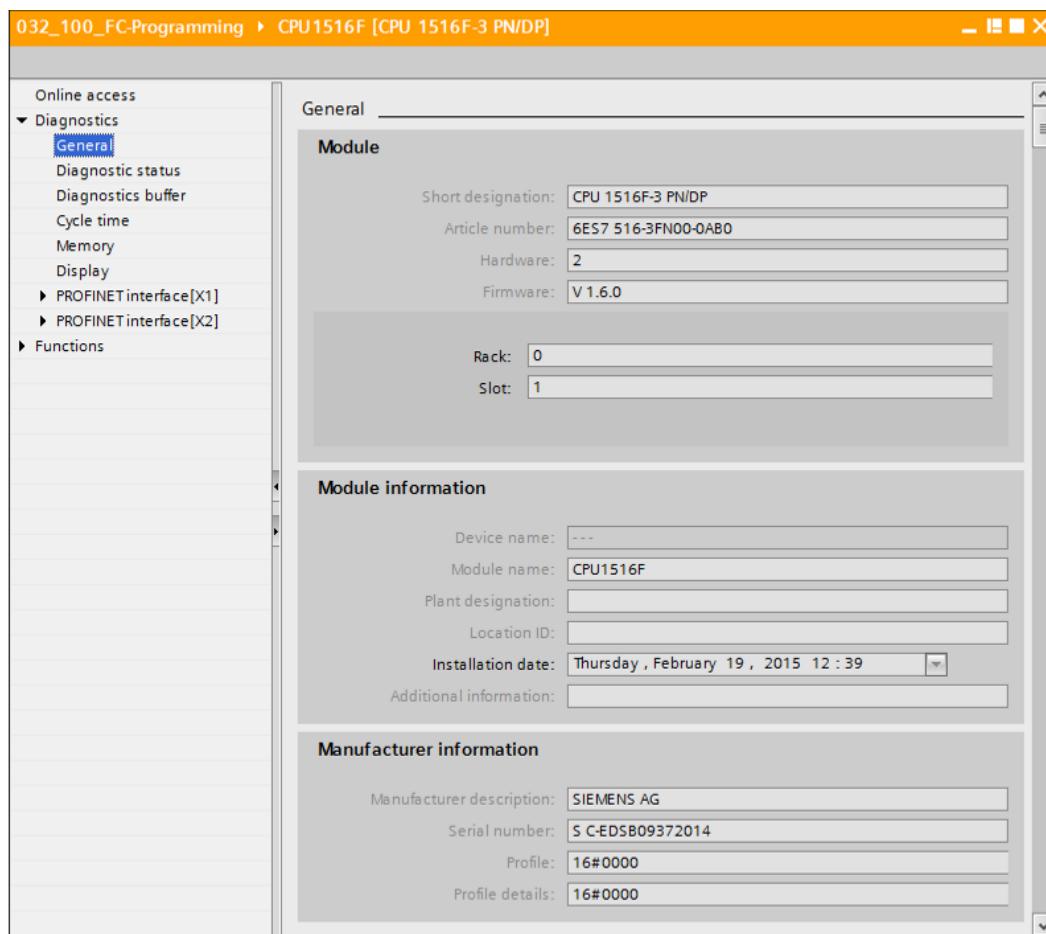
Color	Meaning
Green	No fault or maintenance required
Yellow	Maintenance demanded
Red	Communication error

7.4 Online & diagnostics of the SIMATIC S7 controller

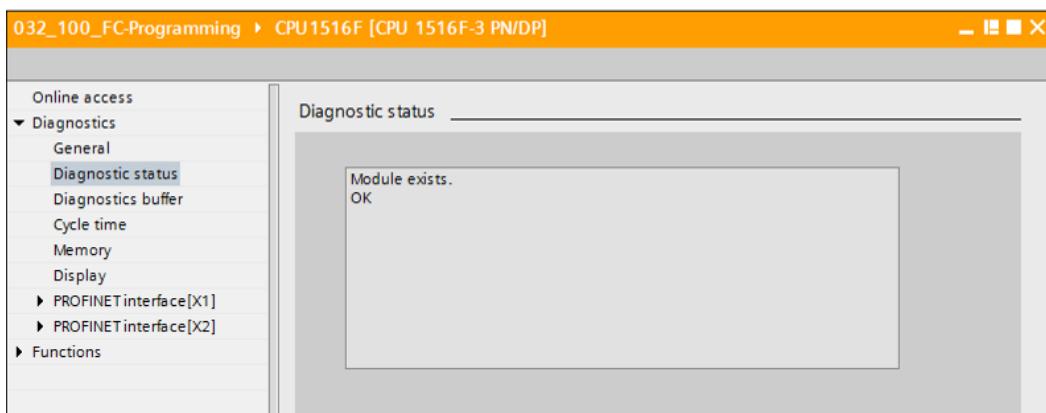
- Double-click "Online & diagnostics" in the project tree.
- (→ Online & diagnostics)
- A CPU operator panel, the cycle time and the memory utilization are displayed in the Online tools on the right side. Switch the CPU to RUN here. (→ RUN)



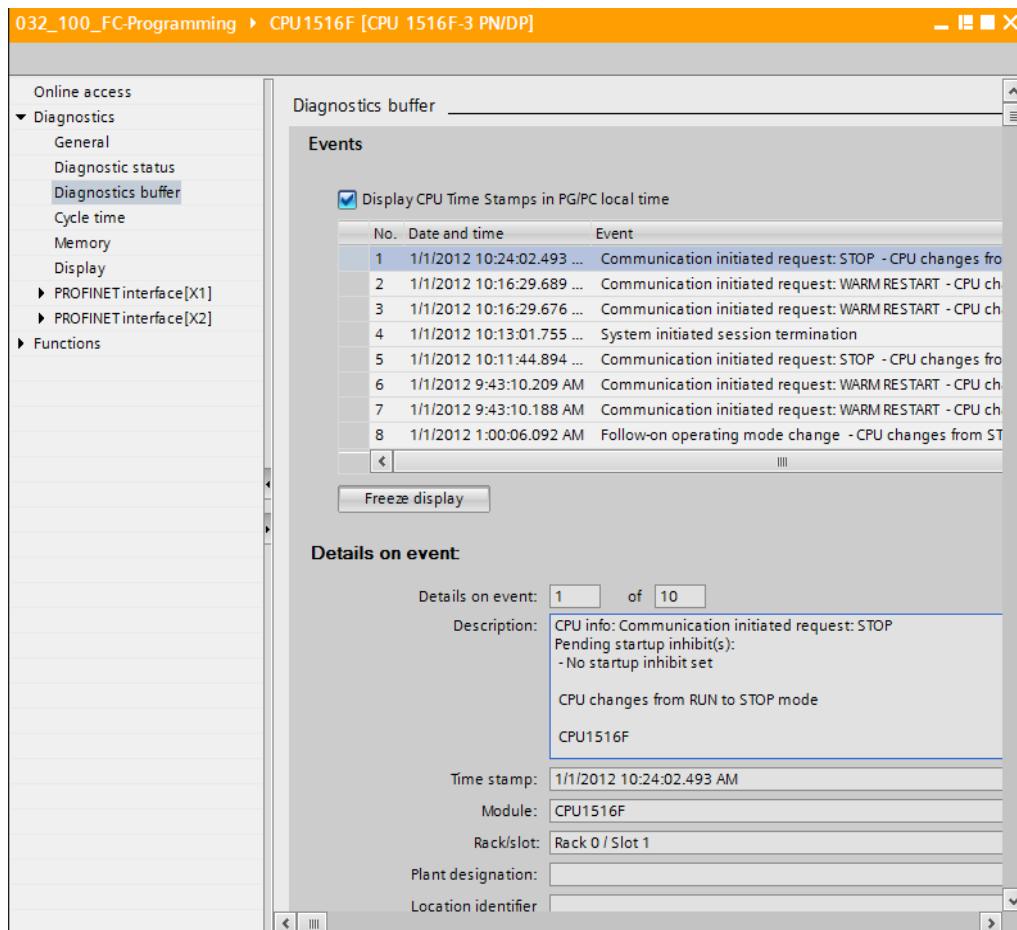
- The working area window contains general information about the CPU. (→ General)



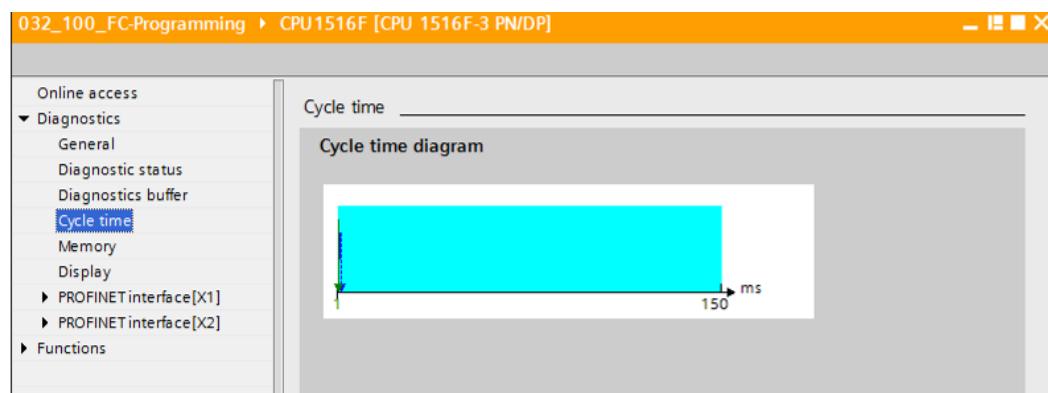
- If diagnostic information is available, it is displayed in Diagnostic status.
(→ Diagnostic status)



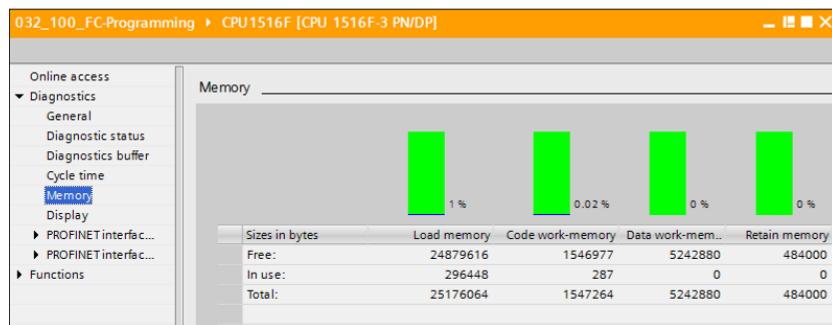
- Detailed Information on the individual events is displayed in Diagnostics buffer.
 (→ Diagnostics buffer)



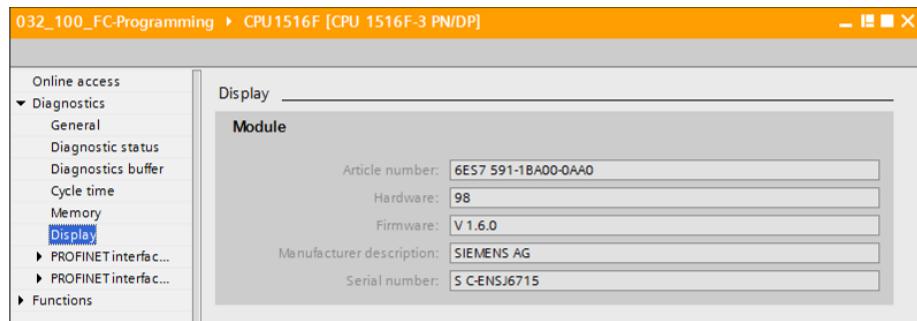
- Next you receive information about the cycle time of the executed program.
 (→ Cycle time)



→ The memory utilization can be seen here in detail. (→ Memory)

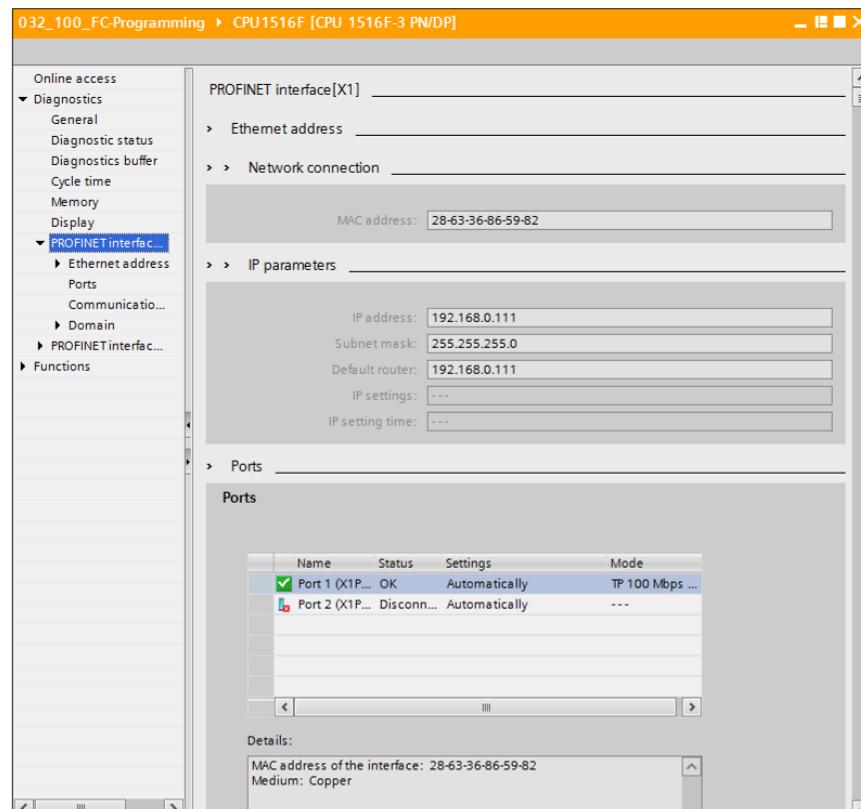


→ Information about the display is also available for the CPU 1516F. (→ Display)



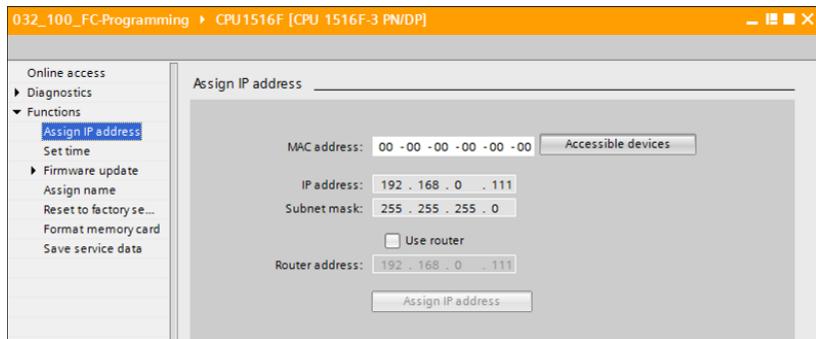
→ The network settings and the status of the PROFINET interfaces [X1] and [X2] can also be displayed.

(→ PROFINET interface [X1] or → PROFINET interface [X2])



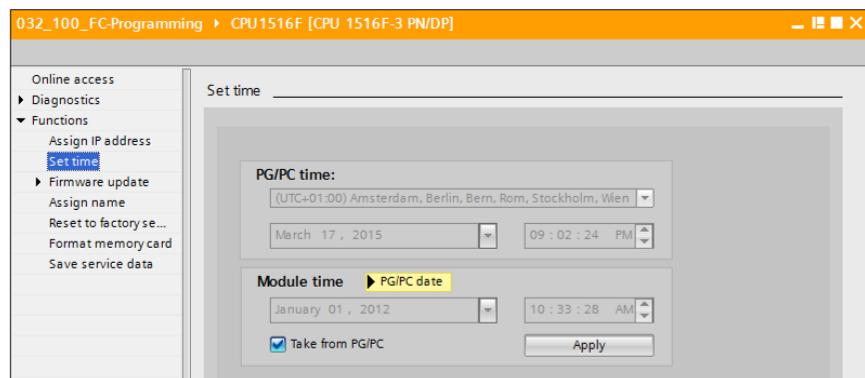
- In "Assign IP address" under Functions, you can assign the IP address to a controller.
However, this is only possible when no hardware has been downloaded to the CPU.

(→ Functions → Assign IP address)



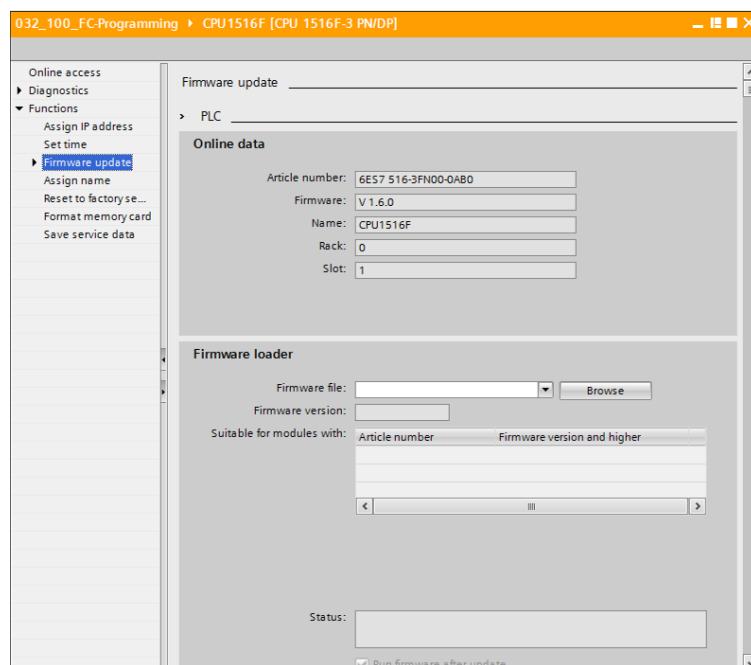
- Under "Set time", you can set the time of the CPU.

(→ Functions → Set time)



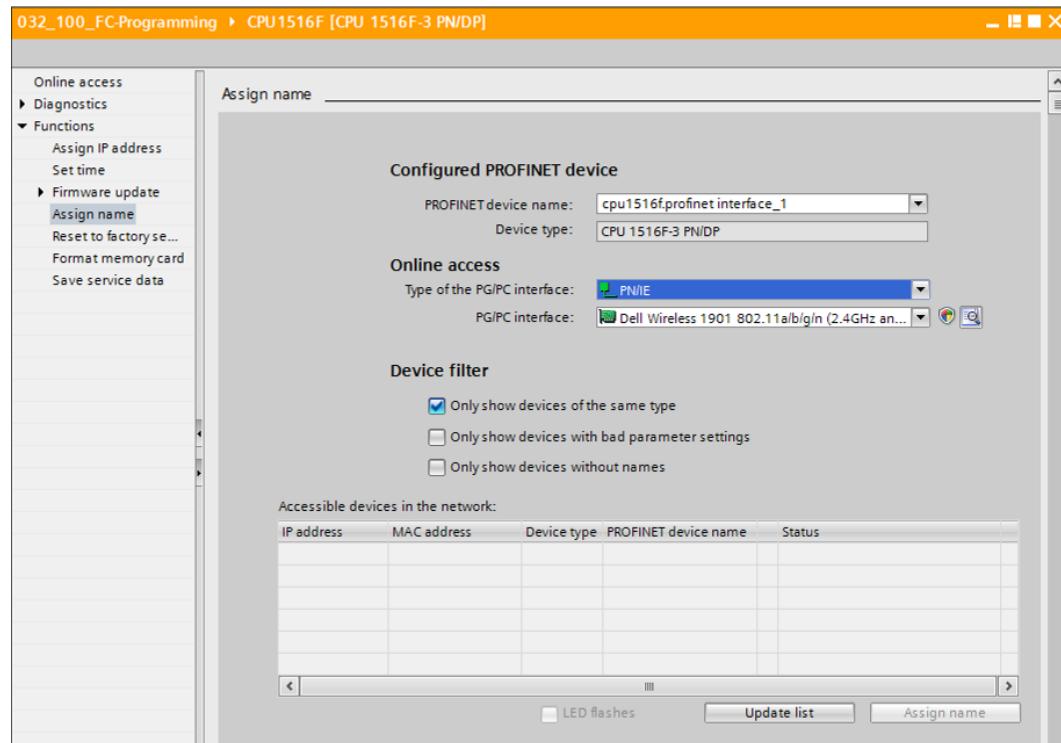
- Under "Firmware update", you can update the firmware of the PLC or the display.

(→ Functions → Firmware update)



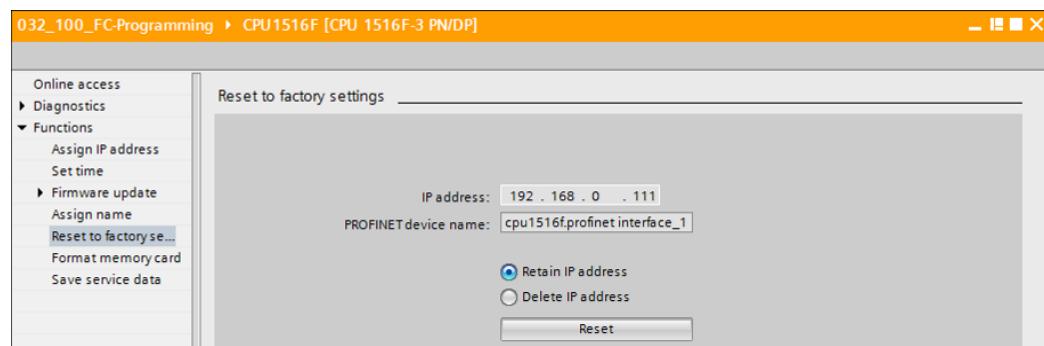
- Under "Assign name", you can assign a PROFINET device name to the configured field devices on PROFINET. The device name of the CPU cannot be changed here. It can only be changed by downloading a modified hardware configuration.

(→ Functions → Assign name)



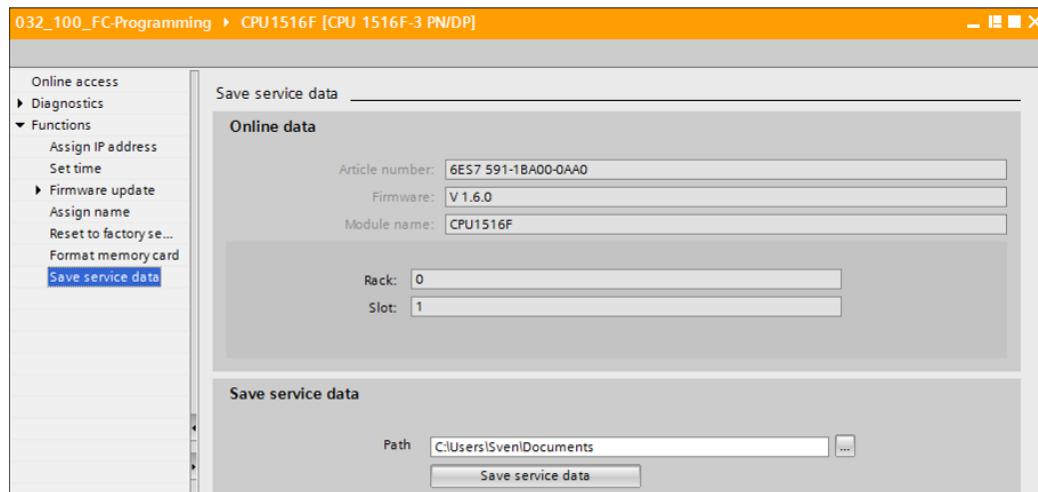
- Under "Reset to factory settings", you can restore the factory settings for the CPU. After restoring the factory settings, the CPU configuration and the program must be imported again from the inserted memory card. Therefore, the memory card must be formatted before the restoring the factory settings.

(→ Format memory card → Format → Reset to factory settings → Retain or delete IP address → Reset)



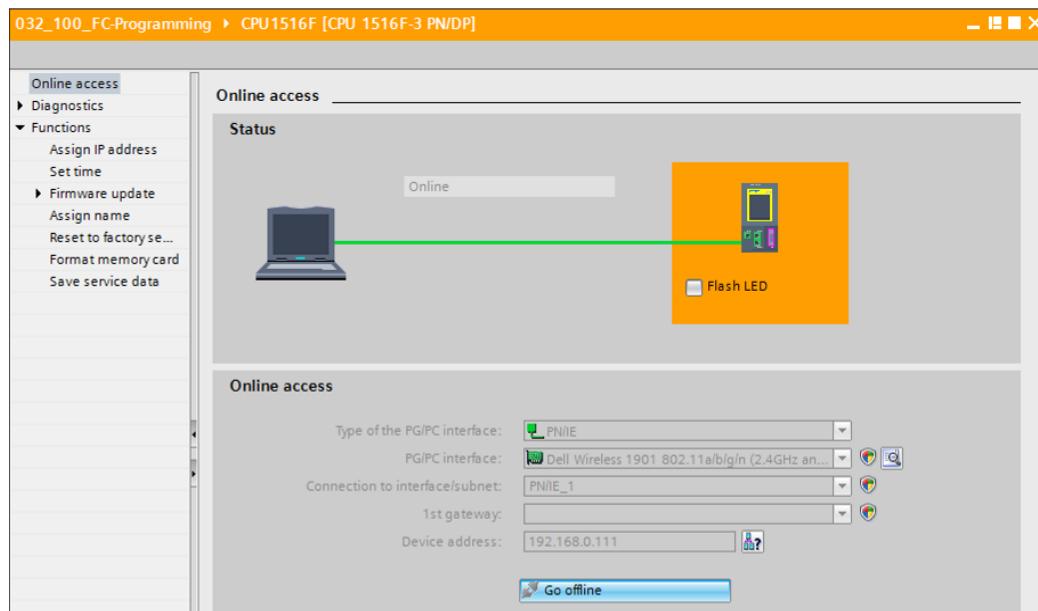
→ Finally, the service data can be saved under Functions.

(→ Functions → Save service data)



→ The online connection should be disconnected again before the next chapter.

(→ Online access → Disconnect online connection)



→ The TIA Portal is now back in offline mode. The orange-colored bars and the diagnostic symbols are no longer displayed.

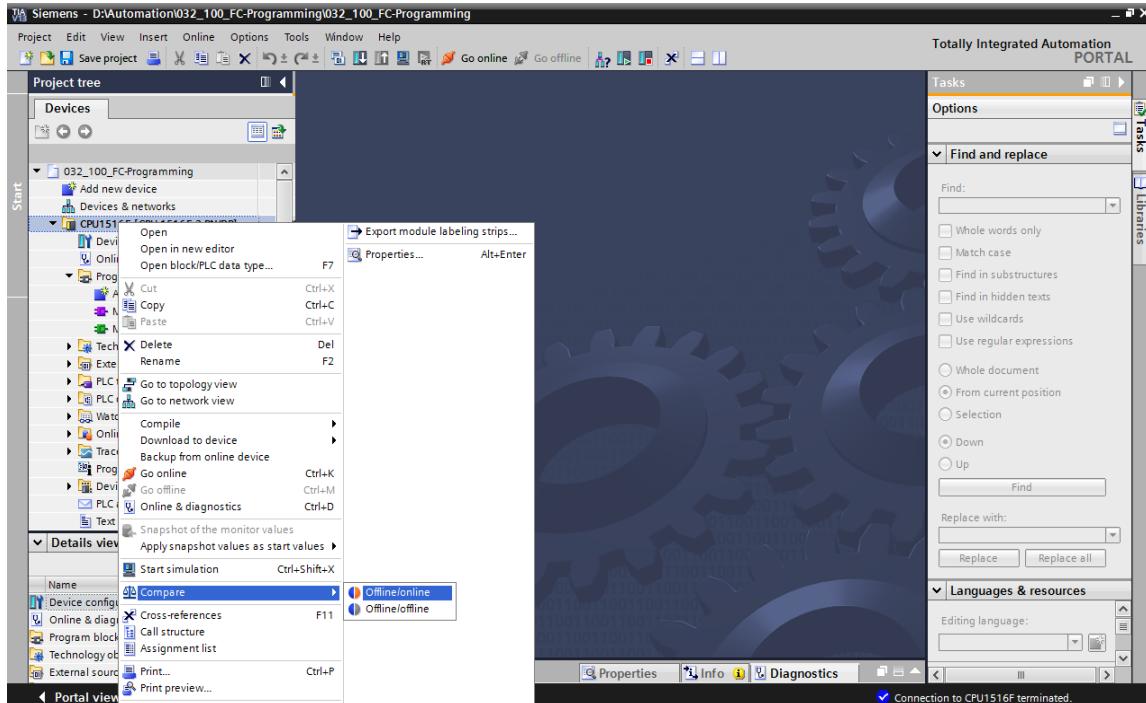
7.5 Online/offline comparison

- It is often important to know whether the saved data matches the data loaded in the controller. First, remove the negation from the "Safety_shutoff_active" tag at the AND function in the "MOTOR_MANUAL [FC1]" block.

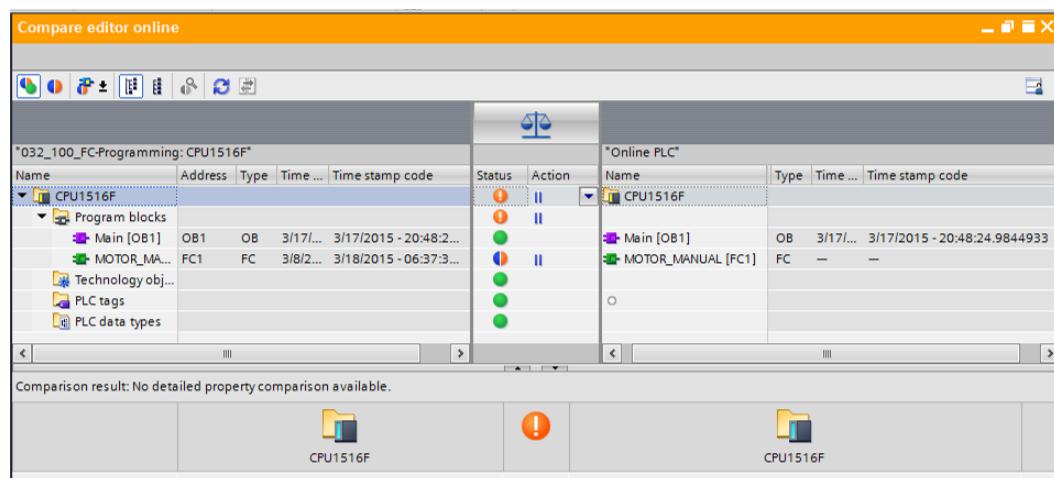
Save the "MOTOR_MANUAL [FC1]" block, but do **NOT** download it to the controller.

Close the "MOTOR_MANUAL [FC1]" block again.

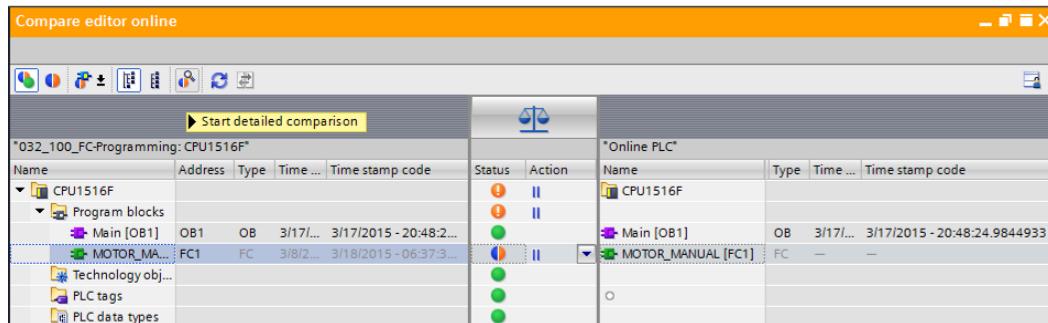
- To compare, right-click the "PLC_1" controller and select "Compare", "Offline/online".
(→ Select controller → Compare → Offline/online)



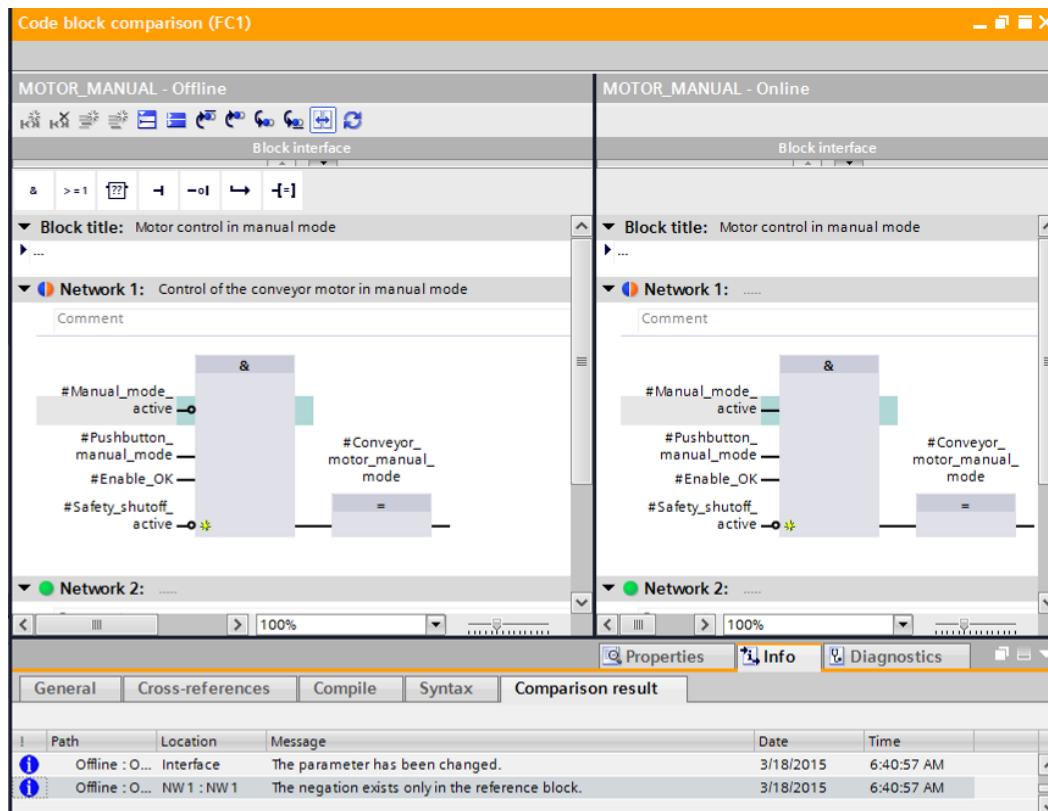
- The Compare editor online opens.



- If, for example, block differences are indicated , first select the block involved. You can then click the  button to "Start detailed comparison".
 (→ MOTOR_MANUAL → Start detailed comparison).



- The selected offline/online block will be compared in the code block comparison. A detailed description of the difference is shown in the comparison result.

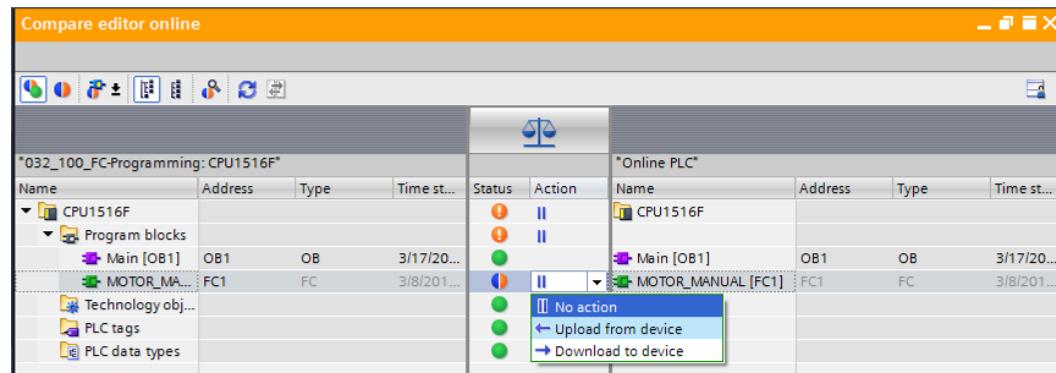


- Close the window of the code block comparison.

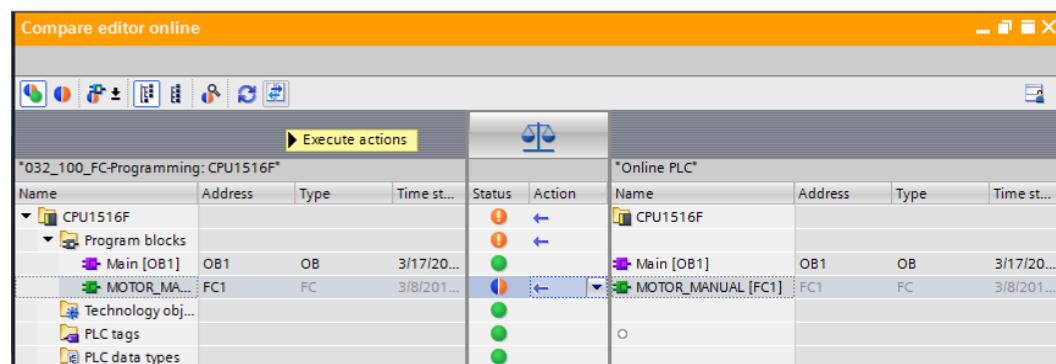
- An action can be selected for the block involved in the Compare editor.

Either the "MOTOR_MANUAL" block will be downloaded from the programming device to the controller and overwritten there or the "MOTOR_MANUAL" block will be imported from the controller and overwritten in the TIA Portal.

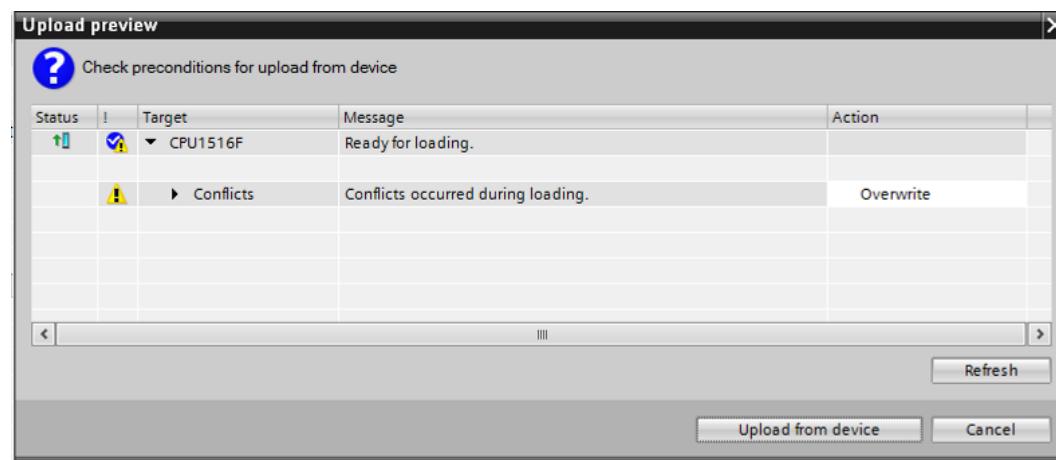
Select the "Upload from device" action. (← Upload from device)



- Click the "Execute actions" button . (→ Execute actions)



- Confirm "Upload from device". (→ Upload from device)



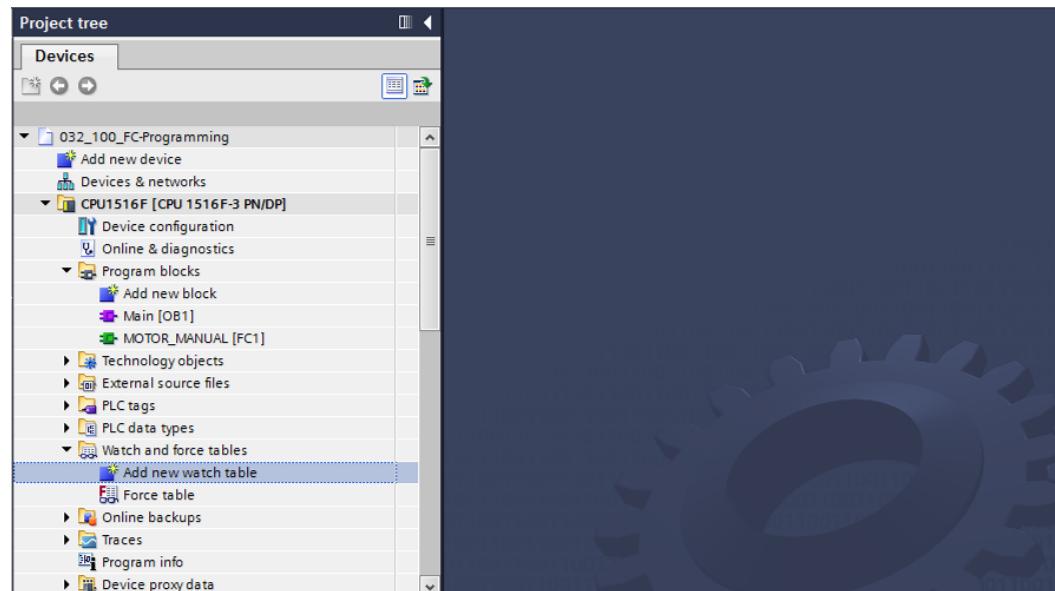
- After the upload, there are no more differences. You should now save your project again and close the online connection.

7.6 Monitor and modify tags

→ To monitor and modify tags, you need a watch table.

Double-click "Add new watch table" in the project tree.

(→ Add new watch table)



→ Open the newly created "Watch_table_1" by double-clicking it. (→ "Watch_table_1")

→ You can enter individual tags in the table or you can select the "Tag_table_sorting_station" and the tags to be monitored, and drag them from the Details view to the watch table.

(→ Default tag table)

The screenshot shows the TIA Portal interface with the 'Watch table' table open. The table has columns: Name, Address, Display format, Monitor value, Modify value, and a delete icon. The table contains the following data:

	Name	Address	Display format	Monitor value	Modify value
1	"-S0"	%I0.2	Bool		
2	"-S3"	%I1.4	Bool		
3	"-K0"	%I0.1	Bool		
4	"-B1"	%I0.5	Bool		
5	"-S4"	%I1.5	Bool		
6	"-A1"	%Q0.0	Bool		
7	"-Q1"	%Q0.0	Bool		
8	<Add new>				

Below the table, the 'Details view' shows a list of PLC tags:

Name	Data type	Details
-Q1	Bool	%Q0.0
-Q2	Bool	%Q0.1
-Q3	Bool	%Q0.2
-S0	Bool	%I0.2

- To have all monitoring and modifying functions available for selection, the following columns can be displayed:

'All modify columns' and 'All expanded mode columns' .

Continue by selecting the trigger timing for the monitoring.

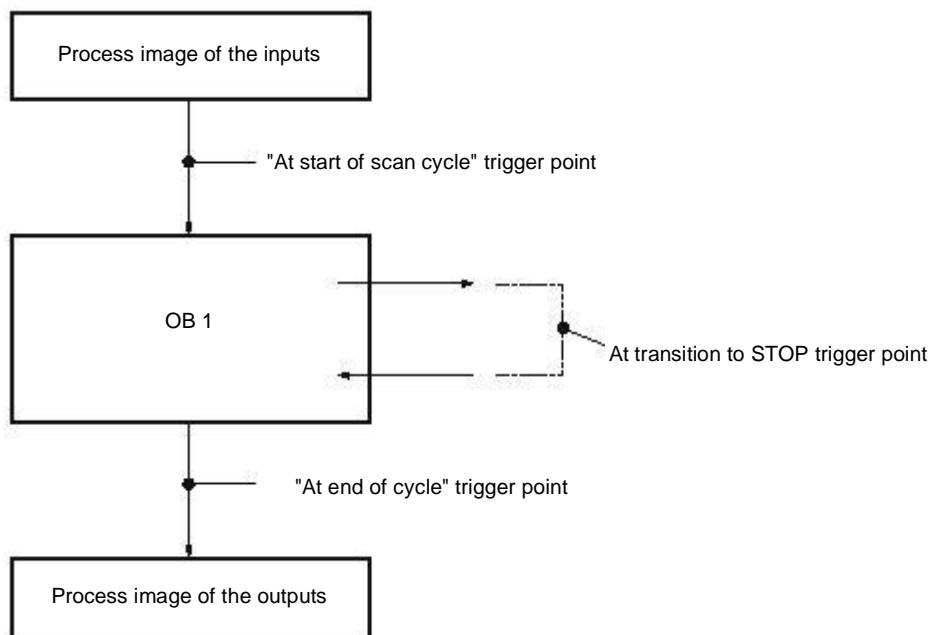
(→ Permanent)

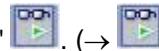
032_100_FC-Programming > CPU1516F [CPU 1516F-3 PN/DP] > Watch and force tables > Watch table_1									
	i	Name	Address	Display form...	Monitor val...	Monitor with trig...	Modify with trig...	Modify value	Comment
1		"-S0"	%I0.2	Bool		Permanent	Permanent		
2		"-S3"	%I1.4	Bool		Permanent	Permanent		
3		"-K0"	%I0.1	Bool		Permanent	Permanent		
4		"-B1"	%I0.5	Bool		Permanent	Permanent		
5		"-S4"	%I1.5	Bool		Permanent	Permanent		
6		"-A1"	%I0.0	Bool		Permanent	Permanent		
7		"-Q1"	%Q0.0	Bool	▼	Permanent	Permanent		
8		<Add new>				Permanent	Permanent		

Permanent
Permanently, at start of scan cycle
Once only, at start of scan cycle
Permanently, at end of scan cycle
Once only, at end of scan cycle
Permanently, at transition to STOP
Once only, at transition to STOP

The following monitoring and modifying modes are available:

- Permanent (in this mode, the inputs are monitored/modified at the start of the cycle and the outputs at the end.)
- Once only, at start of scan cycle
- Once only, at end of scan cycle
- Permanently, at start of scan cycle
- Permanently, at end of scan cycle
- Once only, at transition to STOP
- Permanently, at transition to STOP



- Next, click "Monitor all values once and now"  or "Monitor all values according to trigger settings"  . (→  Monitor all)

	Name	Address	Display form...	Monitor value	Monitor with trig...	Modify with trigge	Modify value	Comment
1	"-S0"	%IO.2	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
2	"-S3"	%I1.4	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
3	"-K0"	%IO.1	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
4	"-B1"	%IO.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
5	"-S4"	%I1.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
6	"-A1"	%IO.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
7	"-Q1"	 %Q0.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
8	<Add new>							

- To modify tags, enter the desired "Modify values". Next, click  to "Modify all activated values once and now" or  to "Modify all activated values by Modify with trigger condition".
- (→ TRUE →  modifies all activated values by "Modify with trigger condition")

	Name	Add...	Display form...	Monitor value	Monitor with trig...	Modify with trigge	Modify value	Comment
1	"-S0"	 %IO.2	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
2	"-S3"	%I1.4	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
3	"-K0"	%IO.1	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
4	"-B1"	%IO.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
5	"-S4"	%I1.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
6	"-A1"	%IO.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
7	"-Q1"	 %Q0.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input checked="" type="checkbox"/> TRUE	
8	<Add new>							

- Confirm the warning with 'Yes'. (→ Yes)



- The output becomes active even though the programmed conditions are not met.

	Name	Address	Display form...	Monitor value	Monitor with trig...	Modify with trigge	Modify value	Comment
1	"-S0"	%IO.2	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
2	"-S3"	%I1.4	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
3	"-K0"	%IO.1	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
4	"-B1"	%IO.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
5	"-S4"	%I1.5	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
6	"-A1"	%IO.0	Bool	<input type="checkbox"/> FALSE	Permanent	Permanent	<input type="checkbox"/>	
7	 "-Q1"	 %Q0.0	Bool	<input checked="" type="checkbox"/> TRUE	Permanent	Permanent	<input checked="" type="checkbox"/>	
8	<Add new>							

Note: If the watch table is closed or the connection to the PLC is lost, all modify commands become ineffective.

7.7 Force tags

→ The “Force” function can be used to assign a fixed value to tags. Force values are specified in a similar way as for the "Modify tags" function but, in contrast, are retained after the CPU is stopped. The main differences between "Modify tags" and the "Force" function are as follows: In contrast with "Modify tags", the "Force" function does not allow you to assign values to data blocks, timers, counters and bit memory.

IO device inputs (e.g., IWxx:P) cannot be modified, although they can be pre-assigned by the "Force" function.

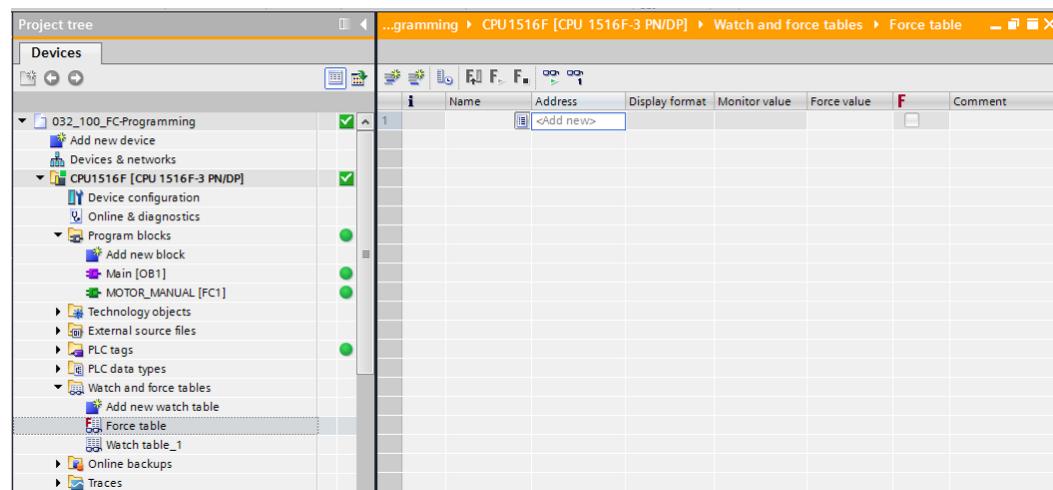
Unlike with the "Modify" function, values permanently assigned by the "Force" function cannot be overwritten by the user program.

If you close the force table, the force values are retained. This is not the case with the "Modify" function.

If the online connection to the CPU is interrupted, the tags assigned with the "Force" function retain their value.

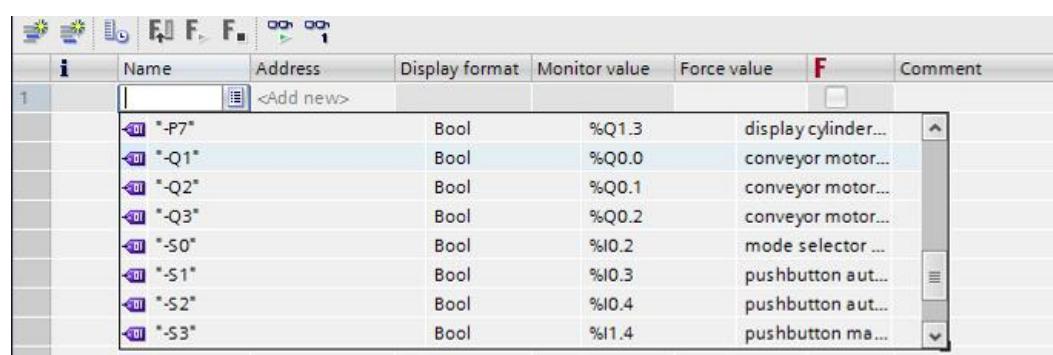
→ To force tags, you must first double-click the force table to open it.

(→ Force table)



The screenshot shows the TIA Portal interface with the Project tree on the left. Under the 'Devices' tab, a project named '032_100_FC-Programming' is expanded, showing various sub-nodes like 'Add new device', 'CPU1516F [CPU 1516F-3 PN/DP]', 'Program blocks', 'Technology objects', and 'Watch and force tables'. Within 'Watch and force tables', the 'Force table' node is selected. On the right, a table titled 'Force table' is displayed with one row currently present. The table has columns for Name, Address, Display format, Monitor value, Force value, and Comment. The 'Name' column contains the value '1'. The 'Address' column contains '<Add new>'. The 'Force value' column contains a green checkmark icon.

→ Select the “Q1” operand with address %Q0.0 from the list. (→ Q1)



The screenshot shows the 'Force table' dialog box. The table lists eight operands, each with a checkbox in the 'Force value' column. The rows are as follows:

	Name	Address	Display format	Monitor value	Force value	Comment
1	<Add new>					
	"-P7"	Bool	%Q1.3	display cylinder...	<input checked="" type="checkbox"/>	
	"-Q1"	Bool	%Q0.0	conveyor motor...	<input checked="" type="checkbox"/>	
	"-Q2"	Bool	%Q0.1	conveyor motor...	<input checked="" type="checkbox"/>	
	"-Q3"	Bool	%Q0.2	conveyor motor...	<input checked="" type="checkbox"/>	
	"-S0"	Bool	%I0.2	mode selector ...	<input checked="" type="checkbox"/>	
	"-S1"	Bool	%I0.3	pushbutton aut...	<input checked="" type="checkbox"/>	
	"-S2"	Bool	%I0.4	pushbutton aut...	<input checked="" type="checkbox"/>	
	"-S3"	Bool	%I1.4	pushbutton ma...	<input checked="" type="checkbox"/>	

→ With forcing, the operands are entered with direct IO access (%Q0.0:P).

	i	Name	Address	Display format	Monitor value	Force value	F	Comment
1		*-Q1*:P	%Q0.0:P	Bool	0:0		<input type="checkbox"/>	
2			<Add new>				<input checked="" type="checkbox"/>	

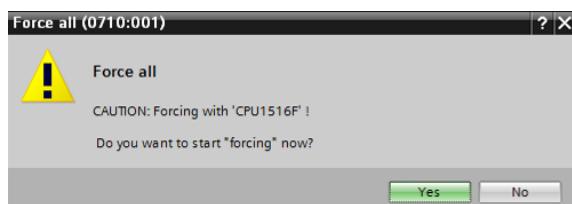
→ Enter the desired force value and activate it .

Click "Start or replace forcing" . The new force request is transferred to the CPU.

(→ %Q0.0:P → TRUE →  →  Start or replace forcing)

	i	Name	Address	Display format	Monitor value	Force value	F	Comment
1		*-Q1*:P	%Q0.0:P	Bool	0:0	TRUE	<input checked="" type="checkbox"/>	
2			<Add new>				<input type="checkbox"/>	

→ Confirm the warning with 'Yes'. (→ Yes)



→ Forcing is activated and the yellow **MAINT LED** on the CPU lights up. In addition, an **F** on a red background is shown at the top right of the display of the S7-1500.

	i	Name	Address	Display format	Monitor value	Force value	F	Comment
1		*-Q1*:P	%Q0.0:P	Bool	0:0	TRUE	<input checked="" type="checkbox"/>	

Note: If the watch table is closed or the connection to the PLC is lost, **forcing remains active**, and the yellow **FRCE LED** on the CPU continues to be lit.

→ If you want to '**Stop forcing**', simply click "Stop forcing" , and confirm the next dialog with "Yes".

(→  Stop forcing) 'Yes' (→ Yes)

	i	Name	Address	Display format	Monitor value	Force value	F	Comment
1		*-Q1*:P	%Q0.0:P	Bool	0:0	TRUE	<input checked="" type="checkbox"/>	
2			<Add new>				<input type="checkbox"/>	

Forcing is stopped and the yellow **MAINT LED** on the CPU goes out.

→ If there is already a force request in the controller, this is indicated by the  symbol in the watch table.

i	Name	Address	Display form..	Monitor val..	Monitor with trig..	Modify with trigge..	Modify v...
1	"-S0"	%I0.2	Bool	FALSE	Permanent	Permanent	
2	"-S3"	%I1.4	Bool	FALSE	Permanent	Permanent	
3	"-K0"	%I0.1	Bool	FALSE	Permanent	Permanent	
4	"-B1"	%I0.5	Bool	FALSE	Permanent	Permanent	
5	"-S4"	%I1.5	Bool	FALSE	Permanent	Permanent	
6	"-A1"	%I0.0	Bool	FALSE	Permanent	Permanent	
7		%Q0.0	Bool	FALSE	Permanent	Permanent	
8							

→ If you now click , additional information is displayed. (→)

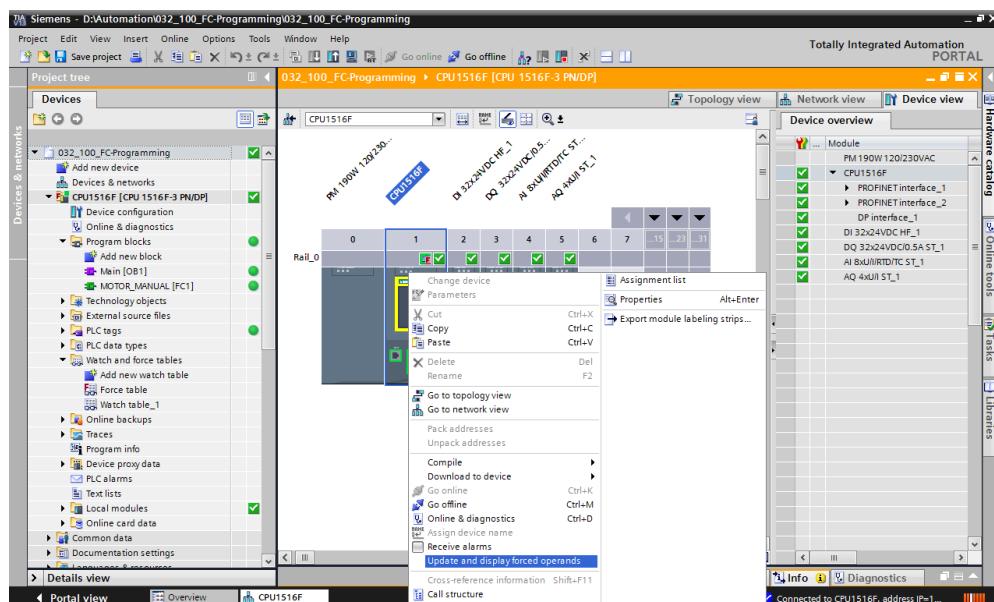
i	Name	Address	Display form..	Monitor val..	Monitor with trig..	Modify with trigge..	Modify v...
1	"-S0"	%I0.2	Bool	FALSE	Permanent	Permanent	
2	"-S3"	%I1.4	Bool	FALSE	Permanent	Permanent	
3	"-K0"	%I0.1	Bool	FALSE	Permanent	Permanent	
4	"-B1"	%I0.5	Bool	FALSE	Permanent	Permanent	
5	"-S4"	%I1.5	Bool	FALSE	Permanent	Permanent	
6	"-A1"	%I0.0	Bool	FALSE	Permanent	Permanent	
7		%Q0.0	Bool	FALSE	Permanent	Permanent	
8				The I/O belonging to address "%Q0.0" will be forced with the value "TRUE".			

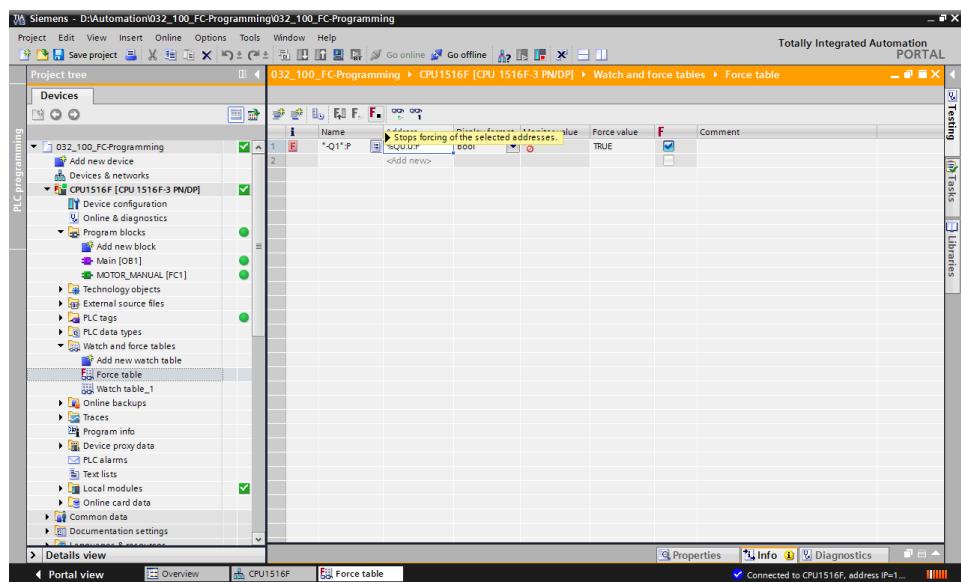
→ If there is already a force request in the controller, it can also be displayed and stopped via the online device view. For this, you need to right-click the CPU in online mode of the device view and select "Update and display forced operands".

(→ right-click the CPU → Update and display forced operands")

→ The force table with the current force requests will now be displayed and you can stop

these. (→ Stop forcing)





7.8 Checklist

No.	Description	Completed
1	Project 032-100_FB-programming... successfully retrieved.	
2	CPU 1516F from project 032-100_FB-Programming... successfully downloaded.	
3	CPU 1516F connected online.	
4	Status of the CPU 1516F checked with Online & diagnostics.	
5	Offline/online comparison of blocks in the CPU 1516F performed.	
6	Watch_table_1 created.	
7	Tags (-S0 / -S3 / -K0 / -B1 / -S4 / -A1 / -Q1) entered in watch table.	
8	Switch on conveyor motor forwards by modifying the output (-Q1 = 1) in watch table.	
9	Switch off conveyor motor forwards by modifying the output (-Q1 = 0) in watch table.	
10	Open force table	
11	Tag (-Q1:P) entered in force table.	
12	Switch on conveyor motor forwards by forcing the output (-Q1 = 1) in force table.	
13	Force output -Q1 to switch off again.	



Training Curriculums

Siemens Automation Cooperates with Education | 05/2017

TIA Portal Module 008

Diagnostics via the Web
with SIMATIC S7-1500

Table of contents

1	Goal.....	258
2	Prerequisite.....	258
3	Required hardware and software	258
4	Theory	259
4.1	System diagnostics: Automated creation of error messages.....	259
4.2	Diagnostics via web server	259
4.3	Diagnostics with the integrated display.....	260
5	Task.....	261
6	Planning.....	261
7	Structured step-by-step instructions.....	261
7.1	Retrieve an existing project.....	261
7.2	Configure the web server.....	262
7.3	Configure the display	265
7.4	Configure system diagnostics	266
7.5	Activate the diagnostics of the power supply for the AQ module and download the PLC	267
7.6	Trigger error message	269
7.7	Display alarms in Online & diagnostics.....	270
7.8	Diagnostics for the S7-1500 via the web	271
7.9	Diagnostics for the S7-1500 via the integrated display	279
7.10	Checklist.....	280

WEB SERVER AND ADVANCED DIAGNOSTICS

1 Goal

In this module, the reader will become acquainted with additional tools that are helpful for troubleshooting.

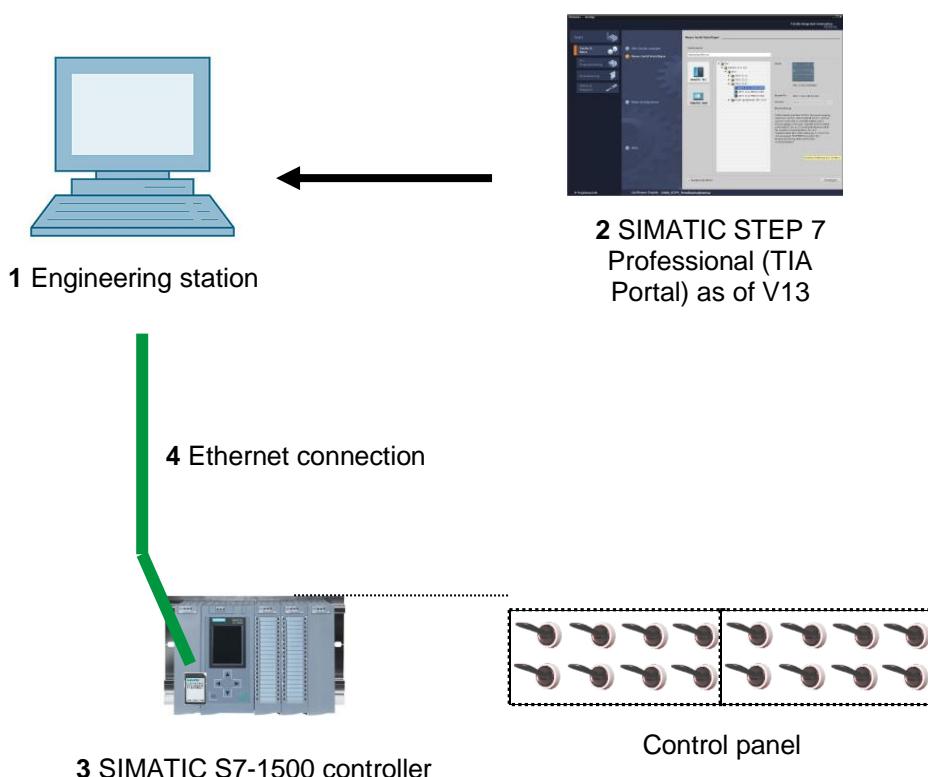
In particular, we will show you how automated alarm texts can be generated in the TIA Portal for hardware faults and system errors. These can then be displayed not only in the TIA Portal but also on the display of the CPU as well as via the web server of the CPU 1516F-3 PN/DP. It is also possible to bring these into the message windows of HMI systems for viewing.

2 Prerequisite

This chapter builds on the hardware configuration of the SIMATIC S7 CPU1516F-3 PN/DP. However, other hardware configurations can be used.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V13
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 System diagnostics: Automated creation of error messages

In the TIA Portal, the diagnostics of devices and modules is collectively referred to as system diagnostics. The monitoring functions are automatically derived from the hardware configuration.

All SIMATIC products have integrated diagnostic functions which you can use to detect and remedy faults. The components automatically signal a possible disruption of operation and provide additional detailed information. Undesired downtimes can be minimized with plant-wide diagnostics.

The following states are monitored by the system in the running plant:

- Device failure
- Pull/plug error
- Module fault
- IO access error
- Channel fault
- Parameter assignment error
- Failure of the external auxiliary voltage

4.2 Diagnostics via web server

The web server enables monitoring and administering of the CPU by authorized users over a network.

This permits evaluation and diagnostics over long distances. Monitoring and evaluation is possible without the TIA Portal; all you need is a web browser.

The web server is deactivated in the delivery state of the CPU. This means that you must load a project in which the web server is activated to enable access using the web browser.

The web server offers the following security functions:

- Access via secure "https" transmission protocol
- User authorization by means of a user list
- Restriction of access from certain interfaces

You need a web browser to access the HTML pages of the CPU.

The following web browsers have been tested for communication with the CPU:

- Internet Explorer (Version 8)
- Mozilla Firefox (Version 21)
- Mobile Safari (iOS5)

The screenshot shows a web-based interface for a TIA Portal module. On the left is a sidebar with navigation links: Start page, Diagnostics, Diagnostic Buffer, Module information, Alarms (which is selected and highlighted in blue), Communication, Topology, Tag status, Watch tables, and Customer pages. The main content area is titled 'Alarms' and shows a single entry for 'entries 1-50'. The table has columns: AlarmNr., Date, Time, Alarm text, State, and Acknowledgement. The entry details are: AlarmNr. 34, Date 01/01/2012, Time 12:25:02.177 am, Alarm text 'Error: Supply voltage missing on Q0 CPU1516F/AQ4xUI ST_1.', State 'incoming', and Acknowledgement is not yet done. Below the table, a 'Details on alarm number: 34' section shows 'Short name: AQ 4xUI ST Order number: 6ES7 532-5HD00-0AB0'. At the bottom of the content area, it says 'Incoming event'.

Figure 1: Web server of the CPU 1516F-3 PN/DP with alarm text from the system diagnostics

Note: Make sure that you protect the CPU from manipulation and unauthorized access through the use of different methods (e.g., limiting network access, using firewalls).

4.3 Diagnostics with the integrated display

The S7-1500 CPU has a front flap with a display and control keys. Control data and status data can be displayed in various menus on the display and numerous settings can be made. You use the control keys to navigate through the menus.

The display of the CPU offers the following functions:

- 6 different display languages can be selected.
- Diagnostic messages are displayed in plain text.
- The interface settings can be changed locally.
- Password assignment for display operation is possible through the TIA Portal.



Figure 2: Display of the CPU 1516F-3 PN/DP with alarm text from the system diagnostics

5 Task

The following advanced diagnostic functions will be shown and tested in this chapter:

- Configuration of web server of the CPU 1516F-3 PN/DP
- Configuration of display of the CPU 1516F-3 PN/DP
- Create hardware fault and system error alarms with the system diagnostics
- Display alarms via the web server of the CPU 1516F-3 PN/DP
- Display alarms via the integrated display of the CPU 1516F-3 PN/DP

6 Planning

The diagnostic functions will be performed using a finished project as an example. A project in the TIA Portal that was previously downloaded to the controller should be open for this. In our case, once you have started the TIA Portal, you will retrieve a previously created project that was archived and download it to the associated controller.

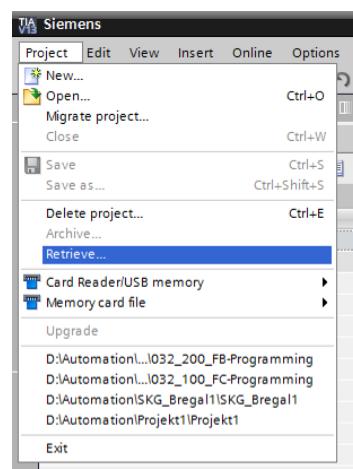
You can then configure the web server, the display and the system diagnostics in the TIA Portal. To test the system diagnostics, we will disconnect the monitored analog output module from its supply voltage.

7 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

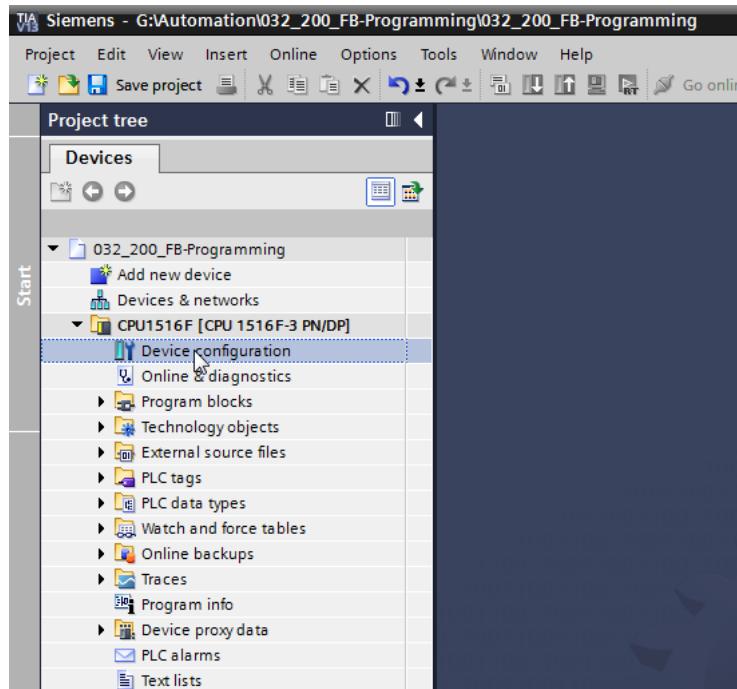
- Before we begin with diagnostics via the web server, we need a project from the SCE_EN_032-410 Basics_Diagnostics module.
(e.g., SCE_EN_032-410_Basics_Diagnostics_2_R1503.zap13)
- To retrieve an existing project that has been archived, you must select the relevant archive with →Project →Retrieve in the project view.
- Confirm your selection with "Open".
(→ Project → Retrieve → Select a .zap archive → Open)



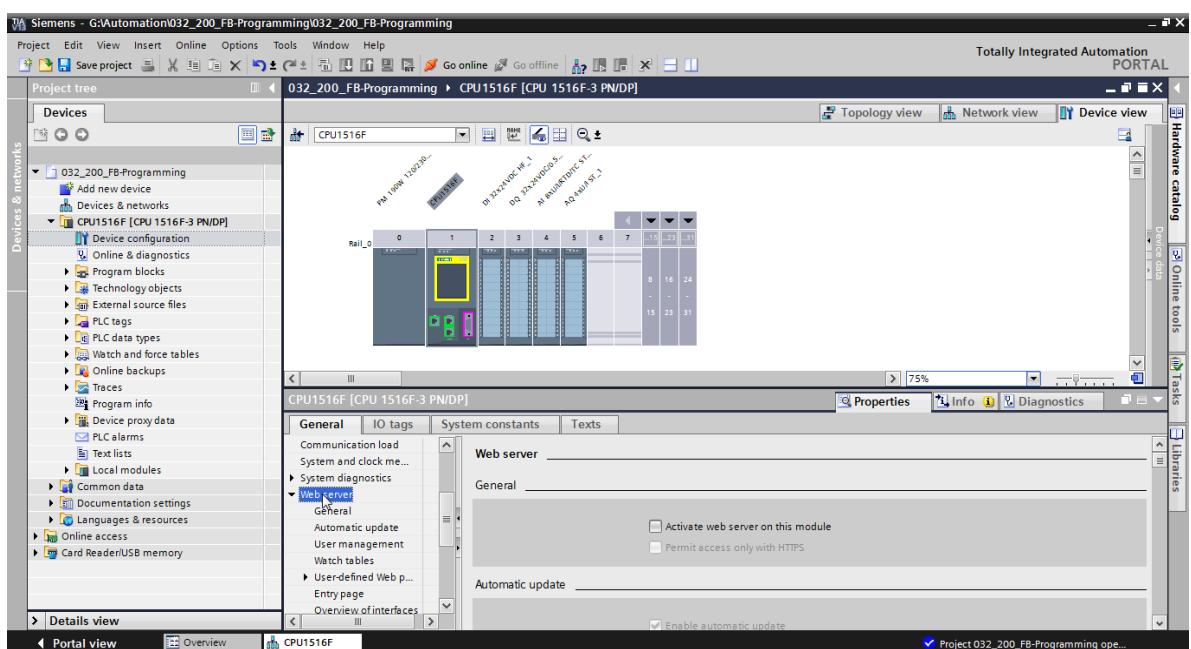
- The next step is to select the target directory where the retrieved project will be stored.
- Confirm your selection with "OK".
- (→ Target directory → OK)

7.2 Configure the web server

- To configure the web server, open the device configuration of the CPU 1516F-3 PN/DP.
- (→ CPU_1516F [CPU 1516F-3 PN/DP] → Device configuration)

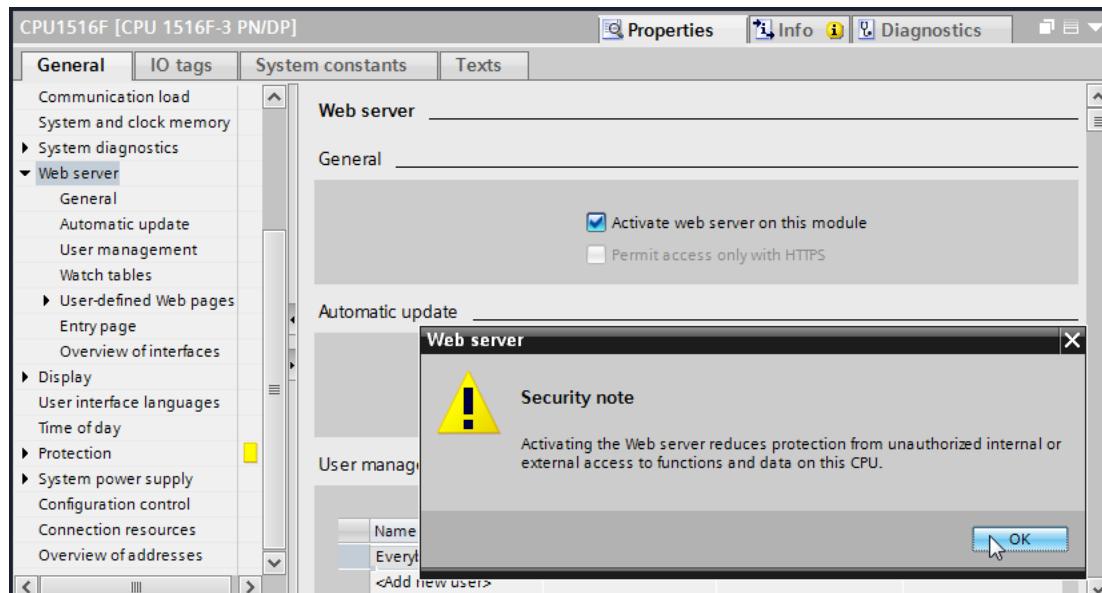


- Select the CPU and choose the 'Web server' menu item in the properties.
- (→ CPU_1516F → Properties → Web server)



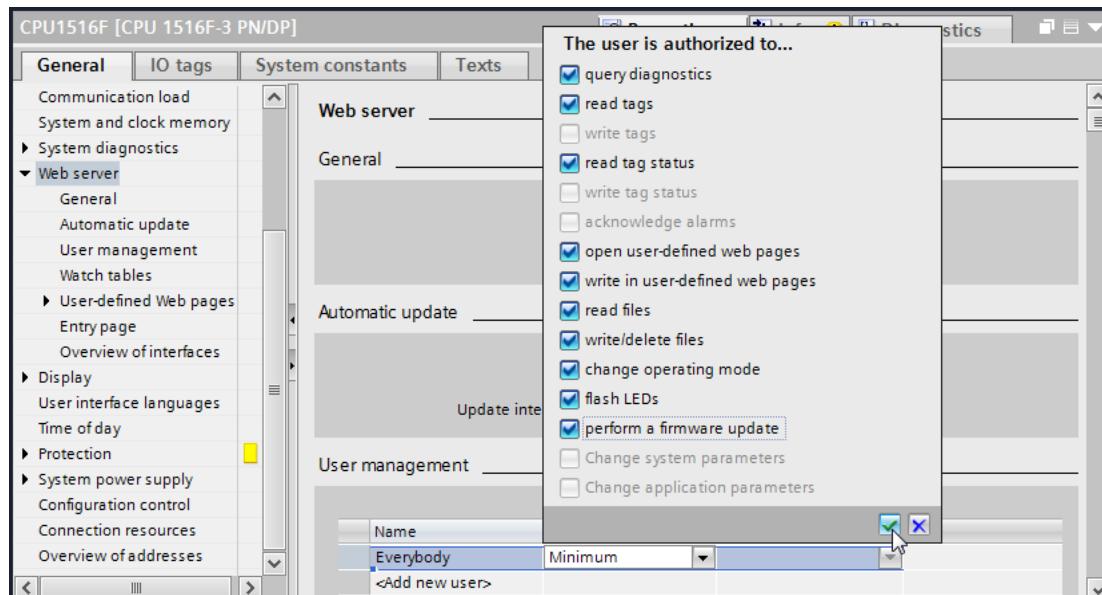
→ Activate the web server on this module and confirm the security note.

(→ Activate web server on this module → OK)



→ Leave the check mark for 'Enable automatic update', and select the security settings of the 'Everybody' user. Enable this user to carry out all possible actions and accept your settings.

(→ →)



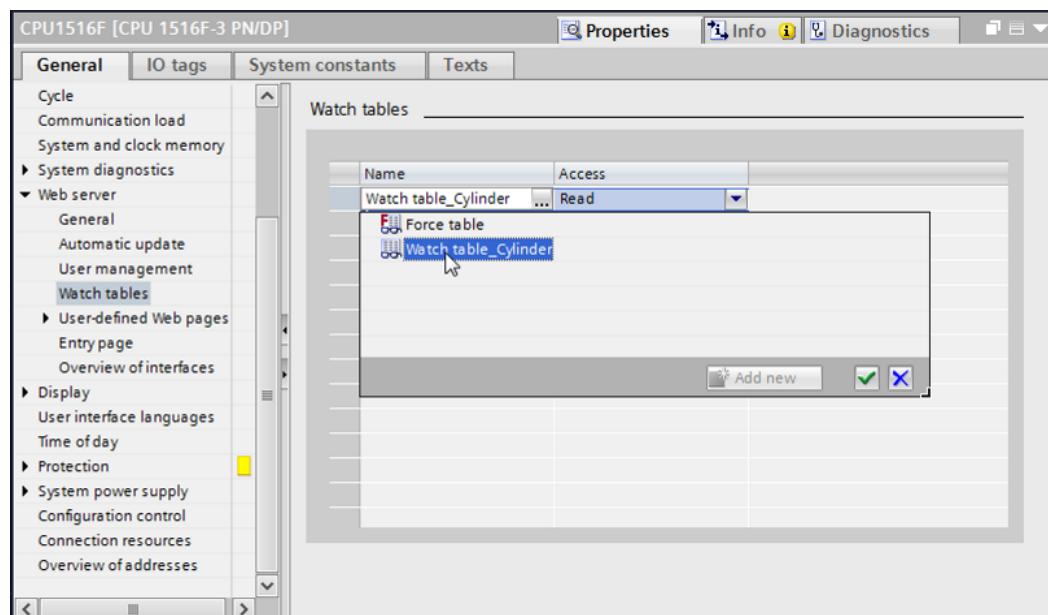
Notes: You can also create multiple users here with different authorizations. These users then require a password.

- As a result of these authorizations, the 'Everybody' user is now automatically assigned the access level 'Administrative'.

Name	Access level	Password
Everybody	Administrative	
<Add new user>		

- In the 'Watch tables' menu item, the 'Watch table_Cylinder' can now be entered in the web server.

(→ Watch table_Cylinder → 

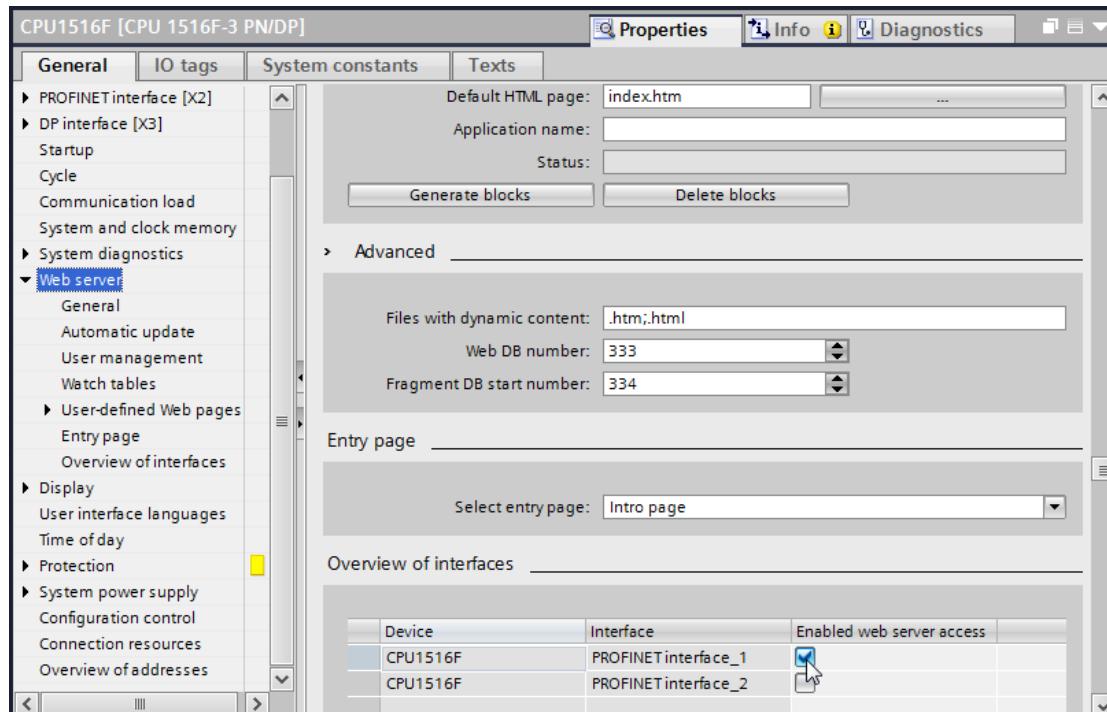


Name	Access
Watch table_Cylinder	Read

- Accessing is read-only. (→ Read)

Name	Access
Watch table_Cylinder	Read
<Add new watch table>	Read

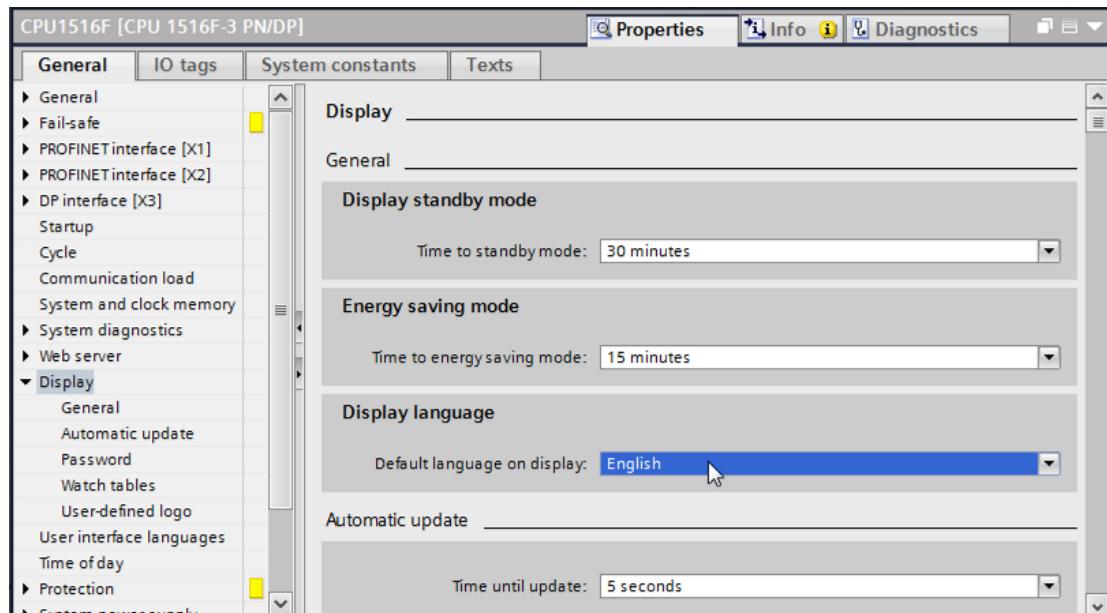
- User-defined web pages will not be created here. For reasons of plant safety / security, we will enable only PROFINET interface_1 for access to the web server.
 (→ Enabled web server access → PROFINET interface_1)



7.3 Configure the display

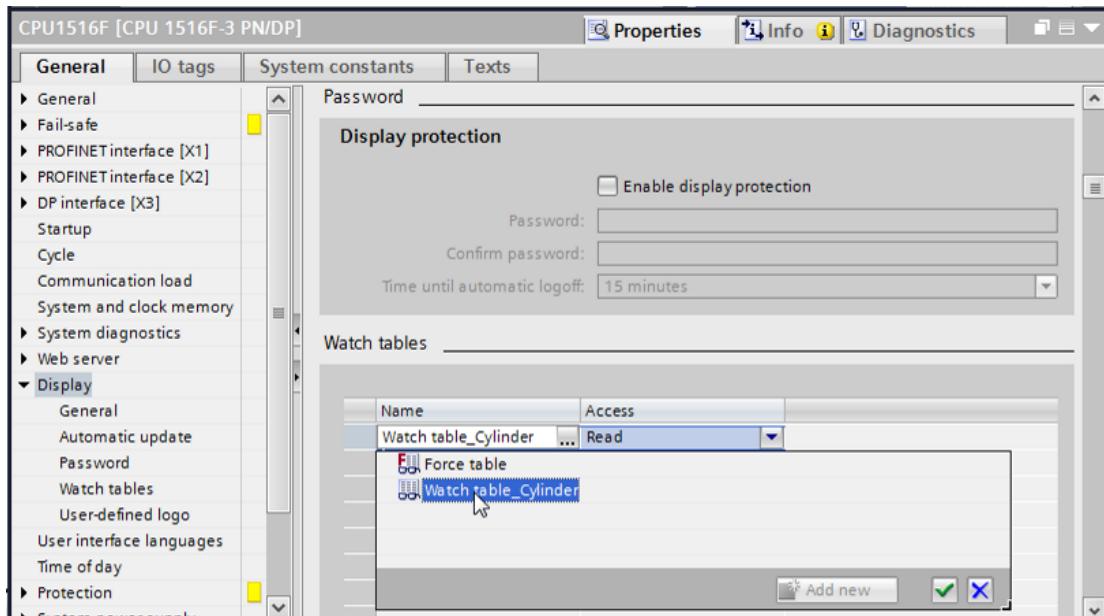
- The settings for the display of diagnostics data can also be changed on the integrated display of the CPU 1516F-3 PN/DP. First, the general settings are selected as shown here.

(→ Display → General)

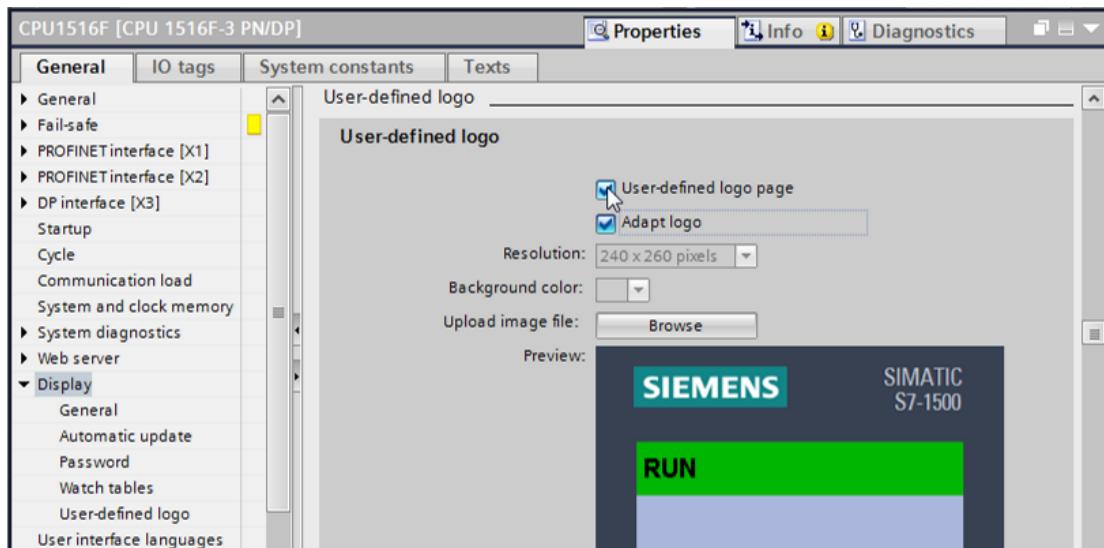


- In the 'Watch tables' menu item, the 'Watch table_Cylinder' can now be entered in the display.

(→ Watch table_Cylinder → 

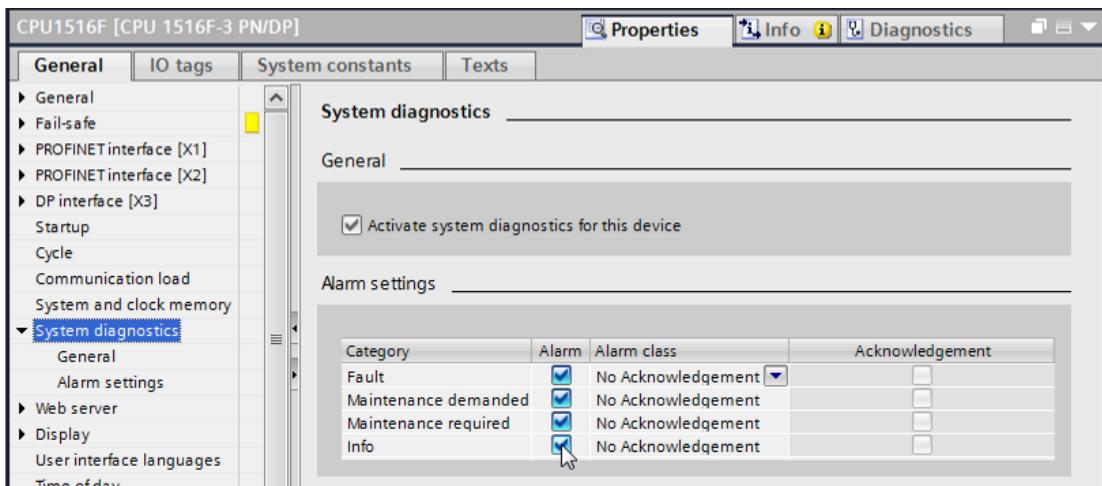


- If desired, a user-defined logo can also be shown on the display
 (→ User-defined logo page)



7.4 Configure system diagnostics

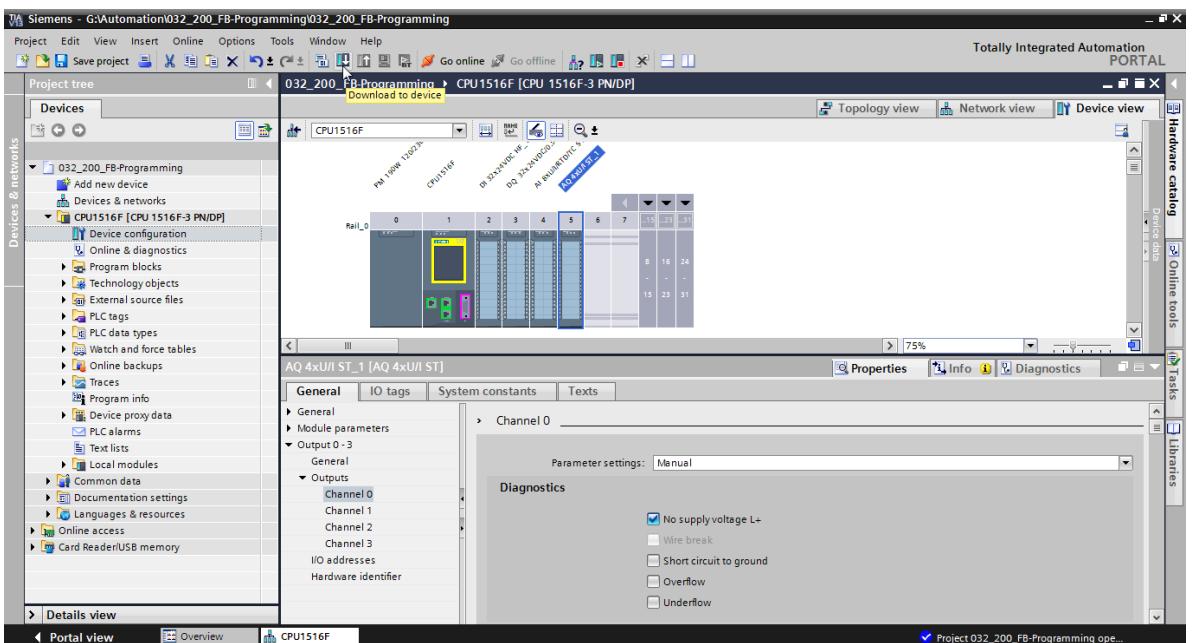
- An important function for effective troubleshooting is the integrated system diagnostics. This is always activated for the SIMATIC S7-1500. The alarm categories can be selected in the alarm settings and, if desired, an 'Acknowledgment' can be specified.



Notes: The indicated alarm class is important so that it can be selected in the alarm windows of the operator panel (e.g., TP1500, TP700, etc.).

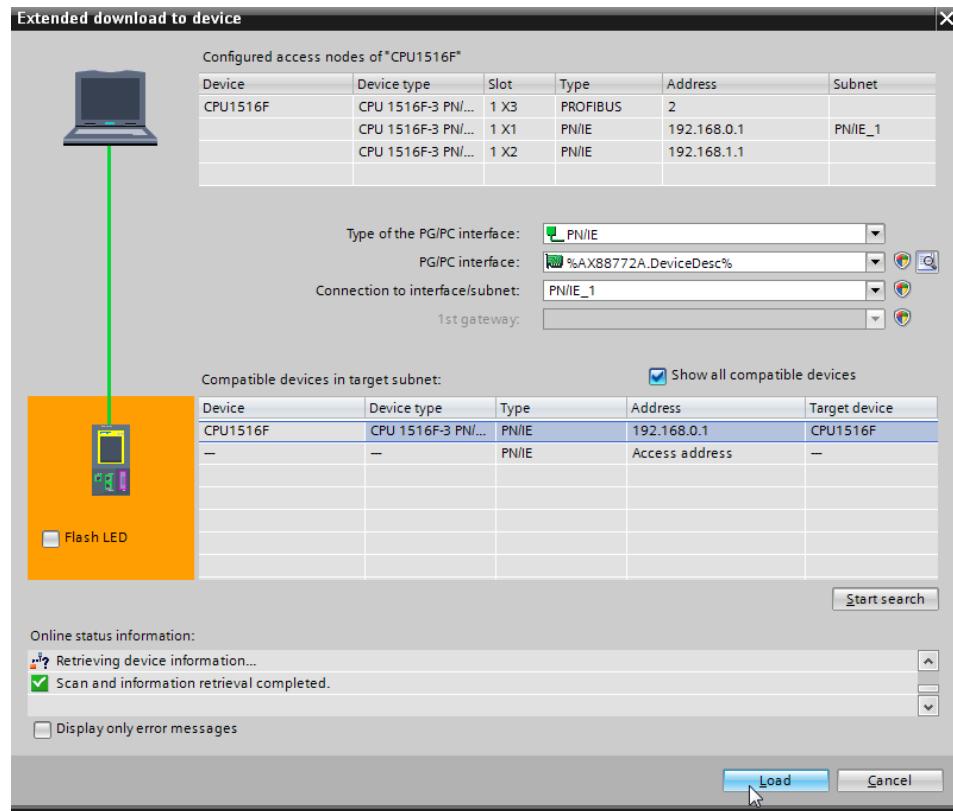
7.5 Activate the diagnostics of the power supply for the analog output module and download the PLC

- Once the web server, display and system diagnostics have been configured in the controller, we also activate the diagnostics for the supply voltage for the analog output module. The controller can then be selected and downloaded together with the created program.
- (→ Device configuration → AQ 4xU/I ST_1 → Output 0 – 3 → Outputs → Channel 0 → Diagnostics → No supply voltage L+ → CPU_1516F [CPU 1516F-3 PN/DP] →

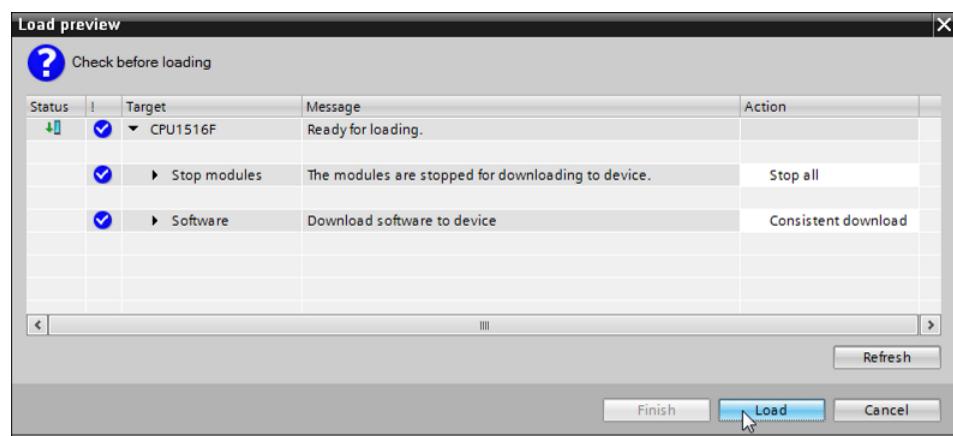


- Select the correct interface and click 'Start search'.
 (→ PN/IE → Selection of the network adapter of the PG/PC → Direct at slot '1 X1' → Start search)

Once "Scan and information retrieval completed" appears, click 'Load'.
 (→ Load)

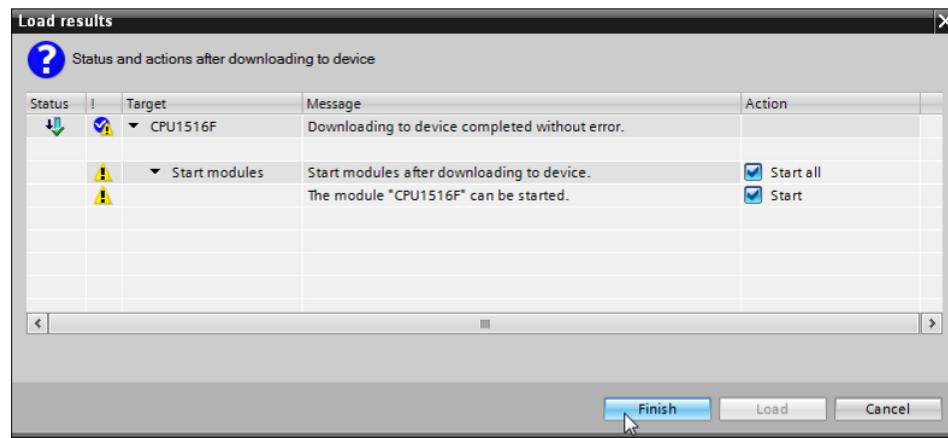


- Before downloading can be started, other actions may have to be selected. Click 'Load' again.
 (→ override all → Load)



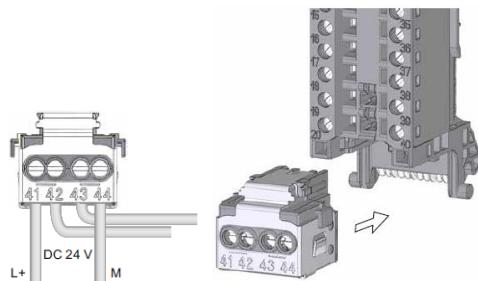
→ After loading, first select the "Start all" check box and click 'Finish'.

(→ Start all → Finish)



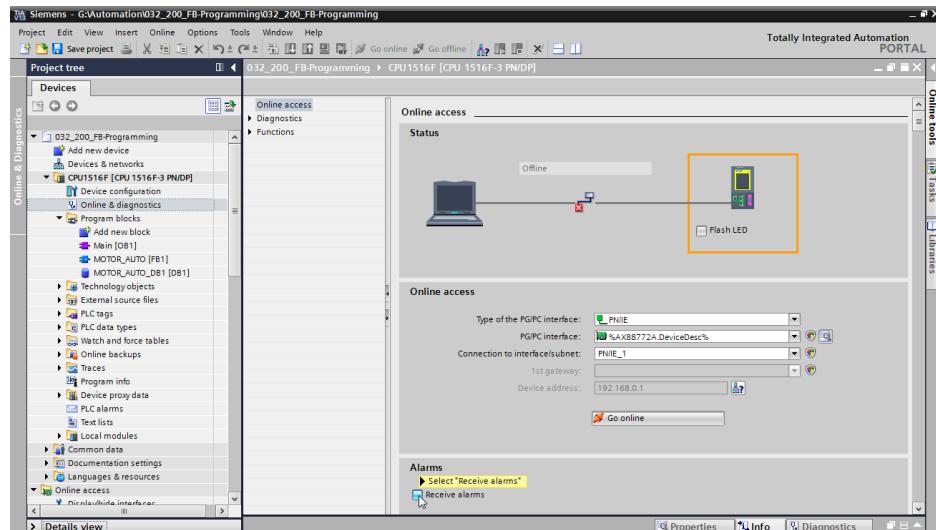
7.6 Trigger error message

→ The power supply of the analog output module is via terminals 41-44 of the supply element. Remove this supply element, as shown here, from the front connector to trigger an error message. Result: the red ERROR LED on the CPU is lit and an error message is triggered. The following pages describe where and how you can view this error message.

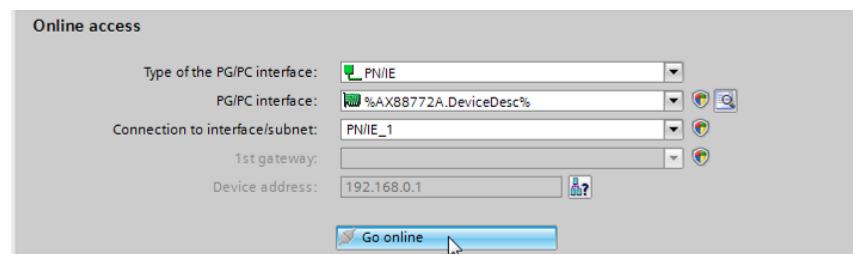


7.7 Display alarms in Online & diagnostics

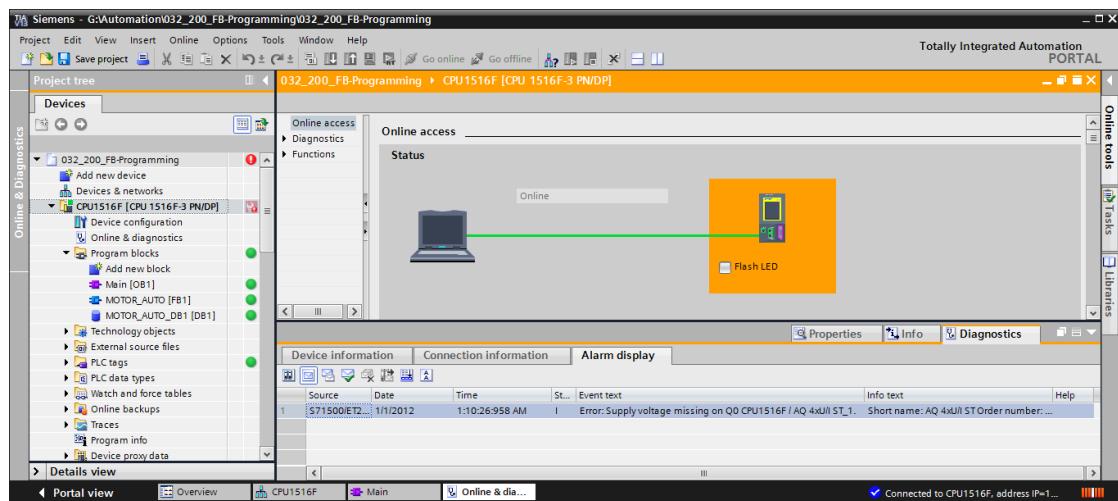
- To get started with the diagnostic functions, we will select our controller 'CPU_1516F' and click 'Online & diagnostics'. Under 'Online access' select 'Receive alarms' for the 'Alarms' item. (→ CPU_1516F → Online & diagnostics → Online access → Alarms → Receive alarms)



- Select the correct interface and click 'Go online'.
(→ Go online)



- The error message can now be checked in the 'Alarm display' under 'Diagnostics'.
(→ Diagnostics → Alarm display)

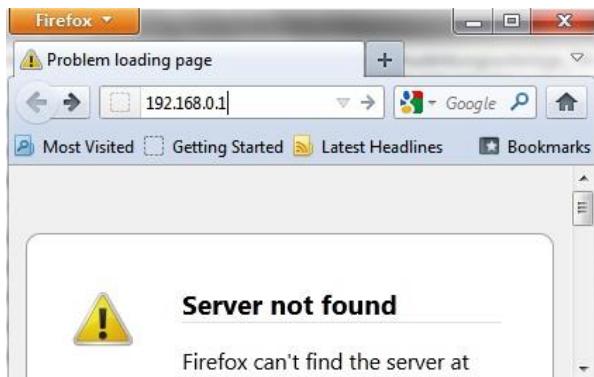


7.8 Diagnostics for the S7-1500 via the web

- To be able to access the Web server of the CPU 315F-2 PN/DP we open any Web browser on a PC that is connected to the CPU via TCP/IP.



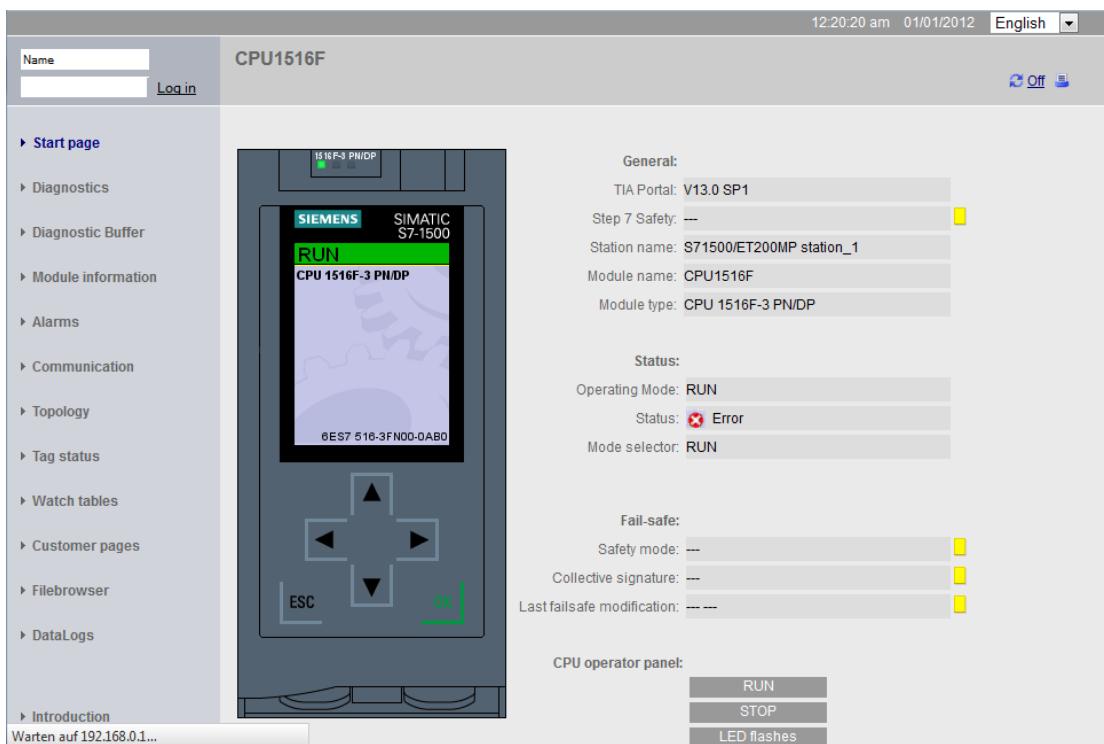
- There we enter the IP address of the CPU 1516F-3 PN/DP. (→ 192.168.0.1)



- On the displayed web page, we first select the language and then click 'ENTER'.
(→ English → ENTER)

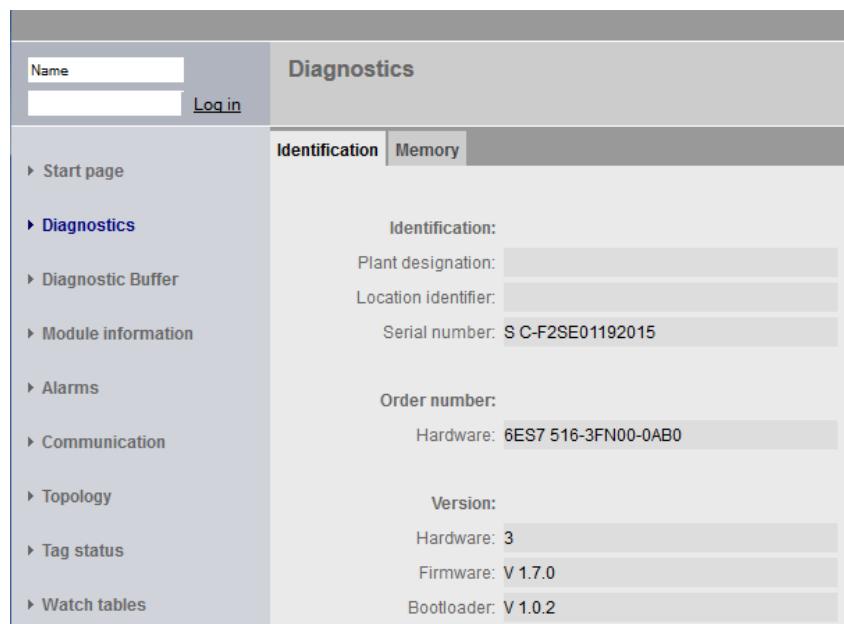


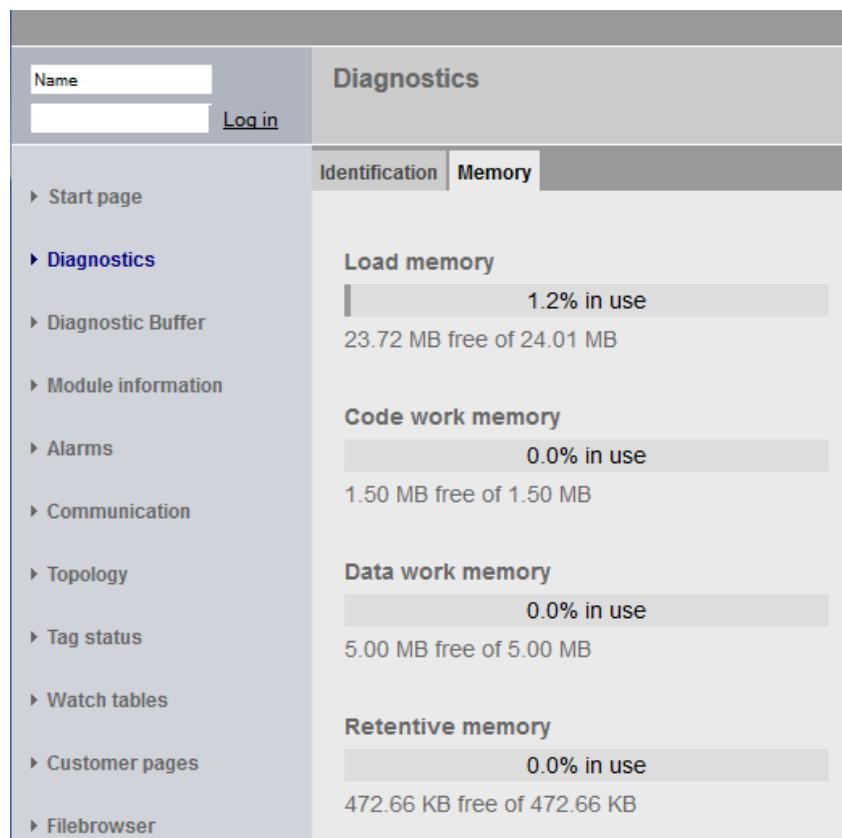
- On the '**Home Page**' we see general information about the PLC and its status.
(→ Home Page)



→ Hardware, Firmware Version and Serial number are displayed besides other information under 'Diagnostics'.

(→ Diagnostics)





- Under 'Diagnostics Buffer' we see descriptive information for all events in the CPU. Event information is recorded in a circular buffer. The most recent alarm is displayed in the top line.

(→ Diagnostics Buffer)

The screenshot shows the 'Diagnostic Buffer' table with the following data:

	Number	Time	Date	State	Event
Start page	1	12:25:06.003 am	01/01/2012	incoming event	Communication initiated request: WARM RESTART Pending startup inhibit - No startup inhibit set - CPU changes from STARTUP to RUN mode
Diagnostics	2	12:25:05.982 am	01/01/2012	incoming event	Communication initiated request: WARM RESTART Pending startup inhibit - No startup inhibit set - CPU changes from STOP to STARTUP mode
Diagnostic Buffer	3	12:25:02.177 am	01/01/2012	Incoming event	Supply voltage missing
Module information	4	12:25:01.475 am	01/01/2012	outgoing event	Supply voltage missing
Alarms	5	12:25:01.389 am	01/01/2012	incoming event	Communication initiated request: STOP Pending startup inhibit(s): - No startup inhibit set - CPU changes from RUN to STOP mode
Communication	6	12:23:51.030 am	01/01/2012	incoming event	Supply voltage missing
Topology	7	12:23:46.084 am	01/01/2012	outgoing event	Follow-on operating mode change
Tag status	8	12:19:21.717 am	01/01/2012	incoming event	Power-on mode set: WARM RESTART to RUN (if CPU was in RUN before Pending startup inhibit(s): - No startup inhibit set - CPU changes from STARTUP to RUN mode) Follow-on operating mode change
Watch tables					Event ID: 16# 08:0011
Customer pages					Details: 3 Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1.
					incoming event

- The status of the individual modules of our SIMATIC S7-1500 is displayed with additional details in the 'Module Information' view.

(→ Module Information)

Slot	State	Name	Order number	I address	Q address	Comment
1	<input checked="" type="checkbox"/>	CPU1516F	6ES7 516-3FN00-0AB0			
2	<input checked="" type="checkbox"/>	DI 32x24VDC HF_1	6ES7 521-1BL00-0AB0	0		
3	<input checked="" type="checkbox"/>	DQ 32x24VDC0.5A ST_1	6ES7 522-1BL00-0AB0		0	
4	<input checked="" type="checkbox"/>	AI 8xI/I/RTD/TC ST_1	6ES7 531-7KF00-0AB0	64		
5	<input checked="" type="checkbox"/>	AQ 4xUI ST_1	6ES7 532-5HD00-0AB0		64	

Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1.

- The alarm texts generated in the CPU 1516F-3 PN/DP are available in 'Alarms'.
 (→ Alarms)

AlarmNr	Date	Time	Alarm text	State	Acknowledgement
34	01/01/2012	12:25:02.177 am	Error: Supply voltage missing on Q0 CPU1516F / AQ 4xUI ST_1.	incoming	

Details on alarm number: 34
 Short name: AQ 4xUI ST Order number: 6ES7 532-5HD00-0AB0
 Incoming event

Note: Here we see the failure of the supply voltage for the digital input module with activated diagnostic error interrupt.

- Details about communication settings and communication errors are displayed under 'Communication'.

PROFINET Interface [X1]:

Network connection:

- MAC address: 28-63-36-87-F3-05
- Name: cpu1516f.profinet interface_1

IP parameter:

- IP Address: 192.168.0.1
- Subnet mask: 255.255.255.0
- Default router: --
- IP settings: IP address set in project

Physical properties:

Port number	Link status	Settings	Mode	Connection medium
X1 P1	OK	--	100 MBit/s full-duplex	Copper cable
X1 P2	disconnected	--	--	Copper cable

Communication

[Log in](#)

Parameter Statistics Resources Connections

- ▶ Start page
- ▶ Diagnostics
- ▶ Diagnostic Buffer
- ▶ Module information
- ▶ Alarms
- ▶ **Communication**
- ▶ Topology
- ▶ Tag status
- ▶ Watch tables
- ▶ Customer pages
- ▶ Filebrowser
- ▶ DataLogs
- ▶ Introduction

Total statistics

Sent data packages:
Sent without errors: 3243312 Bytes

Collision during sending attempt: 0
Canceled due to other errors: 0

Received data packages:
Received without errors: 755370 Bytes

Rejected due to error: 0
Rejected due to resource bottleneck: 0

Statistics X1 P1

Sent data packages:
Sent without errors: 3242928 Bytes

Collision during sending attempt: 0
Canceled due to other errors: 0

Received data packages:
Received without errors: 755370 Bytes

Rejected due to error: 0
Rejected due to resource bottleneck: 0

Resources

Number of connections:
Maximum connections: 256
Connections not in use: 250

Connections:	reserved	in use
ES communication	4	0
HMI communication	4	0
S7 communication	0	0
OpenUser communication	0	0
Web communication	2	6
Other communication	--	0

Connections

State	Local ID (Hex)	Slot of Gateway	Remote address type	Remote address	Type	Type
Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB
Connection is established	0	---	IPv4	192.168.0.108	Adhoc	WEB

→ Devices that are connected to the individual ports of the CPU 1516F-3 PN/DP and the addresses of these devices can be displayed under 'Topology'. There are various views for this. In the case of larger network structures, the entire network structure of a plant can be displayed and faulty connections shown in the status, provided this function is supported by the individual components.

(→ Topology)

The figure consists of three vertically stacked screenshots of the TIA Portal software interface, specifically the 'Topology' view.

Screenshot 1: Graphic View

- The top navigation bar shows 'Name' and 'Log in'.
- The main menu on the left includes 'Start page', 'Diagnostics', 'Diagnostic Buffer', 'Module information', 'Alarms', 'Communication', and 'Topology' (which is selected).
- The right panel displays a network topology diagram. A central node labeled 'cpu1516f S71500/ET20...' has two ports, P1 and P2. Port P1 is connected to another node labeled 'svensons'. A red exclamation mark icon is visible near the connection point.
- Below the diagram are tabs for 'Graphic view', 'Table view', and 'Status overview'.

Screenshot 2: Table View

- The top navigation bar shows the date and time '12:34:58 am 01/01/2012 English'.
- The main menu on the left includes 'Start page', 'Diagnostics', 'Diagnostic Buffer', 'Module information', 'Alarms', 'Communication', and 'Topology'.
- The right panel displays a table titled 'Topology' showing network connections.

Port State	Name	Module type	Port	Partner port Name	Port
	cpu1516f	S71500/ET200MP station	port-001	svensons	port-001
			port-002		
	svensons		port-001	cpu1516f	port-001

Screenshot 3: Status Overview

- The top navigation bar shows 'Name' and 'Log in'.
- The main menu on the left includes 'Start page', 'Diagnostics', 'Diagnostic Buffer', 'Module information', 'Alarms', 'Communication', and 'Topology'.
- The right panel displays a single node labeled 'cpu1516f S71500/ET200...' with a red exclamation mark icon.
- Below the node are tabs for 'Graphic view', 'Table view', and 'Status overview'.

→ Values of the individual tags can be displayed under 'Tag status'.

(→ Tag status)

The screenshot shows the 'Tag status' configuration page. On the left is a sidebar with navigation links: Start page, Diagnostics, Diagnostic Buffer, Module information, Alarms, Communication, Topology, Tag status (which is selected and highlighted in blue), and Watch tables. The main area has a title 'Tag status' and a subtitle 'Enter the address of a tag here which you want to monitor'. It contains a table with three columns: Address, Display format, and Value. There are three rows: -K0 (Bin, 2#0), -A1 (BOOL, FALSE), and New variable (empty). A 'Apply' button is at the bottom of the table.

Address	Display format	Value
-K0	Bin	2#0
-A1	BOOL	FALSE
New variable		

→ 'Tag tables' that are linked with the web server, such as the 'Watch table_Cylinder', can also be displayed.

(→ Tag tables → Watch table_Cylinder)

The screenshot shows the 'Watch tables' configuration page. The sidebar is identical to the previous one. The main area has a title 'Watch tables' and a dropdown menu showing 'Watch table_Cylinder' (which is selected and highlighted in blue) and 'Watch table_Cylinder'. Below the dropdown is a table with four columns: Name, Address, Format, and Value. There are three rows: "-B1" (%E0.5, BOOL, FALSE), "-B2" (%E0.6, BOOL, FALSE), and "-M2" (%A0.3, BOOL, FALSE).

Name	Address	Format	Value
"-B1"	%E0.5	BOOL	FALSE
"-B2"	%E0.6	BOOL	FALSE
"-M2"	%A0.3	BOOL	FALSE

- Individually created pages for the visualization and also for operator control of processes would be seen under 'Customer pages'. (→ Customer pages)

Customer pages

The page is not available.

- ▶ Start page
- ▶ Diagnostics
- ▶ Diagnostic Buffer
- ▶ Module information
- ▶ Alarms
- ▶ Communication
- ▶ Topology
- ▶ Tag status
- ▶ Watch tables
- ▶ Customer pages

- Data can be stored directly on the memory card in the CPU or loaded from there using the 'Filebrowser'. (→ Filebrowser)

Filebrowser

Name	Size	Changed	Delete	Rename
LOG	32768	12:25:42 pm 07/19/2015		
crdinfo.bin	512	12:25:42 pm 07/19/2015		

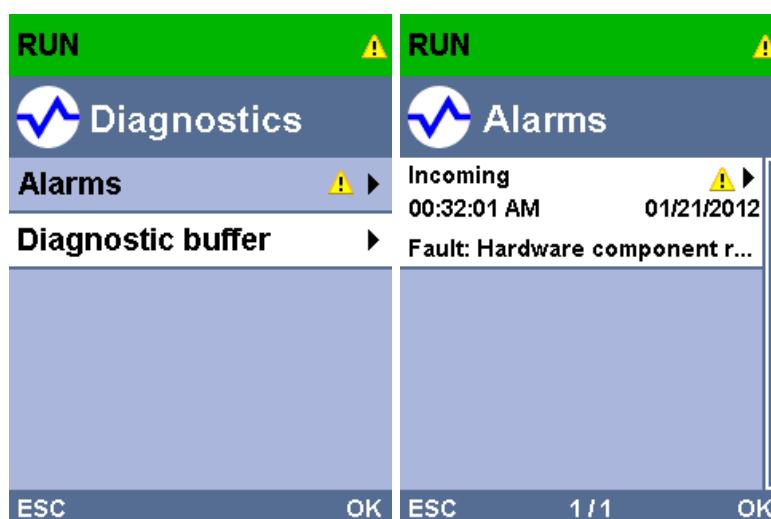
Directory operations:

- This means, for example, that you can read and edit the log files written by the CPU without having to use the TIA Portal. (→ DataLogs)

The screenshot shows the TIA Portal interface with the title bar "12:40:57 am 01/01/2012 English". On the left is a sidebar with links like Start page, Diagnostics, Diagnostic Buffer, etc., and "DataLogs" is selected. The main area is titled "DataLogs" and displays a table with one row: "Name" (No entries currently available), "Size", "Changed", and a "Retrieve and clear" button.

7.9 Diagnostics for the S7-1500 via the integrated display

- The user also has the ability to call up a variety of diagnostic information via the display. For example, the alarm texts generated by the system diagnostics can be displayed in the 'Diagnostics' menu under 'Alarms'.
 (→ Diagnostics → Alarms)



7.10 Checklist

No.	Description	Completed
1	Project 032-410_Basics_Diagnostics_2... successfully retrieved.	
2	Web server for the CPU 1516F from project 032-410_Basics Diagnostics_2... successfully configured.	
3	Display for the CPU 1516F from project 032-410_Basics Diagnostics_2... successfully configured.	
4	System diagnostics for the CPU 1516F from project 032-410_Basics Diagnostics_2... successfully configured.	
5	Diagnostics of the supply voltage for the analog output module activated.	
6	CPU 1516F from project 032-410_Basics Diagnostics_2... successfully downloaded.	
7	Power supply disconnected from analog output module.	
8	Display of alarm text from the system diagnostics in the alarm display of the TIA Portal.	
9	Display of the alarm text from the system diagnostics via the web server of the CPU 1516F.	
10	Display of the alarm text from the system diagnostics on the display of the CPU 1516F.	



Training Curriculum

TIA Portal Module 009

Analog Values
for SIMATIC S7-1500

Table of contents

1	Goal.....	283
2	Prerequisite.....	283
3	Required hardware and software.....	284
4	Theory	285
4.1	Analog signals	285
4.2	Measuring transducers	285
4.3	Analog modules – A/D converter.....	285
4.4	Data types of the SIMATIC S7-1500	286
4.5	Reading/outputting analog values	287
4.6	Normalizing analog values	288
5	Task.....	288
6	Planning.....	288
6.1	Analog control of the conveyor speed	288
6.2	Technology diagram	289
6.3	Reference list	289
7	Structured step-by-step instructions.....	290
7.1	Retrieve an existing project.....	290
7.2	Create the "MOTOR_SPEEDCONTROL" function	291
7.3	Configuration of the analog output channel	298
7.4	Expand the tag table to include analog signals.....	299
7.5	Call the block in the organization block	300
7.6	Save and compile the program	303
7.7	Download the program.....	303
7.8	Monitor program blocks	304
8	Checklist	306

ANALOG VALUES FOR SIMATIC S7-1500

1 Goal

In this chapter, you will become acquainted with the analog value processing of the SIMATIC S7-1500 with the TIA Portal programming tool.

The module explains the acquisition and processing of analog signals and gives a step-by-step description of read and write access to analog values in the SIMATIC S7-1500.

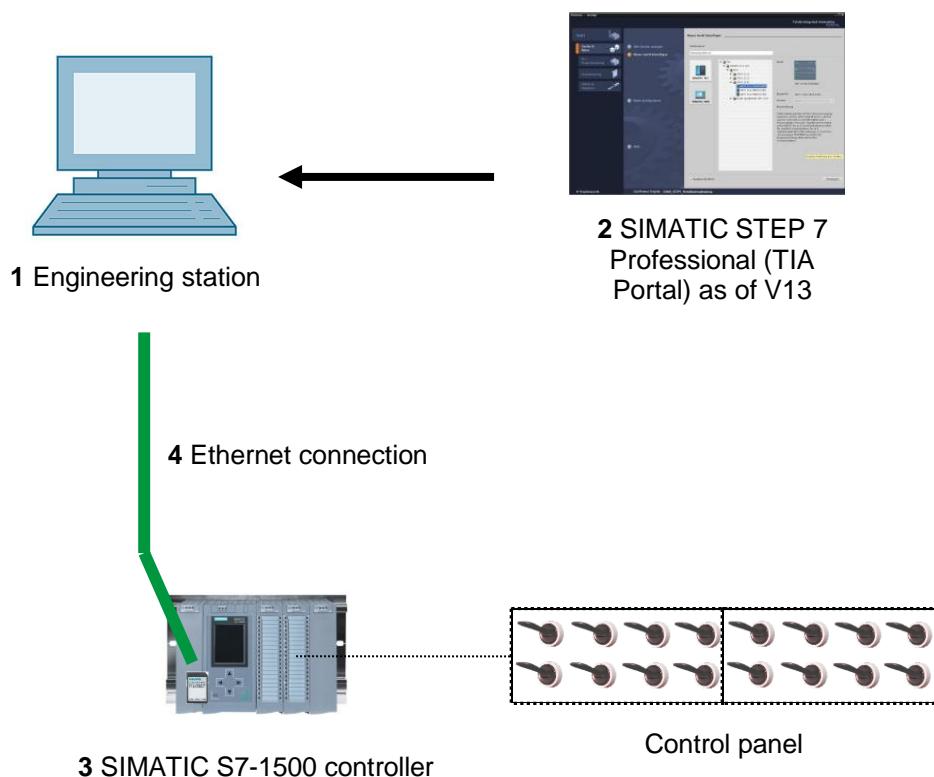
The SIMATIC S7 controllers listed in Chapter 3 can be used.

2 Prerequisite

This chapter builds on the chapter IEC Timers and Counters with the SIMATIC S7 CPU1516F-3 PN/DP. You can use the following project for this chapter, for example: 032-300 IEC Timers and Counters.zap13

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V13
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs and analog inputs and outputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 Analog signals

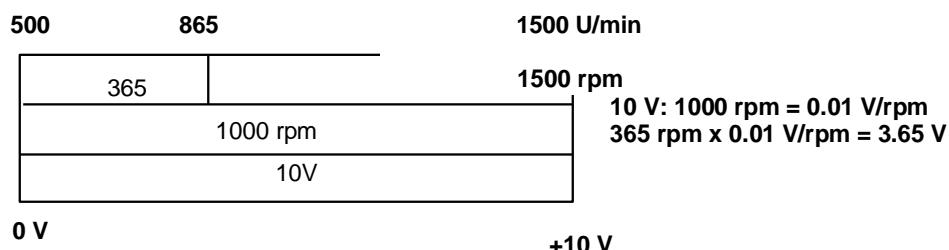
In contrast to a binary signal, which can assume only two signal states ("Voltage present +24 V" and "Voltage not present 0 V"), analog signals can assume any value within a defined range. A typical example of an analog sensor is a potentiometer. Depending on the position of the knob, any resistance can be set, up to the maximum value.

Examples of analog quantities in control engineering:

- Temperature -50 to +150 °C
- Flow rate 0 to 200 l/min
- Speed -500 to +50 rpm
- etc.

4.2 Measuring transducers

These quantities are converted to electrical voltages, currents or resistances with the help of a measuring transducer. If, for example, a speed is to be measured, the speed range of 500 to 1500 rpm can be converted to a voltage range of 0 to +10 V using a measuring transducer. At a measured speed of 865 rpm, the measuring transducer would output a voltage value of +3.65 V.



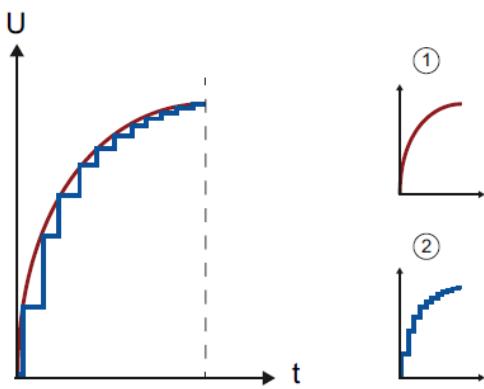
4.3 Analog modules – A/D converter

These electrical voltages, currents or resistances are then connected to an analog module that digitizes this signal for further processing in the PLC.

If analog quantities will be processed with a PLC, the read-in voltage, current or resistance value must be converted to digital information. The analog value is converted to a bit pattern. This conversion is referred to as analog-to-digital conversion (A/D conversion). This means, for example, that the voltage value of 3.65 V is stored as information in a series of binary digits.

The result of this conversion is always a 16-bit word for SIMATIC products. The integrated ADC (analog-to-digital converter) of the analog input module digitizes the analog signal being acquired and approximates its value in the form of a stepped curve. The most important parameters of an ADC are its resolution and conversion rate.

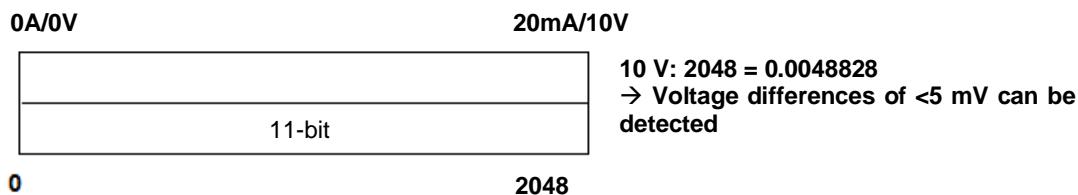
The more binary digits the digital representation uses, the finer the resolution is. For example, if only 1 bit was available for the voltage range of 0 to +10 V, you would only know whether the measured voltage is between 0 and +5 V or between +5 V and +10 V. With 2 bits, the range can be divided into 4 individual ranges, i.e., 0 to 2.5 / 2.5 to 5 / 5 to 7.5 / 7.5 to 10 V. Conventional A/D converters in control engineering use 8 or 11 bits for converting.



1: Analog value

2. Digital value

With 8 bits you have 256 individual ranges, while 11 bits provide a resolution of 2048 individual ranges.



4.4 Data types of the SIMATIC S7-1500

The SIMATIC S7-1500 has many different data types for representing different numerical formats. A list of some of the elementary data types is given below.

Data type	Size (bits)	Range	Example of constant entry
Bool	1	0 to 1	TRUE, FALSE, O, 1
Byte	8	16#00 to 16#FF	16#12, 16#AB
Word	16	16#0000 to 16#FFFF	16#ABCD, 16#0001
DWord	32	16#00000000 to 16#FFFFFFFF	16#02468ACE
Char	8	16#00 to 16#FF	'A', 'r', '@'
Sint	8	-128 to 127	123,-123
Int	16	-32,768 to 32,767	123, -123
Dint	32	-2,147,483,648 to 2,147,483,647	123, -123
USInt	8	0 to 255	123
UInt	16	0 to 65,535	123
UDInt	32	0 to 4,294,967,295	123
Real	32	+/-1.18 x 10⁻³⁸ to +/-3.40 x 10³⁸	123.456, -3.4, -1.2E+12, 3.4E-3
LReal	64	+/-2.23 x 10 ⁻³⁰⁸ to +/ -1.79 x 10 ³⁰⁸	12345.123456789 -1.2E+40
Time	32	T#-24d_20h_31m_23s_648ms to T#24d_20h_31m_23s_647ms Saved as: -2,147,483,648 ms to +2,147,483,647 ms	T#5m_30s 5#-2d T#1d_2h_15m_30x_45ms
String	Variable	0 to 254 characters in byte size	'ABC'

Note: The '**INT**' and '**REAL**' data types play a large role in analog value processing. This is because read-in analog values exist as 16-bit integers in the '**INT**' format, and in order to ensure exact further processing only '**REAL**' floating-point numbers should be used due to rounding errors in the case of '**INT**'.

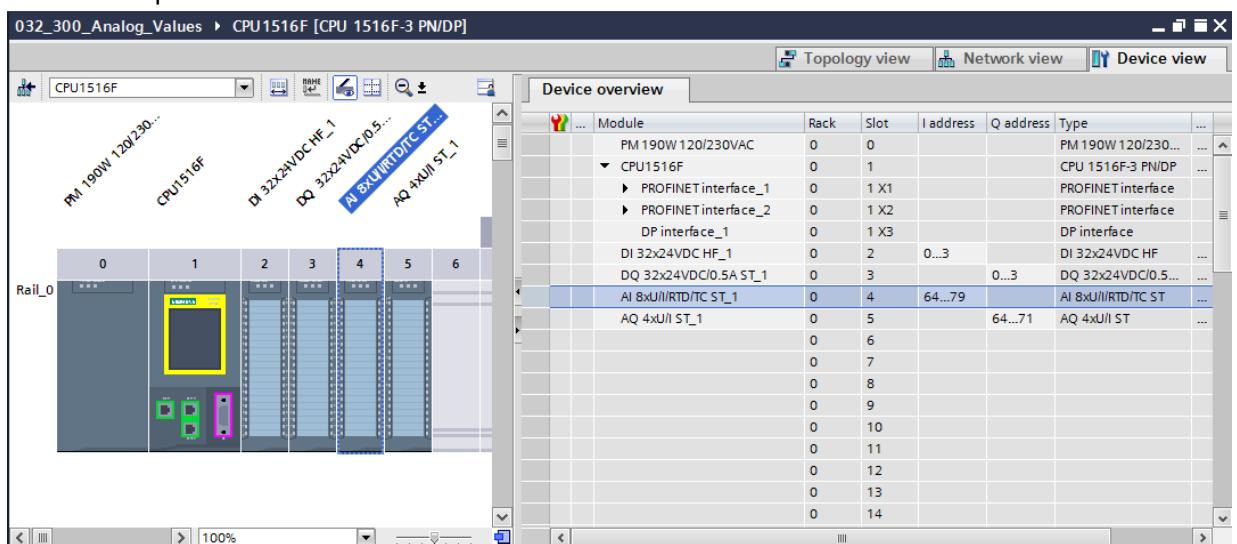
4.5 Reading/outputting analog values

Analog values are read into the PLC or output from the PLC as word information. These words are accessed, for example, with the following operands:

%IW 64	Analog input word 64
%QW 64	Analog output word 64

Each analog value ("channel") occupies one input or output word. The format is '**Int**', an integer.

The addressing of input and output words conforms to the addressing in the device overview. For example:

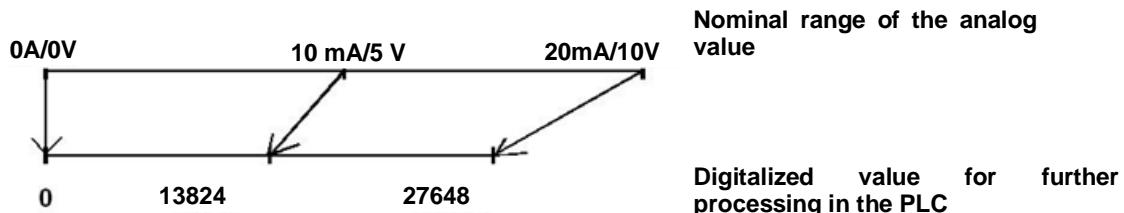


Here, the address of the first analog input would be %IW 64, that of the second analog input %IW 66, that of the third analog input %IW68, that of the fourth analog input %IW70, that of the fifth analog input %IW72, that of the sixth analog input %IW74, that of the seventh analog input %IW 76 and that of the eighth analog input %IW78.

The address of the first analog output would be %QW64, that of the second analog output %QW66, that of the third analog output %QW 68 and that of the fourth analog output %QW70.

The analog value transformation for further processing in the PLC is the same for analog inputs and analog outputs.

The digitized value ranges are as follows:



Often, these digitized values still have to be normalized by further processing them in the PLC in an appropriate manner.

4.6 Normalizing analog values

If an analog input value exists as a digitized value in the range +/- 27648, it must usually still be normalized so that the numerical values correspond to the physical quantities in the process.

Likewise, the analog output usually results from setting of a normalized value that then still has to be scaled to the output value +/- 27648.

In the TIA Portal, ready-made blocks or arithmetic operations are used for normalizing and scaling.

For this to be carried out as exactly as possible, the values for the normalizing must be converted to the REAL data type to minimize rounding errors.

5 Task

In this chapter, a function for analog control of the conveyor speed will be added to the program from chapter "SCE_EN_032-300 IEC Timers and Counters".

6 Planning

The analog control of the conveyor speed will be programmed in the "MOTOR_SPEEDCONTROL" [FC10] function as an expansion of the "SCE_EN_032-300 IEC Timers and Counters" project. This project must be retrieved from the archive in order to add this function. The "MOTOR_SPEEDCONTROL" [FC10] function will be called in the "Main" [OB1] organization block and wired. The control of the conveyor motor must be changed to - Q3 (conveyor motor -M1 variable speed).

6.1 Analog control of the conveyor speed

The speed will be set at an input of the "MOTOR_SPEEDCONTROL" [FC10] function in revolutions per minute (range: +/- 50 rpm). The data type is 32-bit floating-point number (Real).

First, the function will be checked for correct entry of the speed setpoint in the range +/- 50 rpm.

If the speed setpoint is outside the range +/- 50 rpm, the value 0 with data type 16-bit integer (Int) will be output at the output. The return value of the function (Ret_Val) will then be assigned the value TRUE (1).

If the speed setting is within the range +/- 50 rpm, this value will first be normalized to the range 0...1 and then scaled to +/- 27648 with data type 16-bit integer (Int) for output as the speed manipulated value at the analog output.

The output will then be connected with signal U1 (manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm).

6.2 Technology diagram

Here you see the technology diagram for the task.

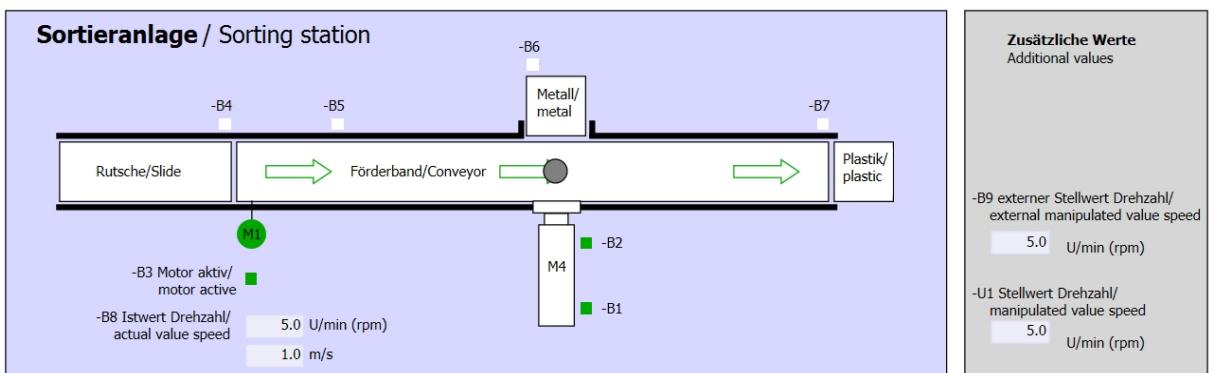


Figure 1: Technology diagram



Figure 2: Control panel

6.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO

I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
-------	------	-----	--------------------------------	----

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

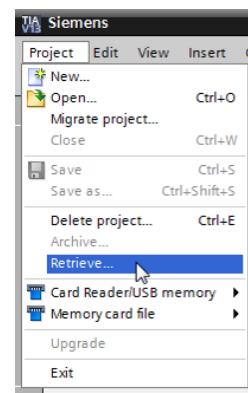
DI	Digital Input	DO	Digital Output
AI	Analog Input	AO	Analog Output
I	Input	Q	Output
NC	Normally Closed		
NO	Normally Open		

7 Structured step-by-step instructions

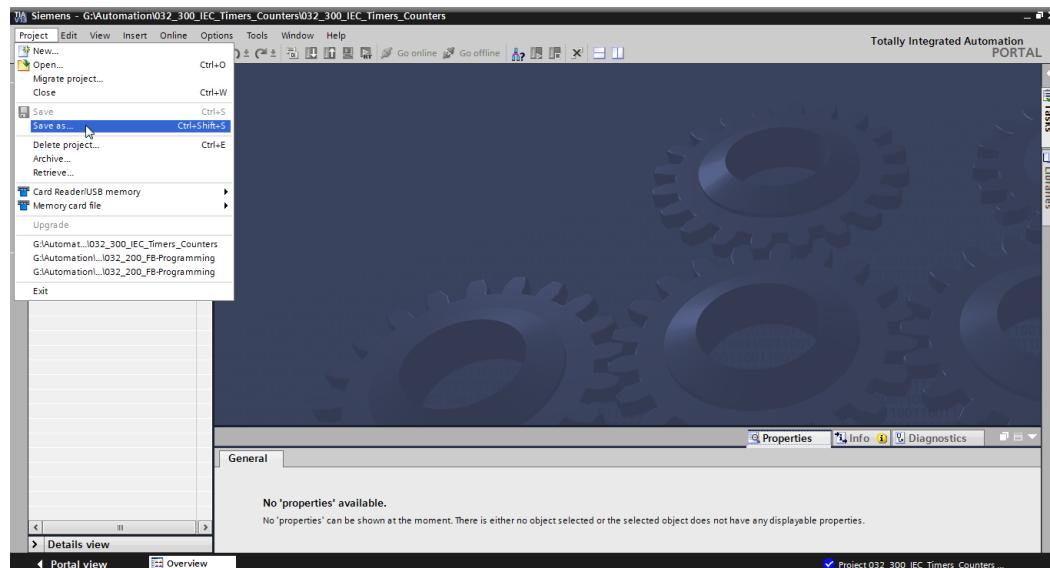
You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

- Before we can expand the "SCE_EN_032-300_IEC_Timers_Counters.zap13 project from chapter "SCE_EN_032-300_IEC_Timers_Counters", we must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open.
(→ Project → Retrieve → Select a .zap archive → Open)

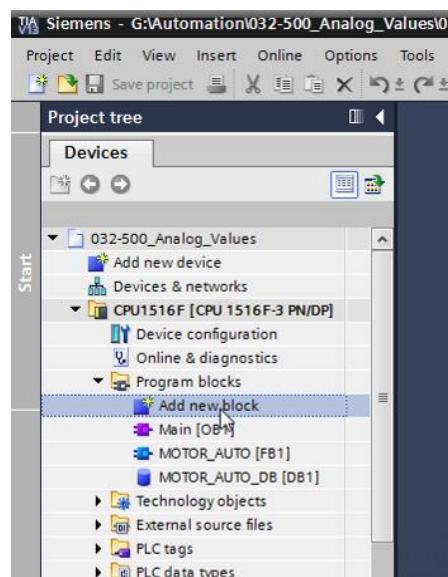


- The next step is to select the target directory where the retrieved project will be stored.
Confirm your selection with "OK".
(→ Target directory → OK)
- Save the opened project under the name 032-500_Analog_Values.
(→ Project → Save as ... → 032-500_Analog_Values → Save)



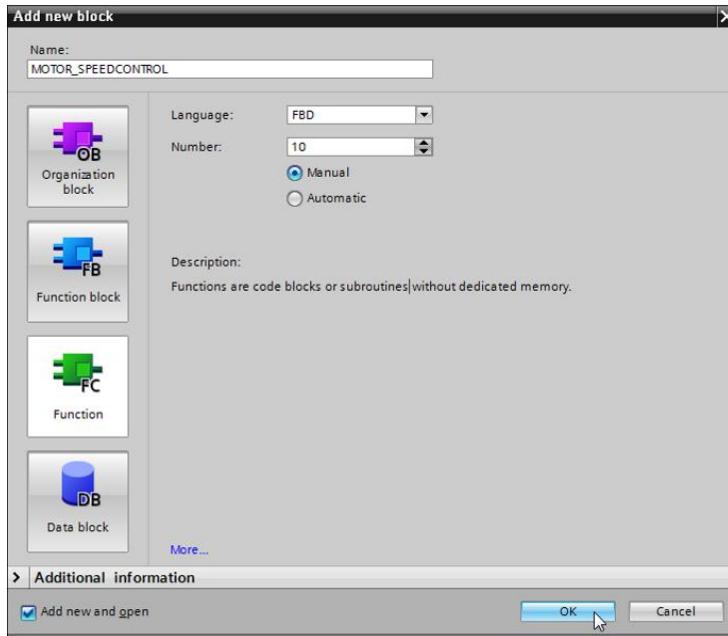
7.2 Create the "MOTOR_SPEEDCONTROL" function

- Select the 'Program blocks' folder of your CPU 1516F-3 PN/DP and then click "Add new block" to create a new function there.
(→ CPU_1516F [CPU 1516F-3 PN/DP] → Add new block)



- Select in the next dialog and rename your new block to: "MOTOR_SPEEDCONTROL". Set the language to FBD and manually assign the number "10". Select the "Add new and open" check box. Click "OK".

→ → Name: MOTOR_SPEEDCONTROL → Language: FBD → Number: 10 Manual
 → Add new and open → OK



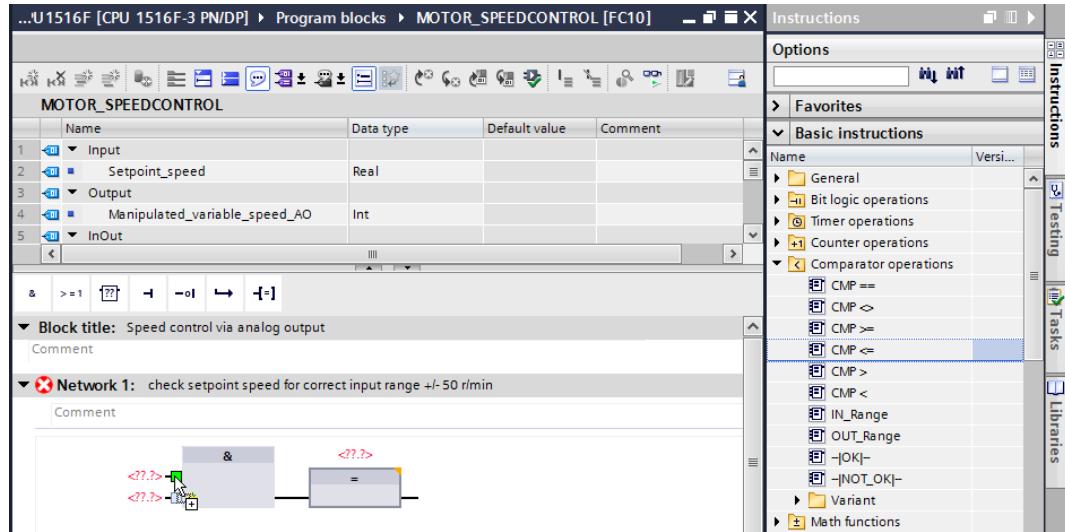
- Create the local tags with their comments as shown here and change the data type of the 'Return' tag from 'Void' to 'Bool'.
 (→ Bool)

Name	Data type	Default value	Comment
1 Input			
2 Setpoint_speed	Real		
3 Output			
4 Manipulated_variable_speed_AO	Int		
5 InOut			
6 <Add new>			
7 Temp			
8 Setpoint_speed_OK	Bool		
9 Manipulated_variable_speed_Norm	Real		
10 Constant			
11 <Add new>			
12 Return			
13 MOTOR_SPEEDCONTROL	Bool		

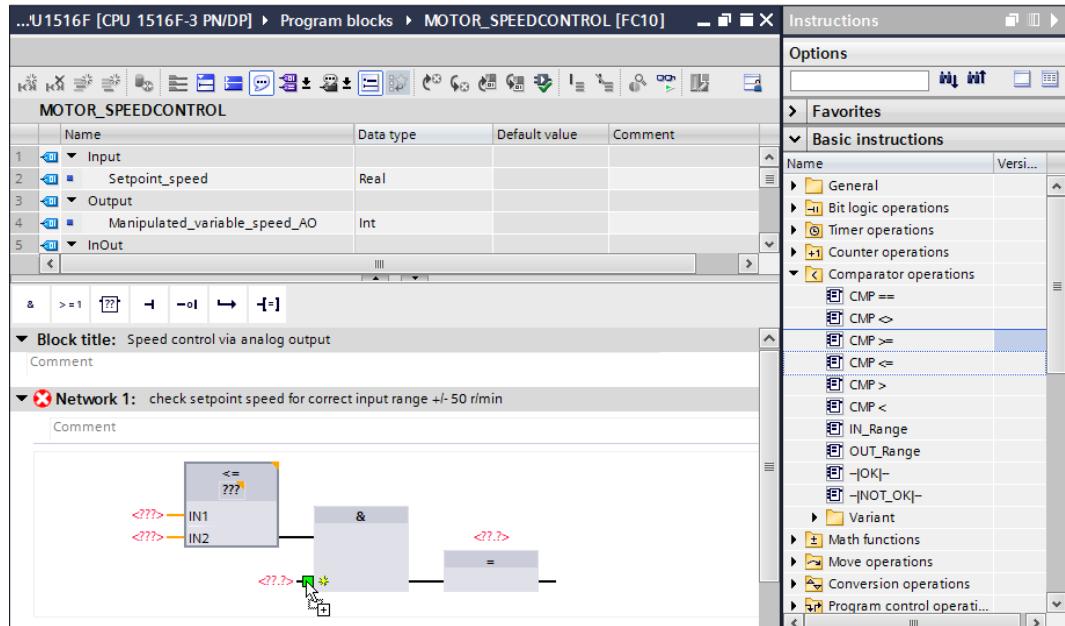
Note: Be sure to use the correct data types.

- Insert an Assignment ' := ' in the first network and an 'And' ' $\&$ ' in front of it. Then use drag-and-drop to move the 'Comparator operation' 'Less or equal' from the 'Basic instructions' onto the first input of the ' $\&$ ' AND logic operation.

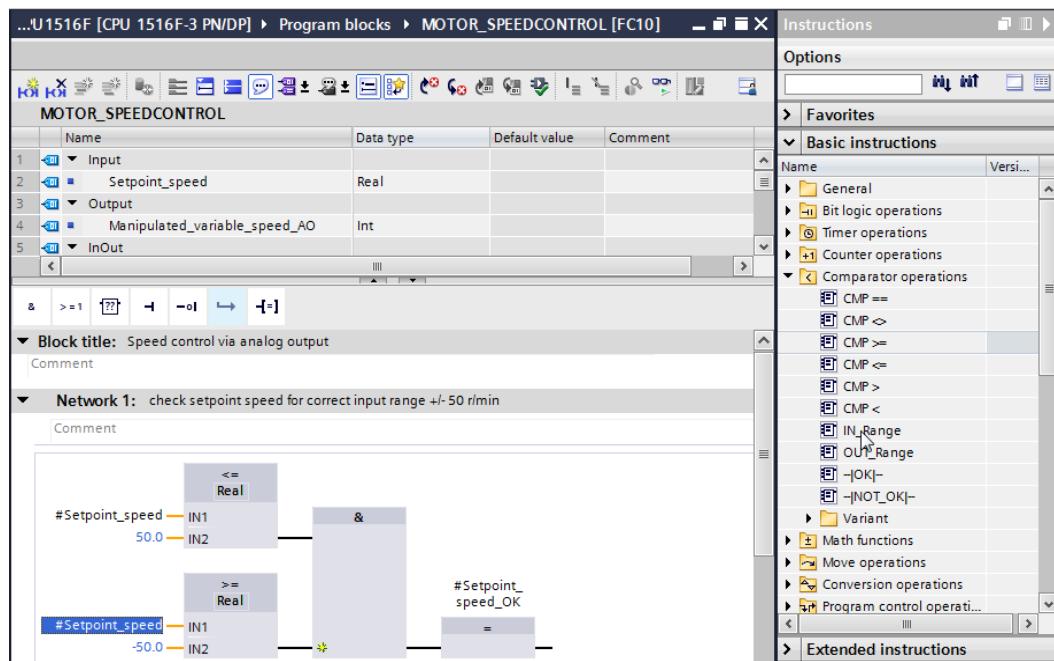
(→ := → $\&$ → Basic instructions → Comparator operations → $\text{CMP} \leq$)



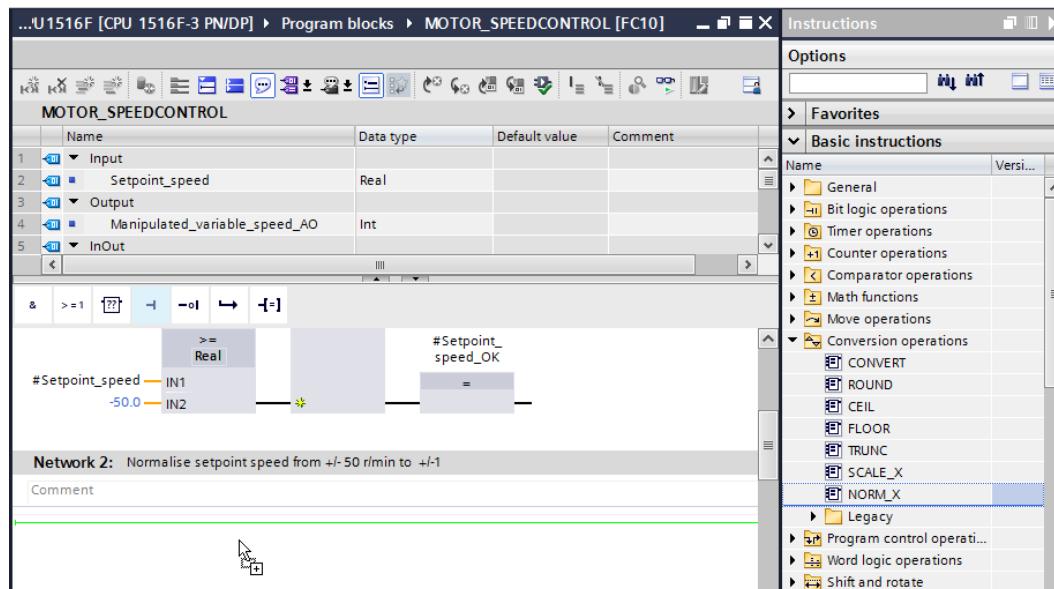
- Next use drag-and-drop to move the 'Comparator operation' 'Greater or equal' onto the second input of the ' $\&$ ' AND logic operation.
(→ Basic instructions → Comparator operations → $\text{CMP} \geq$)



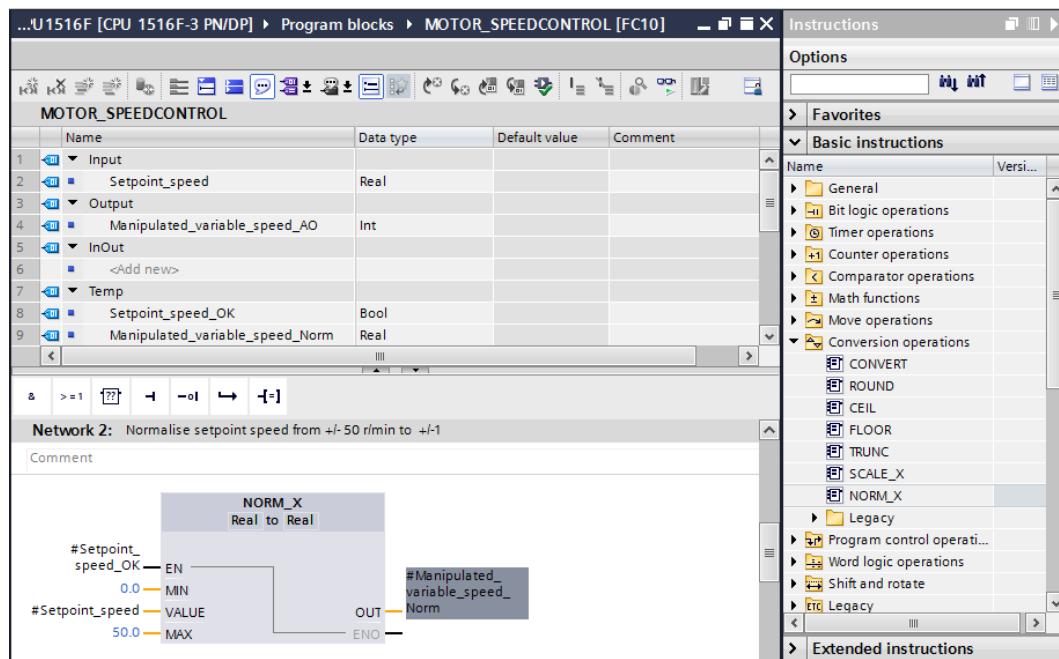
- Connect the contacts in Network 1 with the constants and local tags as shown here. The data types in the comparator operations are automatically adapted to 'Real'.



- Use drag-and-drop to move the 'Conversion operation' 'NORM_X' into Network 2 in order to normalize the speed setpoint of +/- 50 rpm to +/- 1.
(→ Basic instructions → Conversion operations → NORM_X)

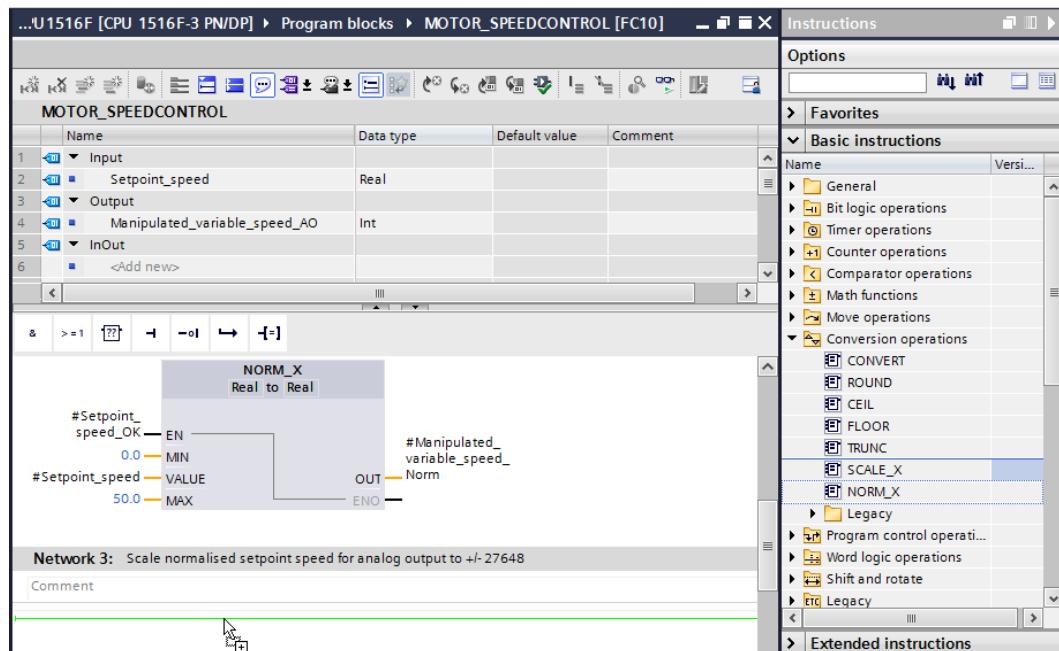


- Connect the contacts in Network 2 with the constants and local tags as shown here. The data types in 'NORM_X' are automatically adapted to 'Real'.

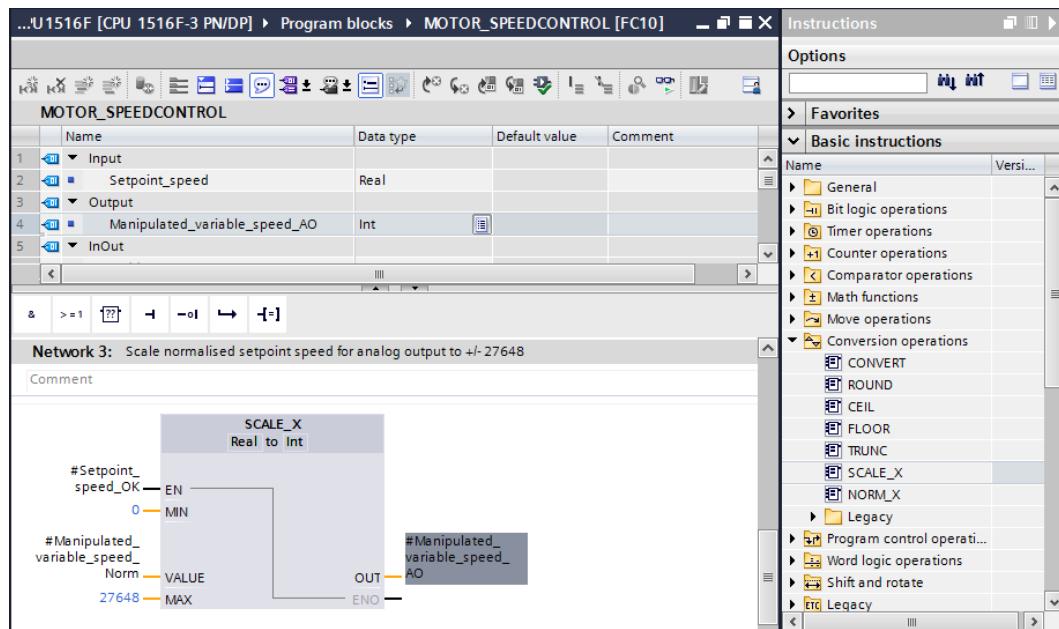


- Use drag-and-drop to move the 'Conversion operation' 'SCALE_X' into Network 3 in order to scale the speed setpoint from the normalized +/- 1 onto the range for the analog output +/- 27468.

(→ Basic instructions → Conversion operations → SCALE_X)

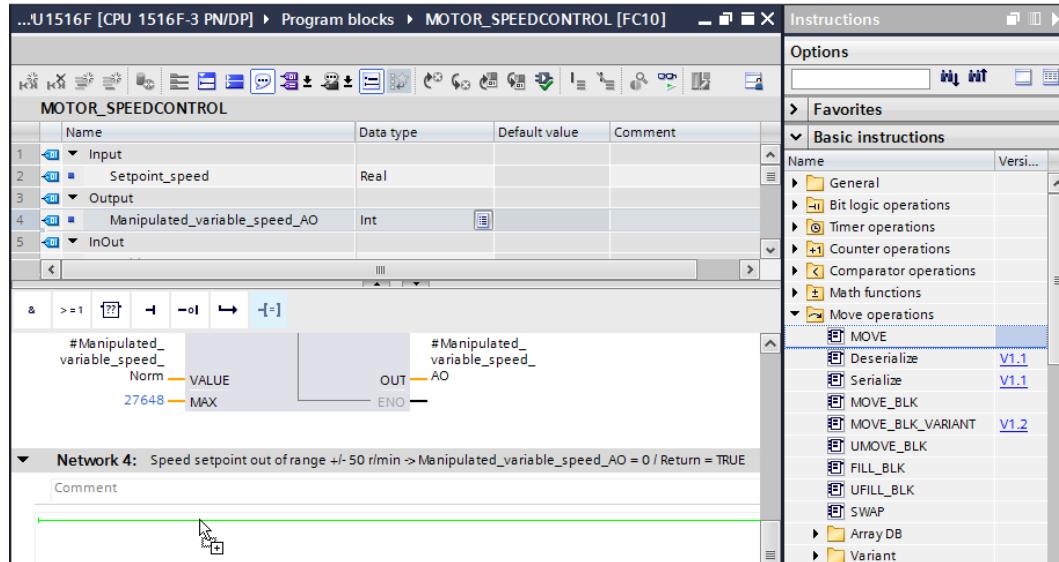


- Connect the contacts with the constants and local tags in Network 3 as well, as shown here. The data types in 'SCALE_X' are automatically changed to 'Real' or 'Int'.

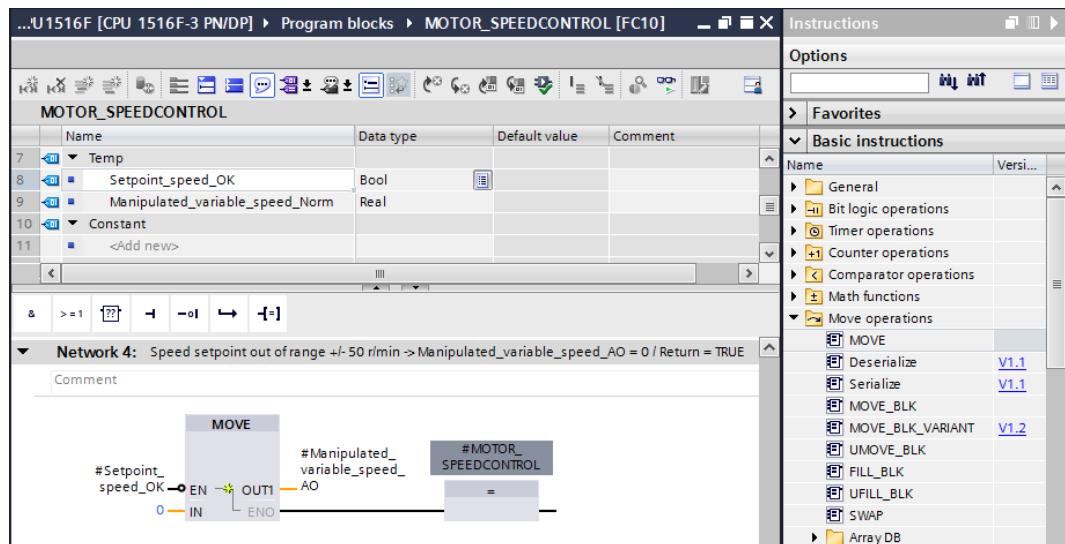


- Insert an Assignment ' $\text{I}=\text{I}$ ' in the fourth network. Use drag-and-drop to move the 'Move' command from the 'Move operations' folder under 'Basic instructions' in front of the Assignment.

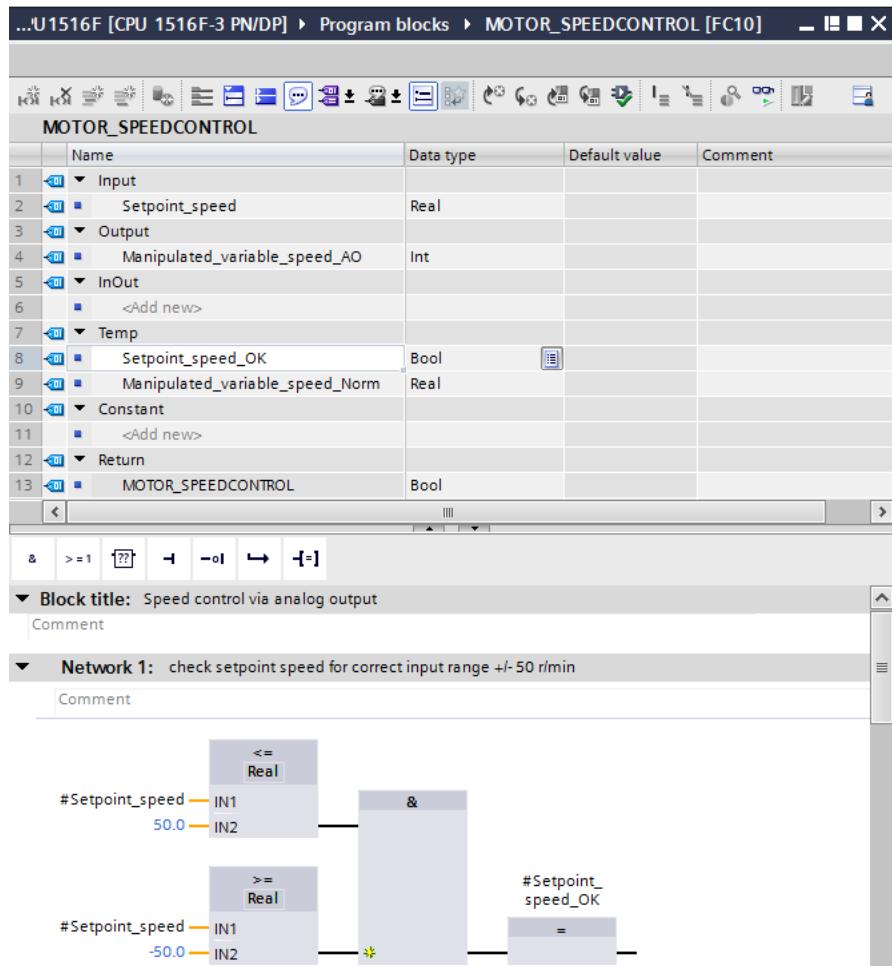
(→ $\text{I}=\text{I}$ → Basic instructions → Move operations → MOVE)

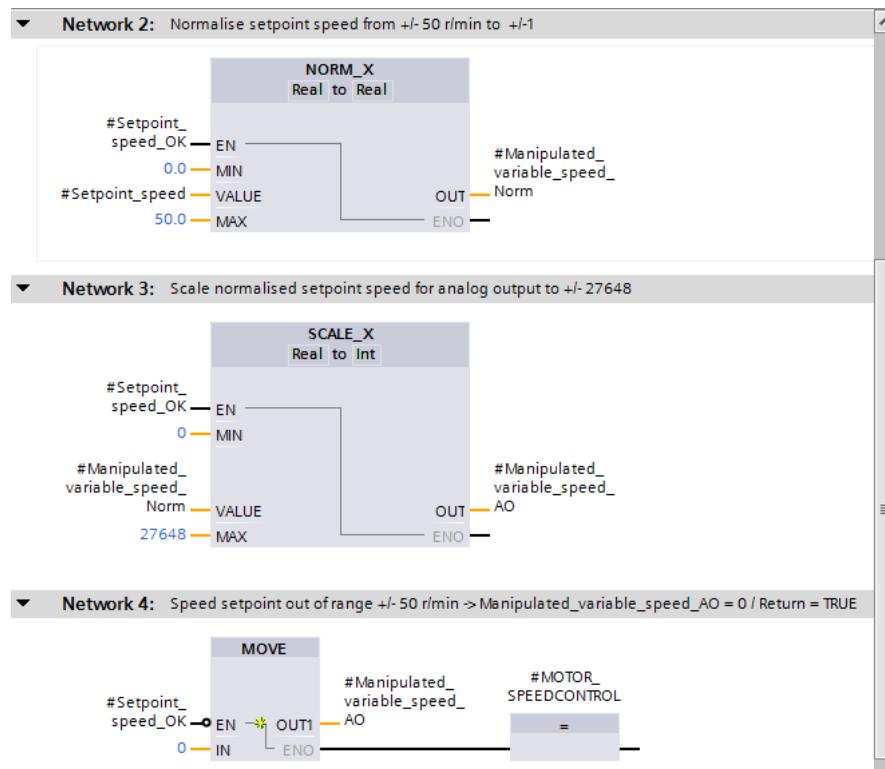


- The contacts in Network 4 will now be connected with constants and local tags as shown here. If the speed setpoint is not within the range +/- 50 rpm, the value '0' is output at the analog output and the value TRUE is assigned to the return value (Return) of the "MOTOR_SPEEDCONTROL" function.



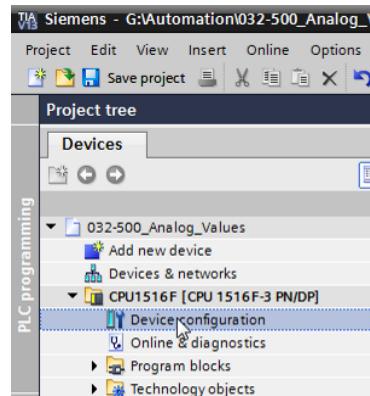
- Do not forget to click Save project. The finished function "MOTOR_SPEEDCONTROL" [FC10] in FBD is shown below.



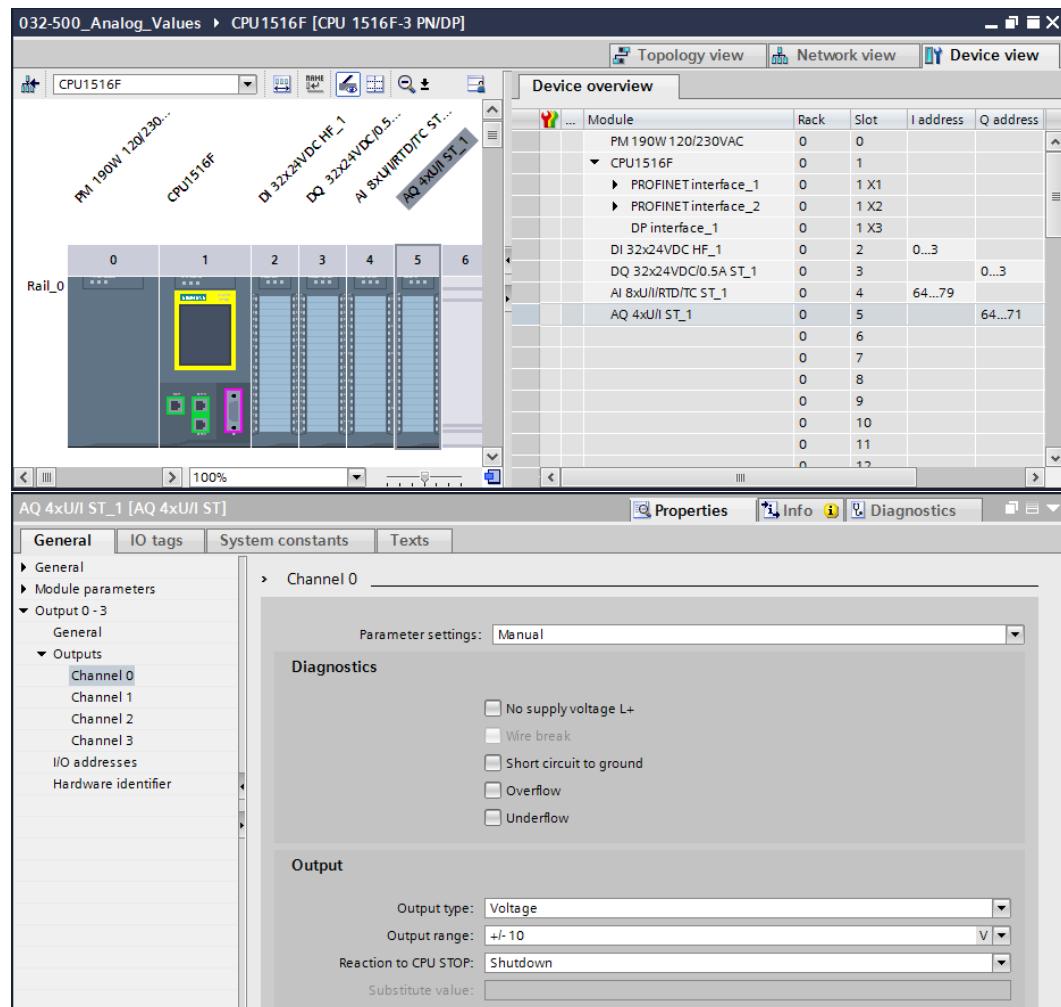


7.3 Configuration of the analog output channel

→ Double-click the 'Device configuration' to open it.

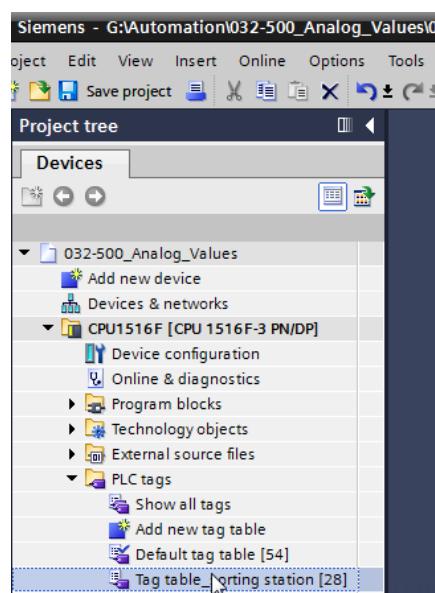


- Check the address setting and the configuration of the analog output channel 0.
 (→ Q address: 64...71 → Properties → General → Output 0 - 3 → Outputs → Channel 0
 → Output type: Voltage → Output range: +/- 10 V → Reaction to CPU STOP: Shutdown)

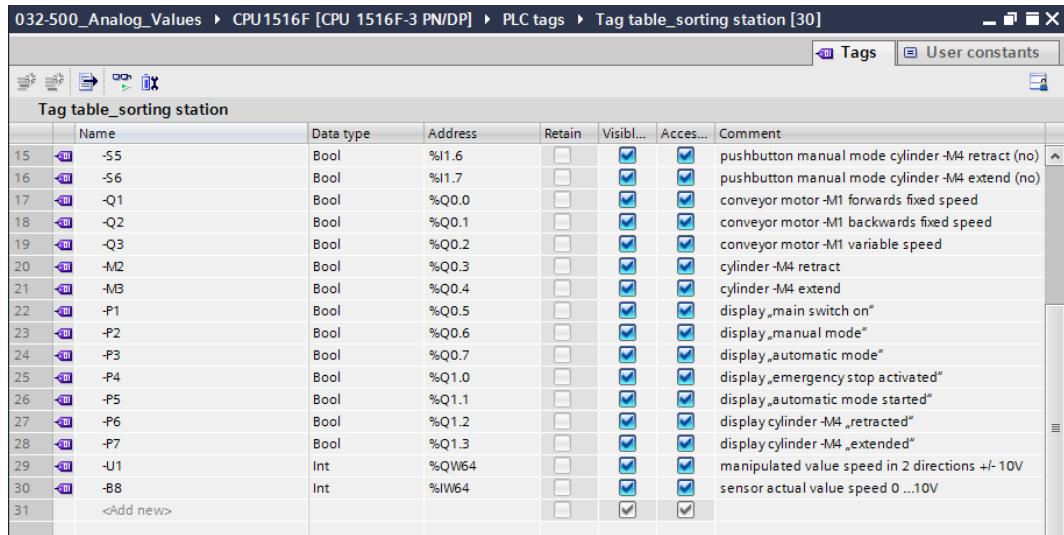


7.4 Expand the tag table to include analog signals

→ Double-click the 'Tag table_sorting station' to open it.



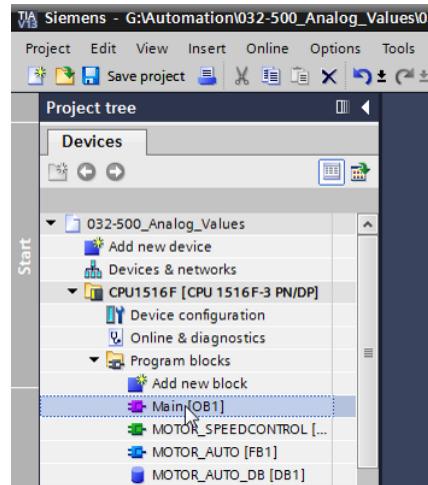
- Add the global tags for the analog value processing to the "Tag table_sorting station". An analog input B8 and an analog output U1 must be added.
 (→ U1 → %QW64 → B8 → %IW64)



	Name	Data type	Address	Retain	Visible...	Access...	Comment
15	-S5	Bool	%I1.6	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder-M4 retract (no)
16	-S6	Bool	%I1.7	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	pushbutton manual mode cylinder-M4 extend (no)
17	-Q1	Bool	%Q0.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor-M1 forwards fixed speed
18	-Q2	Bool	%Q0.1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor-M1 backwards fixed speed
19	-Q3	Bool	%Q0.2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	conveyor motor-M1 variable speed
20	-M2	Bool	%Q0.3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder-M4 retract
21	-M3	Bool	%Q0.4	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	cylinder-M4 extend
22	-P1	Bool	%Q0.5	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display_main switch on
23	-P2	Bool	%Q0.6	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display_manual mode
24	-P3	Bool	%Q0.7	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display_automatic mode
25	-P4	Bool	%Q1.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display_emergency stop activated
26	-P5	Bool	%Q1.1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display_automatic mode started
27	-P6	Bool	%Q1.2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display cylinder-M4 „retracted“
28	-P7	Bool	%Q1.3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	display cylinder-M4 „extended“
29	-U1	Int	%QW64	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	manipulated value speed in 2 directions +/- 10V
30	-B8	Int	%IW64	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	sensor actual value speed 0 ...10V
31	<Add new>				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

7.5 Call the block in the organization block

- Open the "Main [OB1]" organization block with a double-click.



- Add the temporary tag 'Motor_speed_monitoring_Ret_Val' to the local tags of OB1.
 These will be needed in order to interconnect the return value of the
 "MOTOR_SPEEDCONTROL" function.
 (→ Temp → Motor_speed_monitoring_Ret_Val → Bool)

Main				
	Name	Data type	Default value	Comment
1	Input			
2	Initial_Call	Bool		Initial call of this OB
3	Remanence	Bool		=True, if remanent data are available
4	Temp			
5	Motor_speed_monitoring_Ret_Val	Bool		
6	<Add new>			
7	Constant			
8	<Add new>			

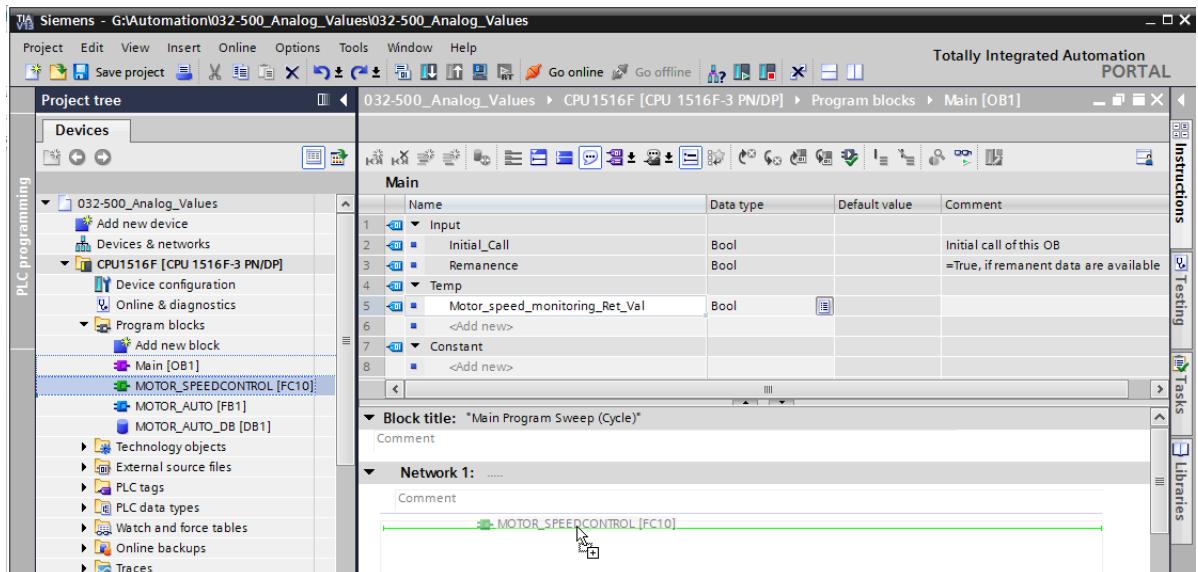
- Select the block title of OB1 and then click 'HOI' to insert a new Network 1 in front of the other networks



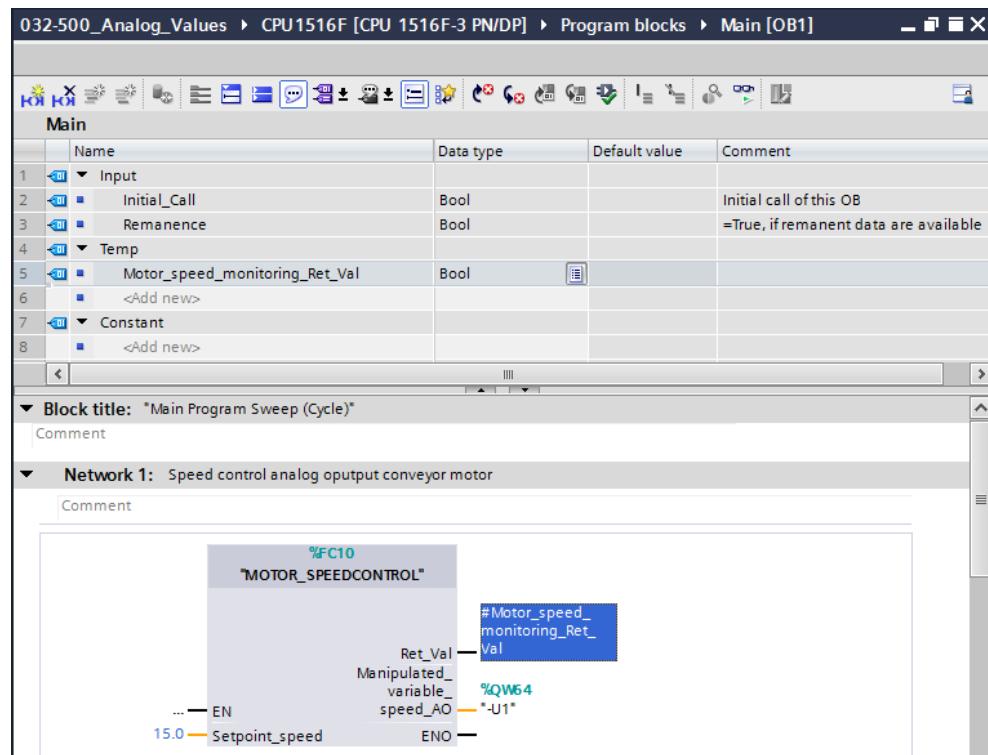
Main				
	Name	Data type	Default value	Comment
1	Input			
2	Initial_Call	Bool		Initial call of this OB
3	Remanence	Bool		=True, if remanent data are available
4	Temp			
5	Motor_speed_monitoring_Ret_Val	Bool		
6	<Add new>			
7	Constant			
8	<Add new>			

Block title: "Main Program Sweep (Cycle)"

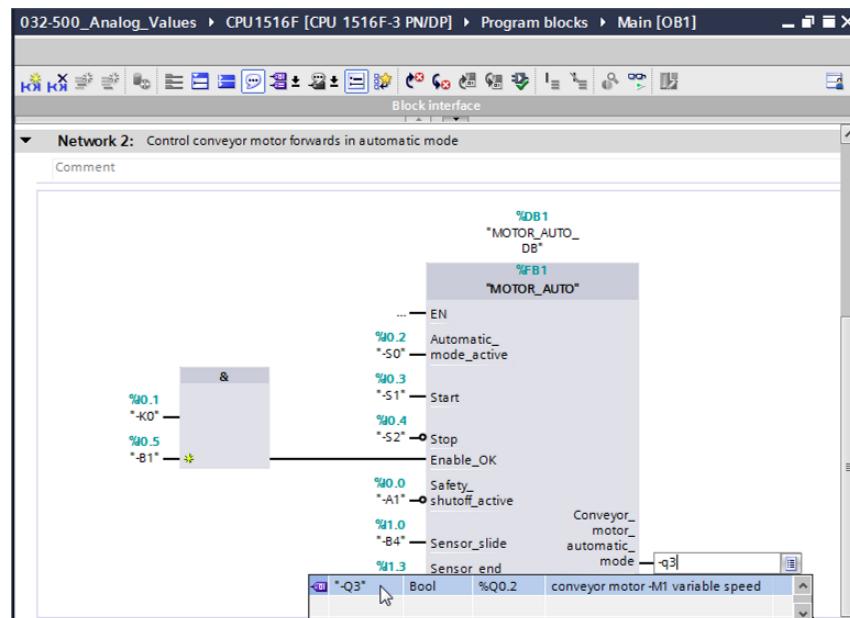
- Use drag-and-drop to move your "MOTOR_SPEEDCONTROL [FC10]" function onto the green line in Network 1.



- Connect the contacts with the constants and global and local tags here as shown.

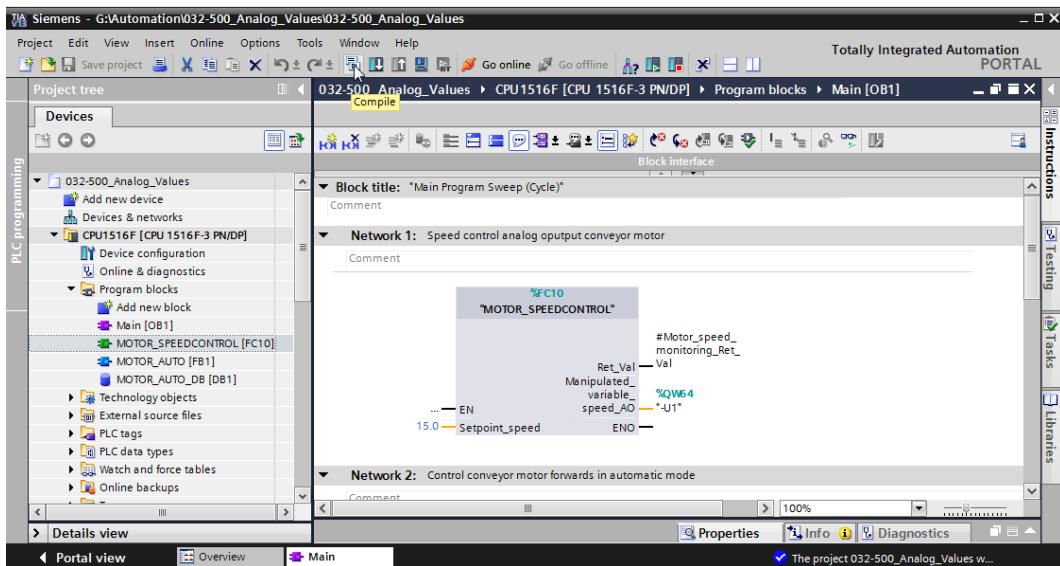


- Change the connection of output tag "Conveyor_motor_automatic_mode" in Network 2 to '-Q3' (Conveyor motor -M1 variable speed) so that the conveyor motor is controlled taking the analog speed setting into consideration.
(→ -Q3)

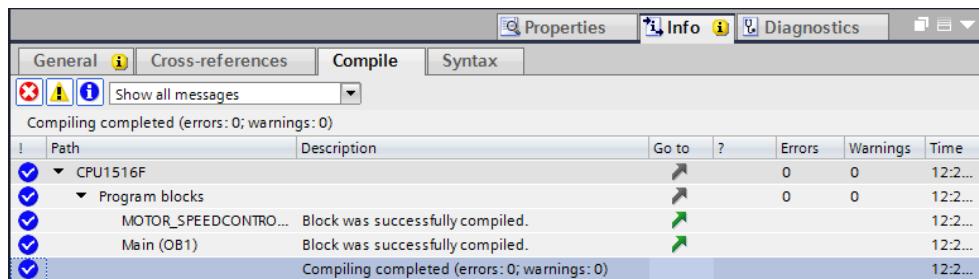


7.6 Save and compile the program

- To save your project, select the button in the menu. To compile all blocks, click the "Program blocks" folder and select the icon for compiling in the menu.
- (→ → Program blocks →



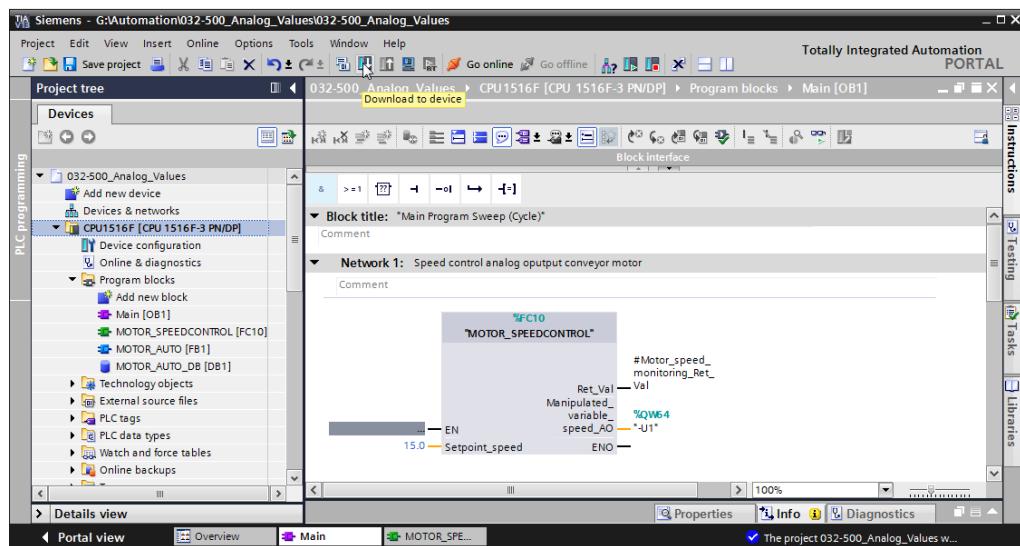
- The "Info", "Compile" area shows which blocks were successfully compiled.



7.7 Download the program

- After successful compilation, the complete controller with the created program including the hardware configuration can, as described in the previous modules, be downloaded.

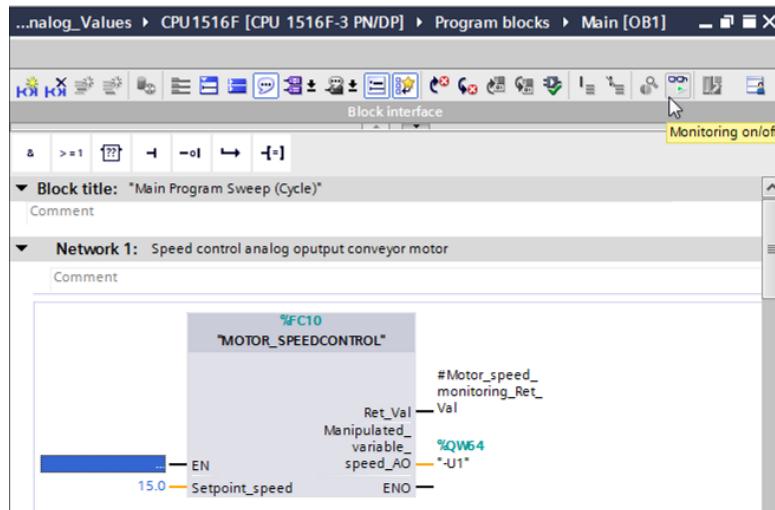


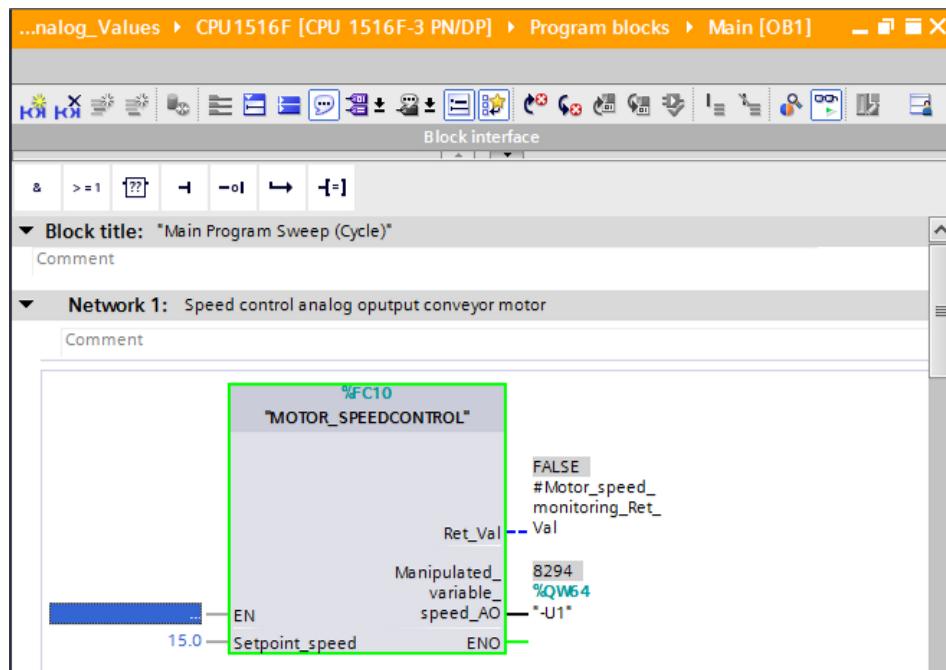


7.8 Monitor program blocks

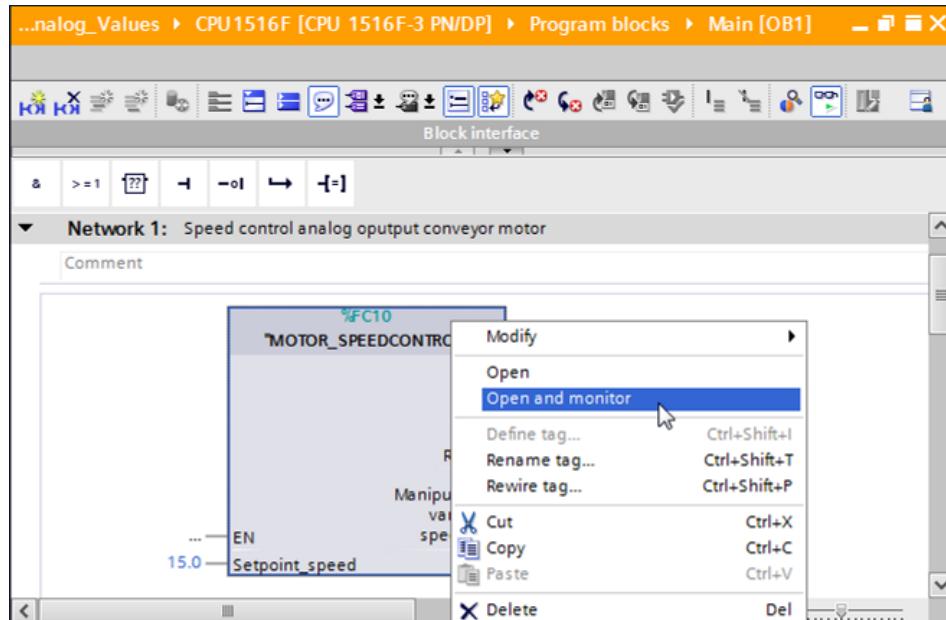
- The desired block must be open for monitoring the downloaded program. The monitoring can now be activated/deactivated by clicking the  icon.

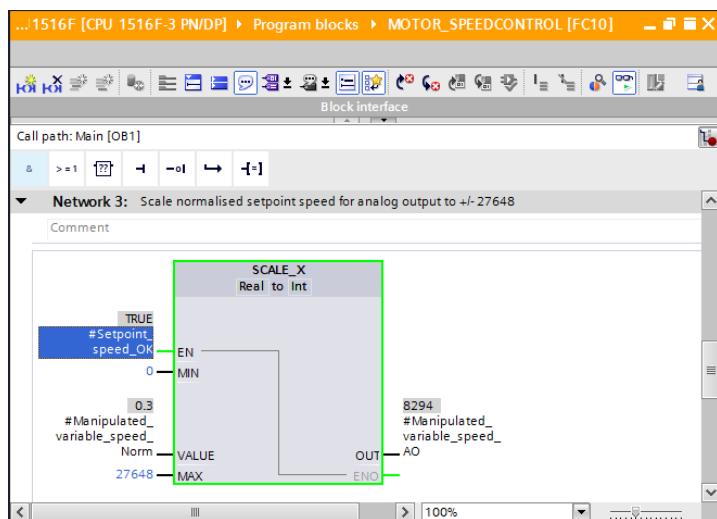
(→ Main [OB1] → 





- The "MOTOR_SPEEDCONTROL" [FC10] function called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking and the program code in the function can thus be monitored.
 (→ "MOTOR_SPEEDCONTROL" [FC10] → Open and monitor)





8 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
4	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
5	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
6	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
7	Manual mode (-S0 = 0) → -Q3 = 0	
8	Switch off station (-K0 = 0) → -Q3 = 0	
9	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
10	Project successfully archived	



Training Curriculum

TIA Portal Module 010

Global Data Blocks
for the SIMATIC S7-1500

Table of contents

1	Goal.....	309
2	Prerequisite.....	309
3	Required hardware and software.....	309
4	Theory	310
4.1	Data blocks	310
4.2	Data types of the SIMATIC S7-1500	311
4.3	Optimized blocks	312
4.4	Download without reinitialization	312
5	Task.....	313
6	Planning.....	313
6.1	Global data block for speed control and speed monitoring of the motor	313
6.2	Technology diagram	313
6.3	Reference list	314
7	Structured step-by-step instructions.....	315
7.1	Retrieve an existing project.....	315
7.2	Create the global data block "SPEED_MOTOR"	316
7.3	Access to data of the data block in the organization block	320
7.4	Save and compile the program	324
7.5	Download the program.....	325
7.6	Monitor/modify values in data blocks.....	325
7.7	Initialize setpoints/reset start values.....	326
7.8	Snapshots in data blocks	328
7.9	Expand data block and download it without reinitialization	330
8	Checklist	334

GLOBAL DATA BLOCKS

1 Goal

In this chapter, you will become acquainted with the use of global data blocks for the SIMATIC S7-1500 with the TIA Portal programming tool.

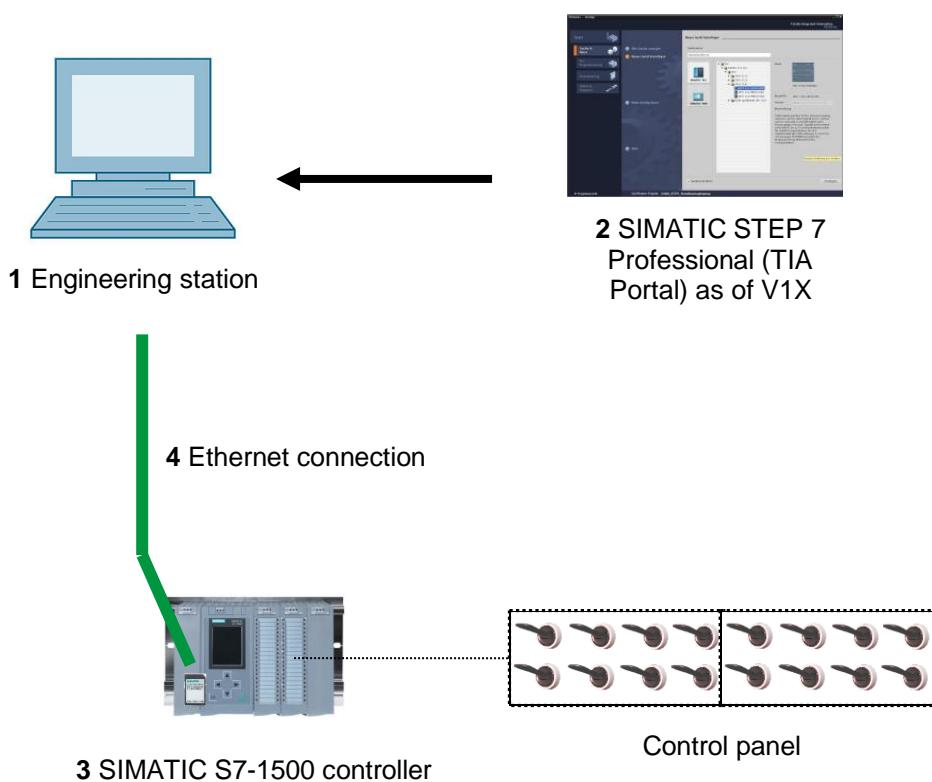
The module explains the structure and creation of and access to global data blocks for the SIMATIC S7-1500. It also shows the steps for creating a global data block in the TIA Portal and for accessing this data in the program with read and write access.

2 Prerequisite

This chapter builds on the chapter Analog Values with the SIMATIC S7 CPU1516F-3 PN/DP.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs and analog inputs and outputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

4.1 Data blocks

In contrast to logic blocks, data blocks contain no instructions. Rather, they serve as memory for user data.

Data blocks thus contain variable data that is used by the user program. You can define the structure of global data blocks as required.

Global data blocks store data that can be used **by all other blocks** (see Figure 1). Only the associated function block should access instance data blocks. The maximum size of data blocks varies depending on the utilized CPU.

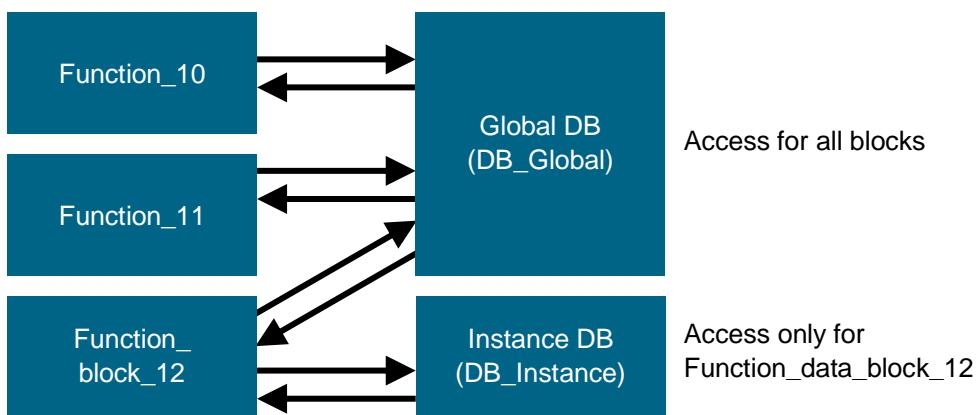


Figure 1: Difference between global DB and instance DB.

Application examples for **global data blocks** are:

- Saving of information about a storage system. "Which product is located where?"
- Saving of recipes for particular products.

The data in data blocks is stored retentively in most cases. This data is then retained in the event of a power failure or after a STOP/START of the CPU.

4.2 Data types of the SIMATIC S7-1500

The SIMATIC S7-1500 has many different data types for representing different numerical formats. A list of some of the elementary data types is given below.

Data type	Size (bits)	Range	Example of constant entry
Bool	1	0 to 1	TRUE, FALSE, O, 1
Byte	8	16#00 to 16#FF	16#12, 16#AB
Word	16	16#0000 to 16#FFFF	16#ABCD, 16#0001
DWord	32	16#00000000 to 16#FFFFFFFF	16#02468ACE
Char	8	16#00 to 16#FF	'A', 'r', '@'
Sint	8	-128 to 127	123,-123
Int	16	-32,768 to 32,767	123, -123
Dint	32	-2,147,483,648 to 2,147,483,647	123, -123
USInt	8	0 to 255	123
UInt	16	0 to 65,535	123
UDInt	32	0 to 4,294,967,295	123
Real	32	+/-1.18 x 10 ⁻³⁸ to +/-.40 x 10 ³⁸	123.456, -3.4, 1.2E+12 3.4E-3
LReal	64	+/-2.23 x 10 ⁻³⁰⁸ to +/-.179 x 10 ³⁰⁸	12345.123456789 -1.2E+40
Time	32	T#-24d_20h_31m_23s_648ms to T#24d_20h_31m_23s_647ms Saved as: -2,147,483,648 ms to +2,147,483,647 ms	T#5m_30s 5#-2d T#1d_2h_15m_30x_45ms
String	Variable	0 to 254 characters in byte size	'ABC'
Array		With arrays, data of a uniform data type is arranged one after the other and addressed consecutively in the address area. The properties of each array element are identical and are configured in the array tag.	
Struct		The STRUCT data type represents a data structure that consists of a fixed number of components of different data types. Components of STRUCT or ARRAY data type can also be nested in a structure.	
...		For other data types, refer to the online help.	

4.3 Optimized blocks

S7-1500 controllers have optimized data storage. In optimized blocks all tags are automatically sorted based on their data type. The sorting ensures that data gaps between the tags are minimized and the tags are stored in a manner that optimizes their access by the controller.

- The tags are always accessed as fast as possible because the file storage by the system is optimized and is independent of the declaration.
- There is no danger of inconsistencies due to incorrect, absolute accesses because symbolic access is generally used.
- Declaration changes do not result in access errors because accesses by process visualization systems, for example, occur symbolically.
- Individual tags can be selectively defined as retentive.
- No settings are needed or possible in the instance data block. Everything will be set in the assigned FB (e.g., retentivity).
- Memory reserves in the data block enable changes to be made without loss of actual values (download without reinitialization).

4.4 Download without reinitialization

To enable the subsequent editing of user programs that are already running in a CPU, the S7-1500 controllers support the option of expanding the interfaces of optimized function or data blocks during operation. You can download the modified blocks without switching the controller to STOP mode and without affecting the actual values of previously downloaded tags.

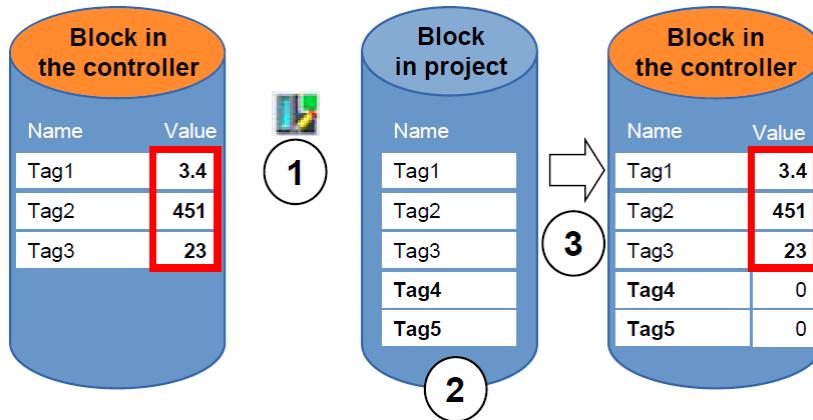


Figure 2: Download without reinitialization

The following steps can be performed while the controller is in RUN mode:

1. Activate "Download without reinitialization"
2. Insert newly defined tags in an existing block
3. Download expanded block to the controller

The newly defined tags are initialized. The existing tags retain their current value.

Prerequisite: a memory reserve must have been defined for the block beforehand and the block with this memory reserve must have downloaded to the CPU.

5 Task

In this chapter, the program from chapter Analog will be expanded to include a data block that centrally provides the parameters for the two functions "MOTOR_SPEEDCONTROL" [FC10] and "MOTOR_SPEEDMONITORING" [FC11].

6 Planning

The data management and setpoint setting for the "MOTOR_SPEEDCONTROL" [FC10] and "MOTOR_SPEEDMONITORING" [FC11] functions will be carried out using the global data block "SPEED_MOTOR" [DB2].

This will be added to the "032-500_Analog_Values" project. This project must be retrieved from the archive beforehand.

In the "Main" [OB1] organization block, the two functions "MOTOR_SPEEDCONTROL" [FC10] and "MOTOR_SPEEDMONITORING" [FC11] must then be connected with the tags from global data block "SPEED_MOTOR" [DB2].

6.1 Global data block for speed control and speed monitoring of the motor

Speed setpoint and actual speed value will be created in Real data format (32-bit floating-point number) as the first tags in the "SPEED_MOTOR" [DB2] data block. The speed setpoint is thereby given the start value + 14 U/min

A structure (Struct) 'Positive_Speed' will then be created for monitoring the positive speed limits.

This structure contains the 2 tags 'Threshold_Error' (start value + 15 rpm) and 'Threshold_Warning' (start value + 10 rpm) in Real data format (32-bit floating-point number) and the 2 tags 'Error' and 'Warning' in Bool data format (binary number).

The structure (Struct) 'Positive_Speed' will then be inserted again as a copy and renamed to 'Negative_Speed' for monitoring the negative speed limits.

The 'Threshold_Error' tag is given the start value - 16 rpm and the 'Threshold_Warning' tag the start value - 14 rpm.

6.2 Technology diagram

Here you see the technology diagram for the task.

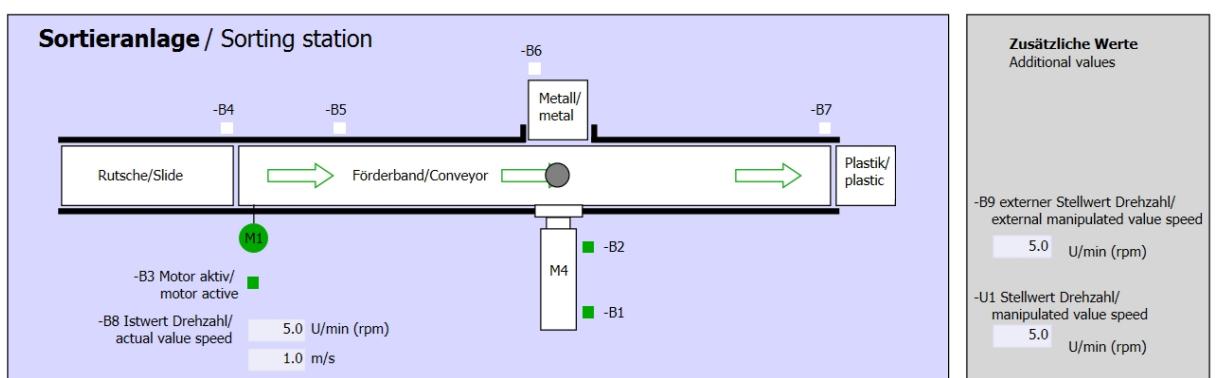


Figure 3: Technology diagram

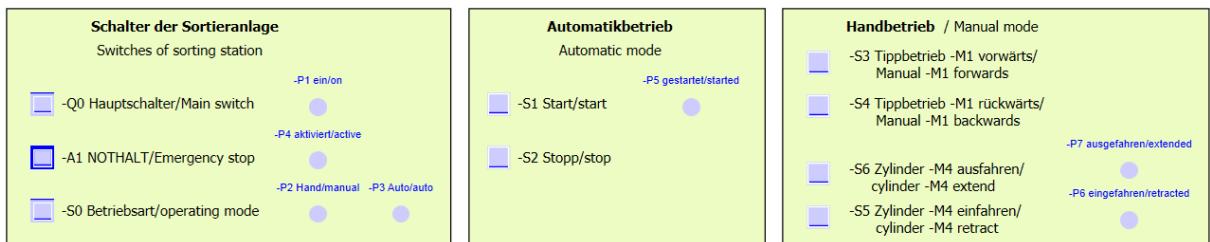


Figure 4: Control panel

6.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
IW64	BOOL	-B8	Sensor actual value speed +/-10V corresponds to +/- 50 rpm	

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

DI	Digital Input	DO	Digital Output
AI	Analog Input	AO	Analog Output
I	Input	Q	Output
NC	Normally Closed		
NO	Normally Open		

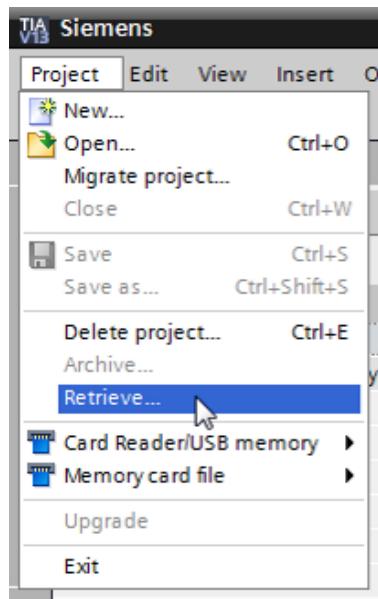
7 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

- Before we can expand the "SCE_EN_032-500_Analog_Values_R1508.zap13" project from chapter "SCE_EN_032-500 Analog Values", we must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open.

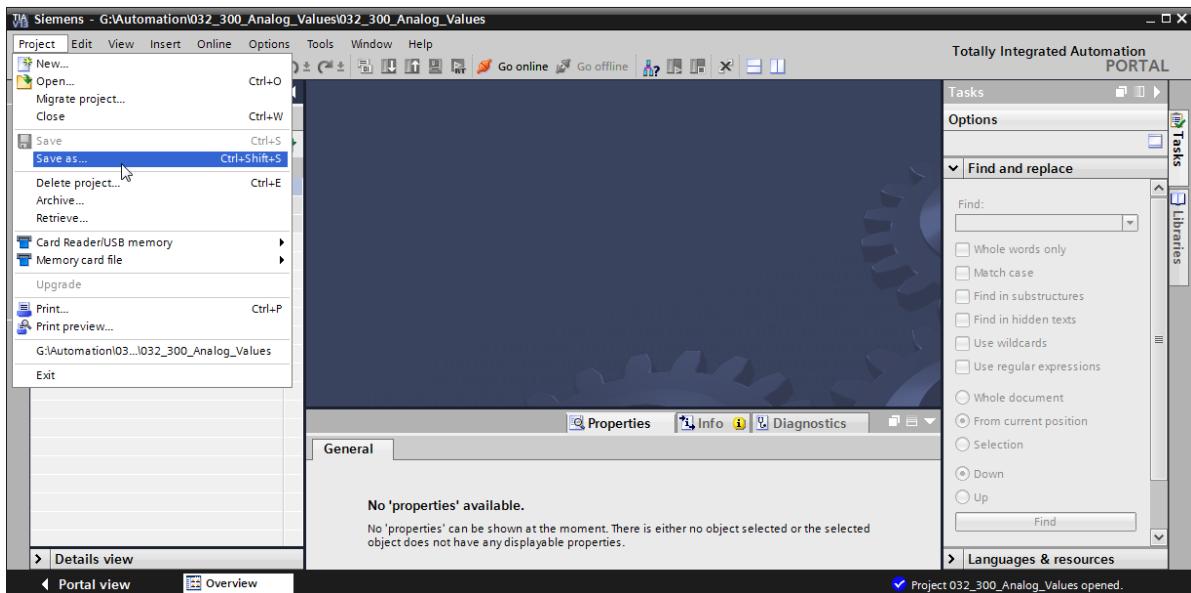
(→ Project → Retrieve → Select a .zap archive → Open)



- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK".

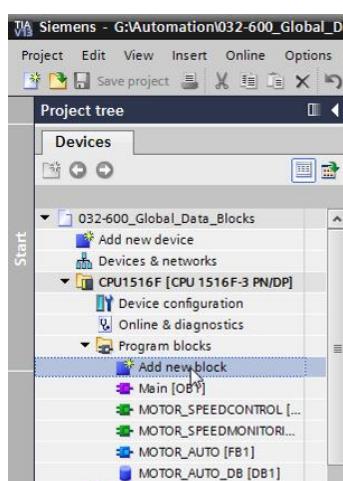
(→ Target directory → OK)

- Save the opened project under the name 032-600_Global_Data_Blocks.
 (→ Project → Save as ... → 032-600_Global_Data_Blocks → Save)



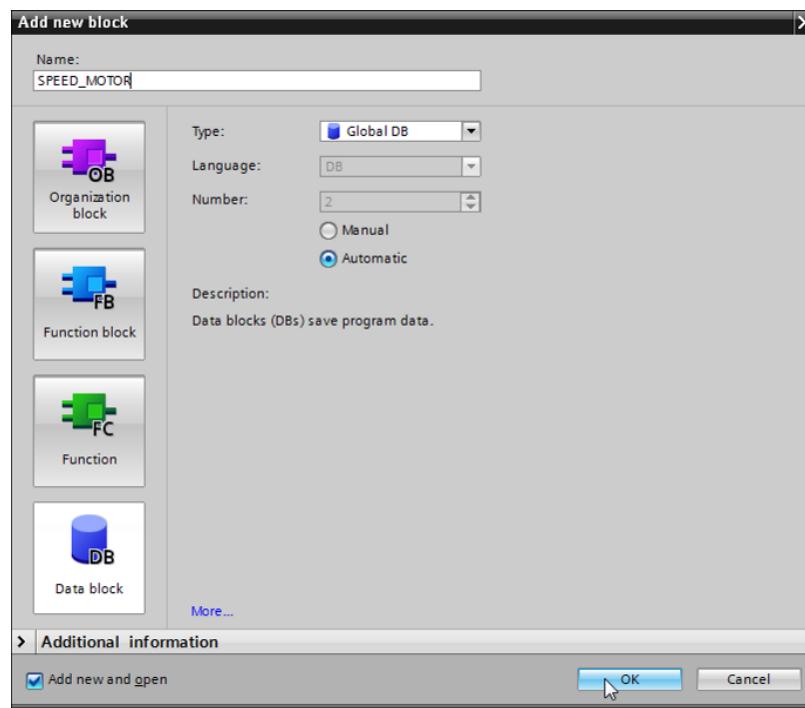
7.2 Create the global data block "SPEED_MOTOR"

- Select the 'Program blocks' folder of your CPU 1516F-3 PN/DP and then click "Add new block" to create a new global data block there.
 (→ CPU_1516F [CPU 1516F-3 PN/DP] → Add new block)



- Select in the next dialog and rename your new block to: "SPEED_MOTOR". Select 'Global DB' as the type. The number '2' will be automatically assigned. Select the "Add new and open" check box. Click "OK".

(→ → Name: SPEED_MOTOR → Type: Global DB → Add new and open → OK)



- The "SPEED_MOTOR" data block is automatically displayed. Start by creating the 'Speed_Setpoint' and 'Speed_Actual_Value' tags shown here with their associated comments. Select 'Real' as the data type. Also set a start value of 10.0 rpm for the 'Speed_Setpoint'.
 (→ Speed_Setpoint → Real → 14.0 → Speed_Actual_Value → Real)

032-600_Global_Data_Blocks > CPU1516F [CPU 1516F-3 PN/DP] > Program blocks > SPEED_MOTOR [DB2]								
SPEED_MOTOR								
1	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)

Note: Be sure to use the correct data types.

- Next we create a tag structure 'Struct' so it can be duplicated later.
 (→ Struct)

032-600_Global_Data_Blocks > CPU1516F [CPU 1516F-3 PN/DP] > Program blocks > SPEED_MOTOR [DB2]								
SPEED_MOTOR								
1	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	<Add new>							
		RTM						
		Real						
		STime						
		SInt						
		String						
		Struct						

→ Name the structure 'Positive_Speed' and enter a comment.

(→ Positive_Speed)

The screenshot shows the 'SPEED_MOTOR [DB2]' data block in the TIA Portal. A new structure named 'Positive_Speed' is being created under the 'Static' category. The 'Comment' column for this structure contains the text 'Parameters for error/warning positive speed'. Other static variables 'Speed_Setpoint' and 'Speed_Actual_Value' are also listed.

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5	<Add new>							
6	<Add new>							

→ Create the tags for the speed monitoring below the structure as shown here.

The screenshot shows the 'SPEED_MOTOR [DB2]' data block. Below the 'Positive_Speed' structure, four new tags are added: 'Threshold_Error', 'Threshold_Warning', 'Error', and 'Warning'. Each tag has its appropriate data type, start value, and checkboxes for visibility and setpoint.

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded

Note: Be sure to use the correct data types.

→ Then select the structure and copy it.

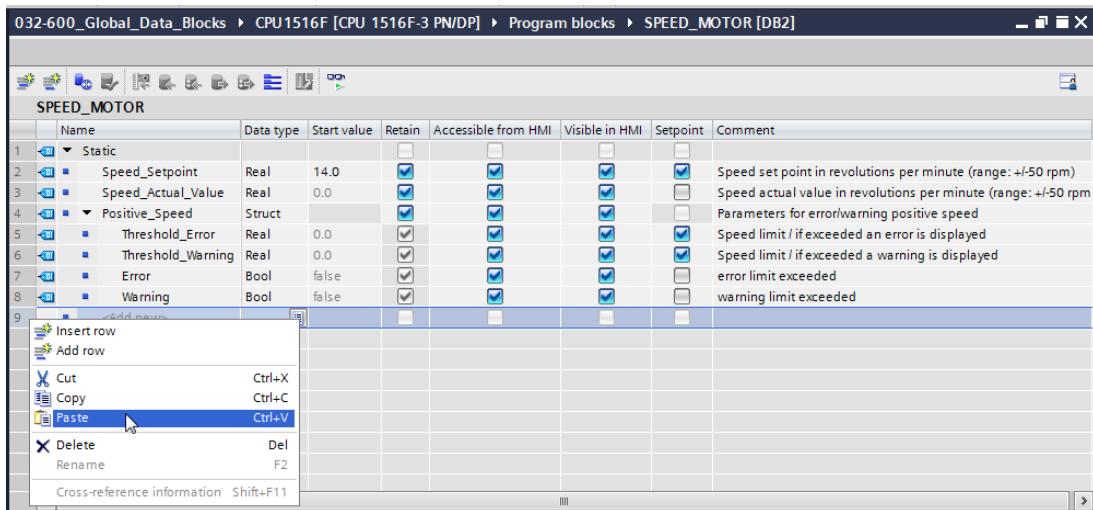
(→ Copy)

The screenshot shows the 'SPEED_MOTOR [DB2]' data block. The 'Positive_Speed' structure is selected, and a context menu is open. The 'Copy' option is highlighted. Other options like 'Insert row', 'Add row', 'Cut', and 'Delete' are also visible.

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5	Insert row		0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Add row		0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Cut		else	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Copy		else	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded

→ Paste the copied structure below the 'Positive_Speed' structure again.

(→ Paste)

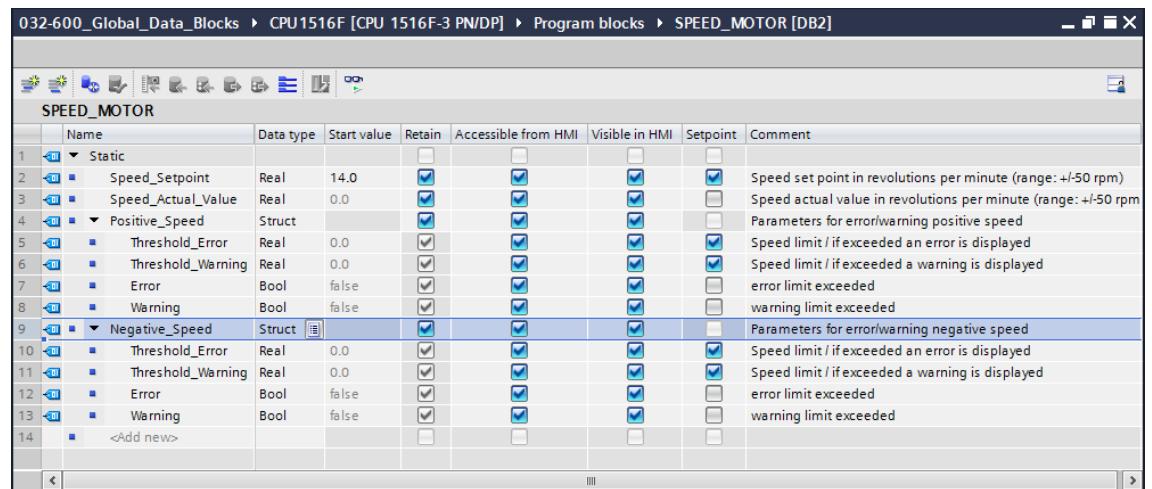


The screenshot shows the TIA Portal interface for creating a global data block named 'SPEED_MOTOR'. A context menu is open at the bottom of the table, with 'Paste' highlighted. The table lists various tags under the 'Static' structure, including Speed_Setpoint, Speed_Actual_Value, Positive_Speed, Threshold_Error, Threshold_Warning, Error, and Warning.

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9	<Add new>							

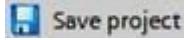
→ Rename the new structure to 'Negative_Speed' and enter a comment.

(→ Negative_Speed)



The screenshot shows the completed global data block 'SPEED_MOTOR'. The 'Negative_Speed' structure has been added as row 9. The table now contains 14 rows. The 'Negative_Speed' structure is defined as a 'Struct' type. The comments for the new structure are: 'Parameters for error/warning negative speed'.

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Static							
2	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5	Threshold_Error	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning negative speed
10	Threshold_Error	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
14	<Add new>							

Do not forget to click . The finished global data block "SPEED_MOTOR" [DB2] is shown below. Check to verify that Retain is selected and the corresponding start value is entered for all tags. The data will thus be retained in the data block even after a power failure or a STOP/START of the CPU. The check boxes for 'Accessible from HMI' and 'Visible in HMI' should also all have a check mark so that all tags in future expansions of this project will be accessible by the visualization systems (HMI). We will select the 'Setpoint' check box only for the default values in our data block.

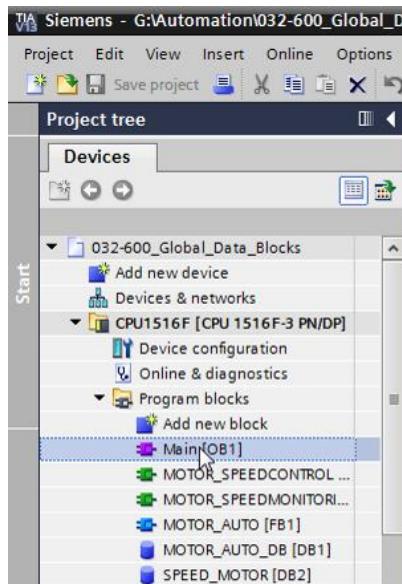
(→

	Name	Data type	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	Speed_Setpoint	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
2	Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
3	Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
4	Threshold_Error	Real	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
5	Threshold_Warning	Real	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
6	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
7	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
8	Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning negative speed
9	Threshold_Error	Real	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
10	Threshold_Warning	Real	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
11	Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
12	Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
13	<Add new>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
14				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

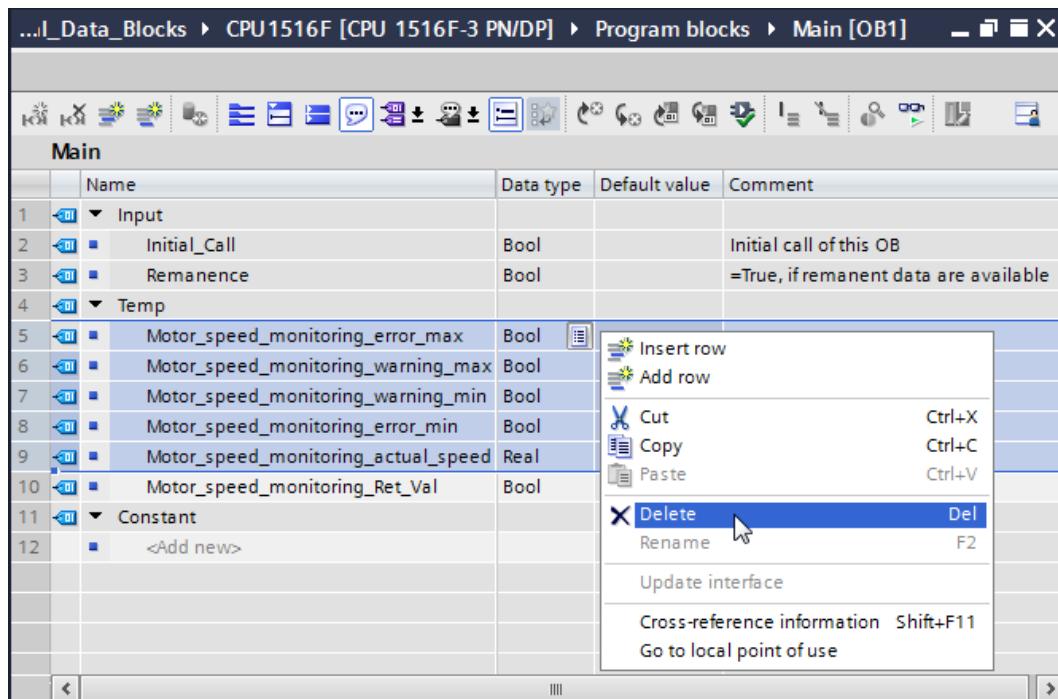
Note: The use of setpoints is described further below in the step-by-step instructions of the module.

7.3 Access to data of the data block in the organization block

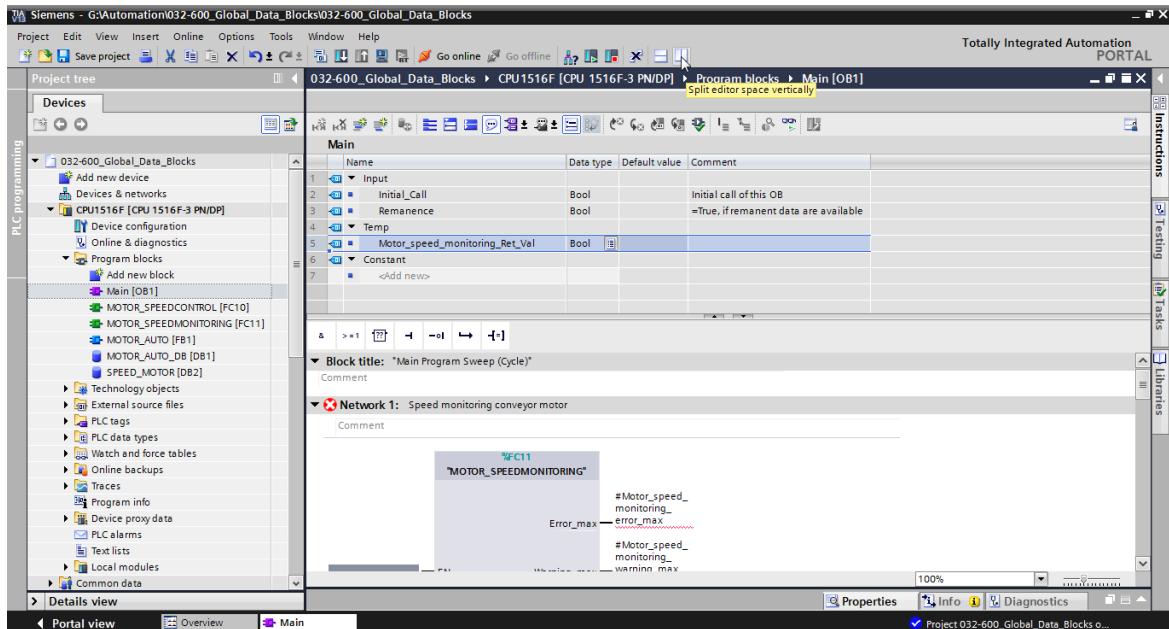
- Open the "Main [OB1]" organization block with a double-click.



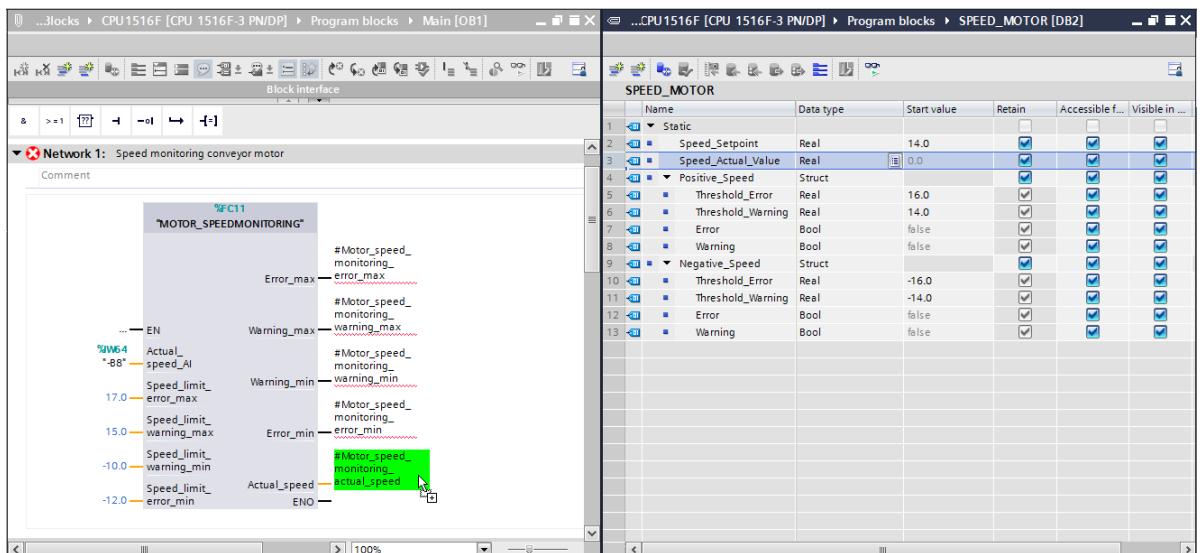
- Delete the temporary tags in Main [OB1] that are no longer needed. Only the Boolean tag 'Motor_Speed_Control_Ret_Val' is still needed.
(→ Delete)



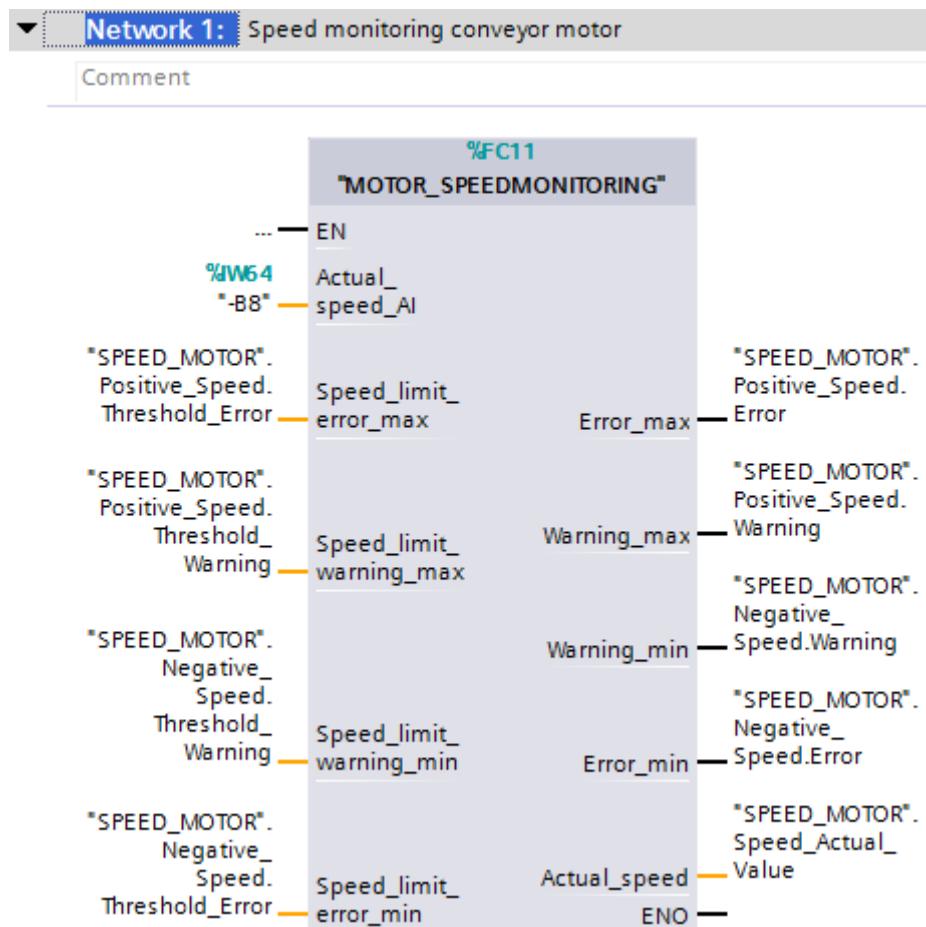
- Have the "SPEED_MOTOR" [DB2] data block and the "Main" [OB1] organization block displayed side by side by clicking the '□' icon to vertically split the editor area.
- (→ □)



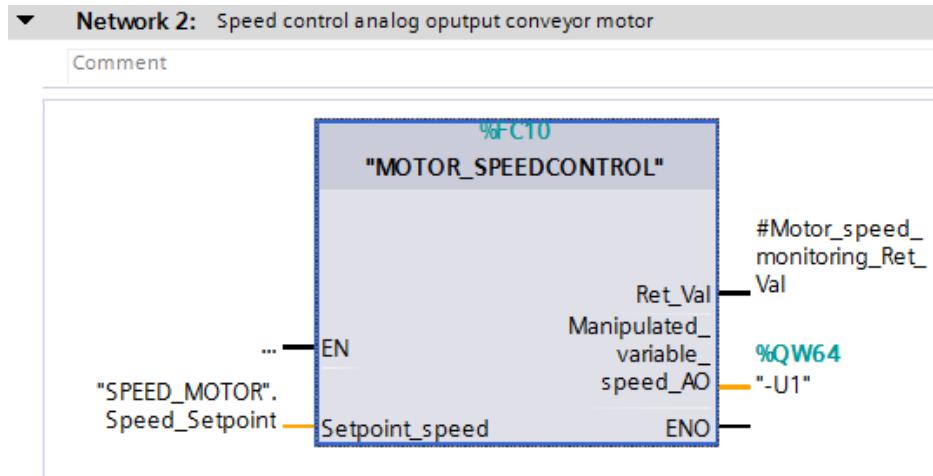
- Use drag-and-drop to move the tags needed for the interconnection from the "SPEED_MOTOR" [DB2] data block onto the connections of the called functions and function blocks in the "Main" [OB1] organization block. First we move the 'Speed_Actual_Value' tag onto the 'Actual_speed' output of the "MOTOR_SPEEDMONITORING" [FC11] block.
- (→ Speed_Actual_Value)



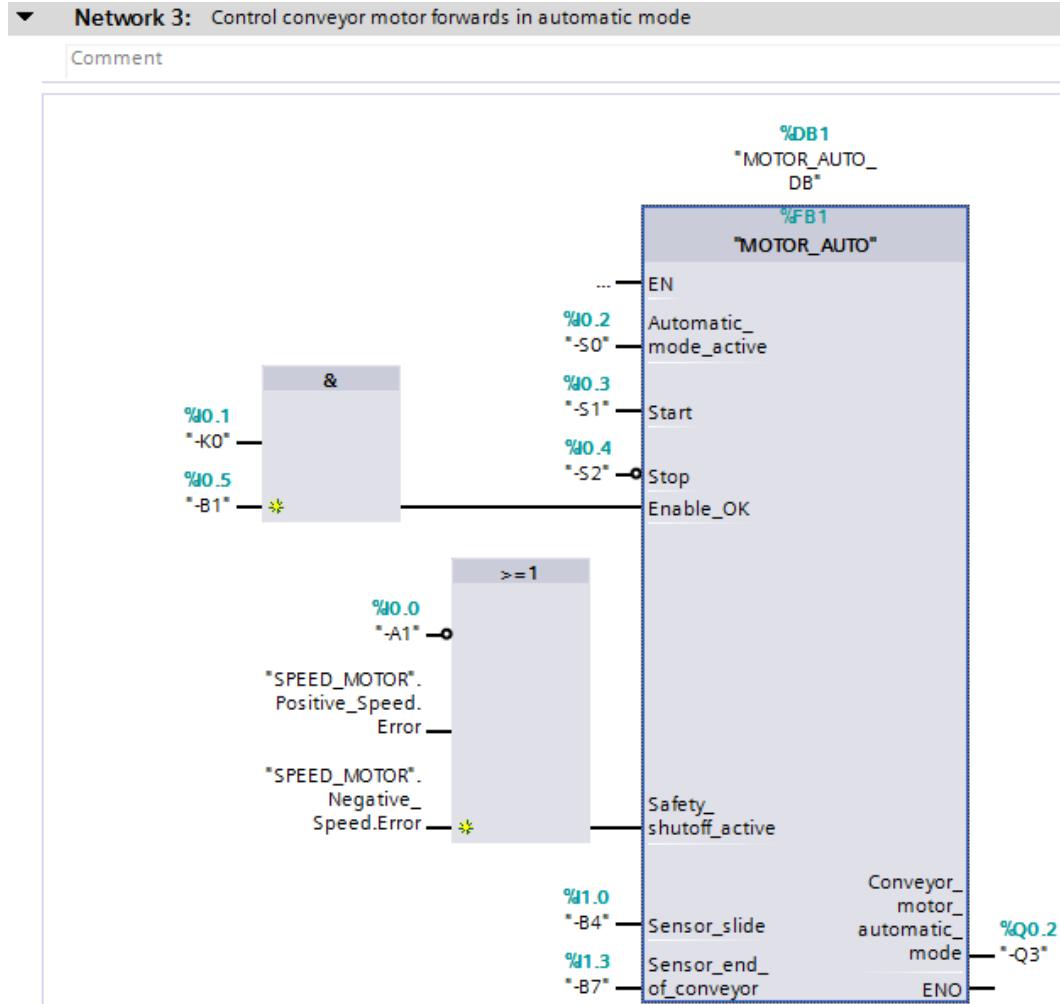
- Also connect the other contacts in Network 1 with tags from the "SPEED_MOTOR" [DB2] data block as shown here.



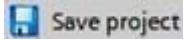
- Connect the contacts in Network 2 with tags from the "SPEED_MOTOR" [DB2] data block as shown here.

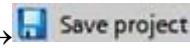


- Connect the contacts in Network 3 with tags from the "SPEED_MOTOR" [DB2] data block as shown here.



7.4 Save and compile the program

- To save your project, click the  button in the menu. To compile all blocks, click the "Program blocks" folder and select the  icon for compiling in the menu.

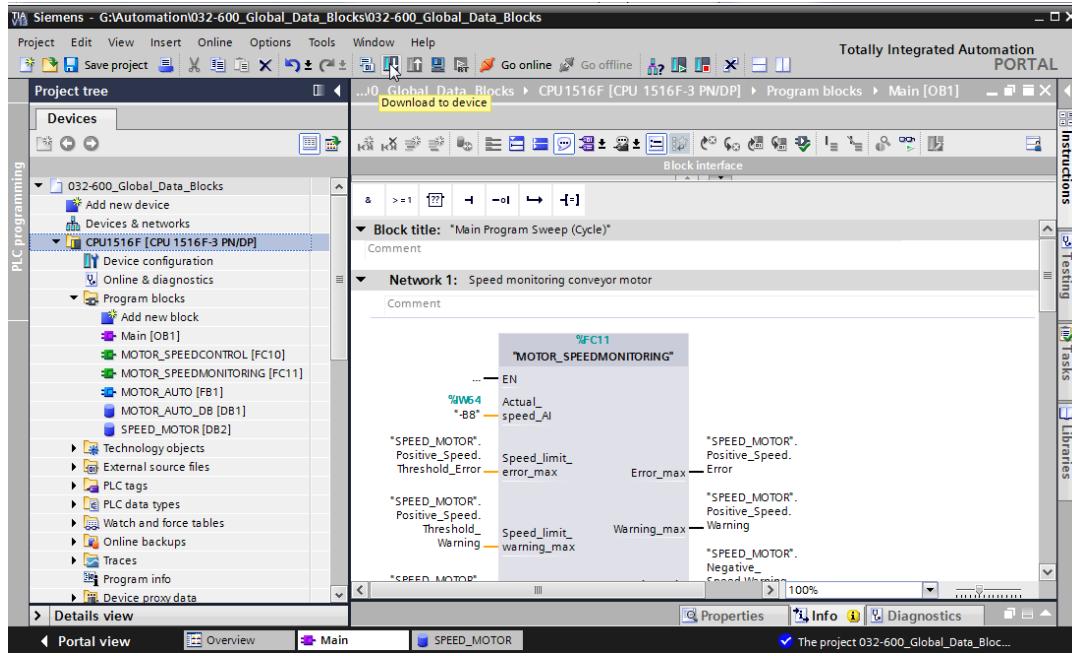
(→  → Program blocks → 

- The "Info", "Compile" area shows which blocks were successfully compiled.

Properties							Info	Diagnostics	
General		Cross-references		Compile		Syntax			
   Show all messages		Compiling completed (errors: 0; warnings: 0)							
!	Path	Description	Go to	?	Errors	Warnings	Time		
✓	CPU1516F		↗		0	0	7:26:42 AM		
✓	Program blocks		↗		0	0	7:26:42 AM		
✓	SPEED_MOTOR (DB2)	Block was successfully compiled.	↗				7:26:42 AM		
✓	Main (OB1)	Block was successfully compiled.	↗				7:26:44 AM		
✓		Compiling completed (errors: 0; warnings: 0)					7:26:48 AM		

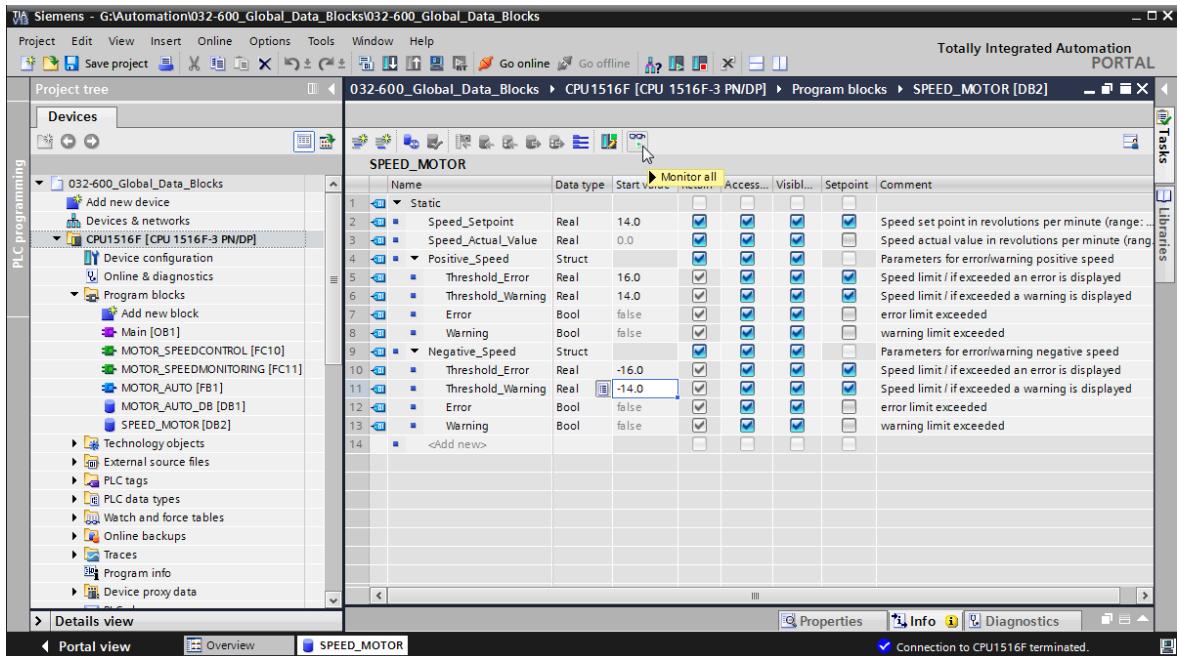
7.5 Download the program

- After successful compilation, the complete controller with the created program including the hardware configuration can, as described in the previous modules, be downloaded.



7.6 Monitor/modify values in data blocks

- The desired block must be open for monitoring the tags of a downloaded data block. The monitoring can then be activated/deactivated by clicking the icon.



- In the 'Monitor value' column, the values currently available in the CPU can be monitored.

	Name	Data type	Start value	Monitor value	Retain	Access...	Visible...	Setpoint	Comment
1	Static								
2	Speed_Setpoint	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per...
3	Speed_Actual_Value	Real	0.0	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Speed actual value in revolutions ...
4	Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Parameters for error/warning posit...
5	Threshold_Error	Real	16.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error i...
6	Threshold_Warning	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warnin...
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		warning limit exceeded
9	Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Parameters for error/warning nega...
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error i...
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warnin...
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		warning limit exceeded
14	<Add new>								

- If your right-click on one of the values, the 'Modify' dialog for modifying this value opens
(→ Modify → Modify value: 14.0 → OK)

	Name	Data type	Start value	Monitor value	Retain	Access...	Visible...	Setpoint	Comment
1	Static								
2	Speed_Setpoint	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per...
3	Speed_Actual_Value								olutions ...
4	Positive_Speed								g posit...
5	Threshold_Error								error i...
6	Threshold_Warning								warnin...
7	Error								g nega...
8	Warning								error i...
9	Negative_Speed								
10	Threshold_Error								
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warnin...
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		warning limit exceeded
14	<Add new>								

7.7 Initialize setpoints/reset start values

- The setpoints can be initialized by clicking the icon. For the tags whose 'Setpoint' check box is selected , the start value will then be applied as the current value.



SPEED_MOTOR [DB2]										
	Name	Data type	Start value	Monitor value	Retain	Accessible...	Visible i...	Setpoint	Comment	
1	Static									
2	Speed_Setpoint	Real	14.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.)	
3	Speed_Actual_Value	Real	0.0	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (.	
4	Positive_Speed	Struct							Parameters for error/warning positive speed	
5	Threshold_Error	Real	16.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
6	Threshold_Warning	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	
9	Negative_Speed	Struct							Parameters for error/warning negative speed	
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	

SPEED_MOTOR [DB2]										
	Name	Data type	Start value	Monitor value	Retain	Accessible...	Visible i...	Setpoint	Comment	
1	Static									
2	Speed_Setpoint	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.)	
3	Speed_Actual_Value	Real	0.0	15.27055	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (.	
4	Positive_Speed	Struct							Parameters for error/warning positive speed	
5	Threshold_Error	Real	16.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
6	Threshold_Warning	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	
9	Negative_Speed	Struct							Parameters for error/warning negative speed	
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	

→ All start values can be reset by clicking the  icon.



SPEED_MOTOR [DB2]										
	Name	Data type	Start value	Monitor value	Retain	Accessible...	Visible i...	Setpoint	Comment	
1	Static									
2	Speed_Setpoint	Real	14.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.)	
3	Speed_Actual_Value	Real	0.0	15.27055	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (.	
4	Positive_Speed	Struct							Parameters for error/warning positive speed	
5	Threshold_Error	Real	16.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
6	Threshold_Warning	Real	14.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	
9	Negative_Speed	Struct							Parameters for error/warning negative speed	
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	

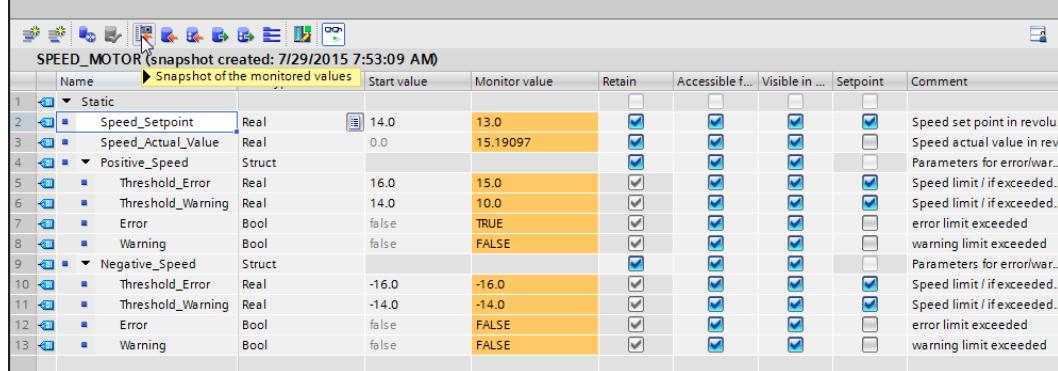
SPEED_MOTOR [DB2]										
	Name	Data type	Start value	Monitor value	Retain	Accessible...	Visible i...	Setpoint	Comment	
1	Static									
2	Speed_Setpoint	Real	0.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.)	
3	Speed_Actual_Value	Real	0.0	15.27055	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (.	
4	Positive_Speed	Struct							Parameters for error/warning positive speed	
5	Threshold_Error	Real	0.0	16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
6	Threshold_Warning	Real	0.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
7	Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	
9	Negative_Speed	Struct							Parameters for error/warning negative speed	
10	Threshold_Error	Real	0.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed	
11	Threshold_Warning	Real	0.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed	
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded	
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded	

7.8 Snapshots in data blocks

- If you click the  icon, a snapshot of the monitored values can be taken in order to apply these values as start values or to transfer them back to the CPU later

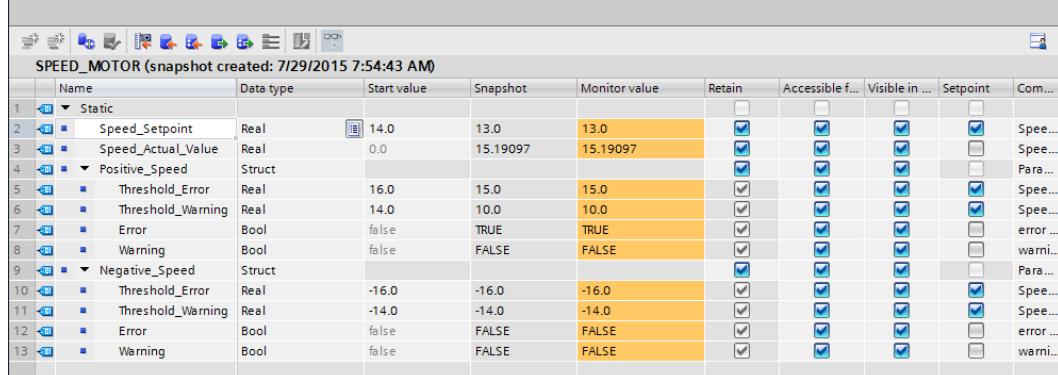
(→ ).

032-600_Global_Data_Blocks > CPU1516F [CPU 1516F-3 PN/DP] > Program blocks > SPEED_MOTOR [DB2]



Name	Start value	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1 Static							
2 Speed_Setpoint	Real 14.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in rev...
3 Speed_Actual_Value	Real 0.0	15.19097	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in rev...
4 Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/war...
5 Threshold_Error	Real 16.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
6 Threshold_Warning	Real 14.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
7 Error	Bool false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8 Warning	Bool false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9 Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/war...
10 Threshold_Error	Real -16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
11 Threshold_Warning	Real -14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded...
12 Error	Bool false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13 Warning	Bool false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded

032-600_Global_Data_Blocks > CPU1516F [CPU 1516F-3 PN/DP] > Program blocks > SPEED_MOTOR [DB2]

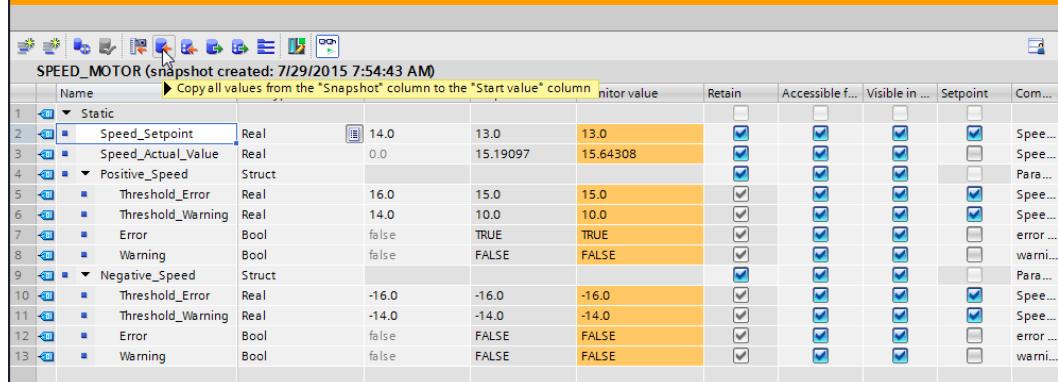


Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1 Static									
2 Speed_Setpoint	Real 14.0	13.0	13.0	<input checked="" type="checkbox"/>	Spe...				
3 Speed_Actual_Value	Real 0.0	15.19097	15.19097	<input checked="" type="checkbox"/>	Spe...				
4 Positive_Speed	Struct			<input checked="" type="checkbox"/>	Para...				
5 Threshold_Error	Real 16.0	15.0	15.0	<input checked="" type="checkbox"/>	Spe...				
6 Threshold_Warning	Real 14.0	10.0	10.0	<input checked="" type="checkbox"/>	Spe...				
7 Error	Bool false	TRUE	TRUE	<input checked="" type="checkbox"/>	error ...				
8 Warning	Bool false	FALSE	FALSE	<input checked="" type="checkbox"/>	warn...				
9 Negative_Speed	Struct			<input checked="" type="checkbox"/>	Para...				
10 Threshold_Error	Real -16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	Spe...				
11 Threshold_Warning	Real -14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	Spe...				
12 Error	Bool false	FALSE	FALSE	<input checked="" type="checkbox"/>	error ...				
13 Warning	Bool false	FALSE	FALSE	<input checked="" type="checkbox"/>	warn...				

- Alternatively, values from the snapshot can be applied by clicking the  icon for all values or by clicking the  icon for the start values only. Only the setpoints are needed here in most cases

(→ )

032-600_Global_Data_Blocks > CPU1516F [CPU 1516F-3 PN/DP] > Program blocks > SPEED_MOTOR [DB2]



Name	Start value	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1 Static							
2 Speed_Setpoint	Real 14.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spe...
3 Speed_Actual_Value	Real 0.0	15.19097	15.64308	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spe...
4 Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
5 Threshold_Error	Real 16.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spe...
6 Threshold_Warning	Real 14.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spe...
7 Error	Bool false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
8 Warning	Bool false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warn...
9 Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
10 Threshold_Error	Real -16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spe...
11 Threshold_Warning	Real -14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spe...
12 Error	Bool false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
13 Warning	Bool false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warn...

Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)									
1	Static								
2	Speed_Setpoint	Real	14.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	Speed_Actual_Value	Real	0.0	15.19097	15.64308	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	Threshold_Error	Real	16.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
6	Threshold_Warning	Real	14.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)									
1	Static								
2	Speed_Setpoint	Real	13.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	Speed_Actual_Value	Real	0.0	15.19097	15.64308	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

- To transfer data captured in the snapshot back to the CPU, you must click .
- (→ 

Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)									
1	Static								
2	Speed_Setpoint	Real	13.0	13.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)									
1	Static								
2	Speed_Setpoint	Real	13.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

- If you want to overwrite all setpoints with the start values, you can initiate this by clicking ''. The values in the CPU for which the 'Setpoint' check box was not selected are thereby retained.

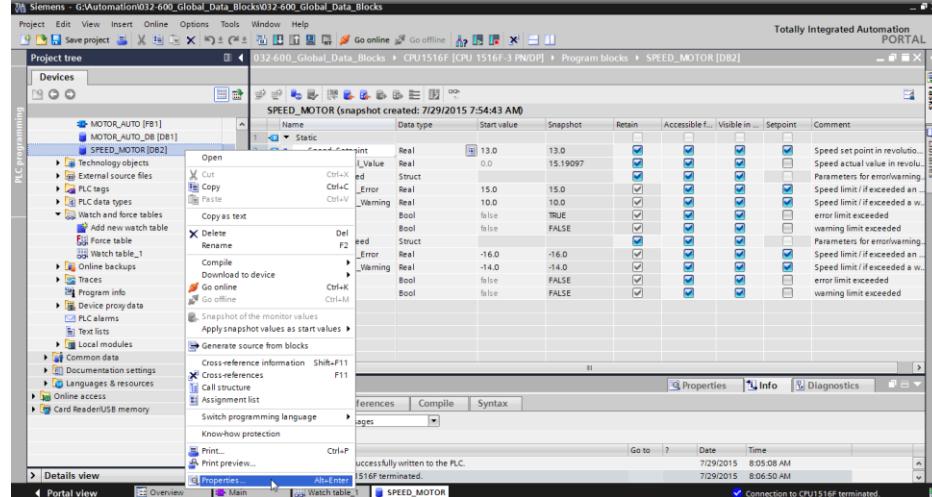
(→ 

SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)										
	Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1	Static									
2	Speed_Setpoint	Real	13.0	13.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...

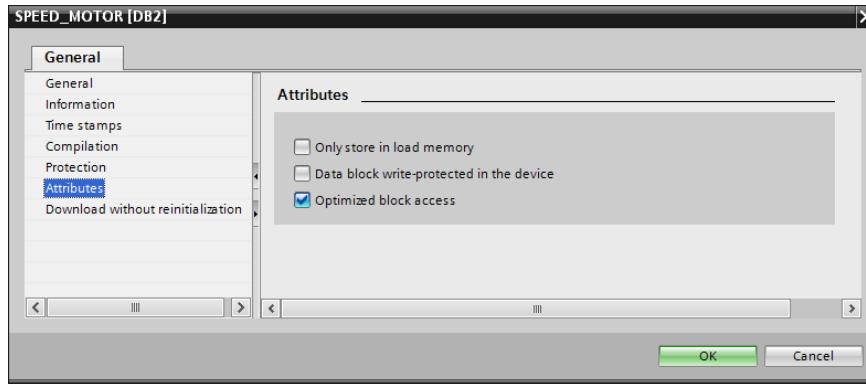
SPEED_MOTOR (snapshot created: 7/29/2015 7:54:43 AM)										
	Name	Data type	Start value	Snapshot	Monitor value	Retain	Accessible f...	Visible in ...	Setpoint	Com...
1	Static									
2	Speed_Setpoint	Real	13.0	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
3	Speed_Actual_Value	Real	0.0	15.19097	15.06981	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
4	Positive_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
5	Threshold_Error	Real	15.0	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
6	Threshold_Warning	Real	10.0	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
7	Error	Bool	false	TRUE	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
8	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...
9	Negative_Speed	Struct				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Para...
10	Threshold_Error	Real	-16.0	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
11	Threshold_Warning	Real	-14.0	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spee...
12	Error	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error ...
13	Warning	Bool	false	FALSE	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warni...

7.9 Expand data block and download it without reinitialization

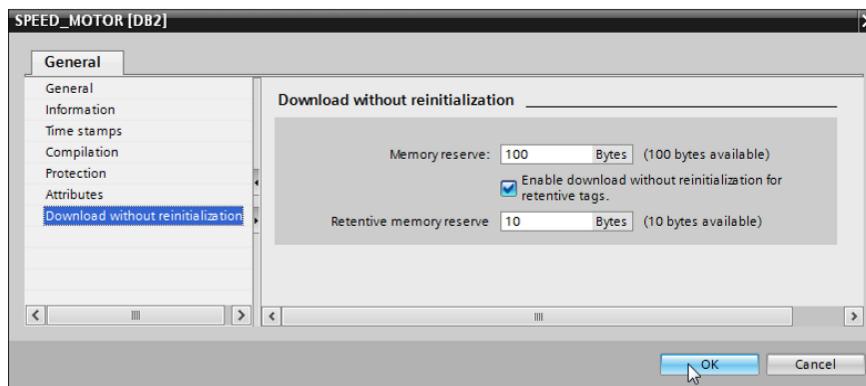
- To enable 'Download without reinitialization' for the "SPEED_MOTOR" [DB2] data block, you must ' Go offline' and then open the properties of the data block.
 (→  Go offline → SPEED_MOTOR [DB2] → Properties)

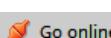
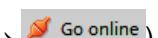


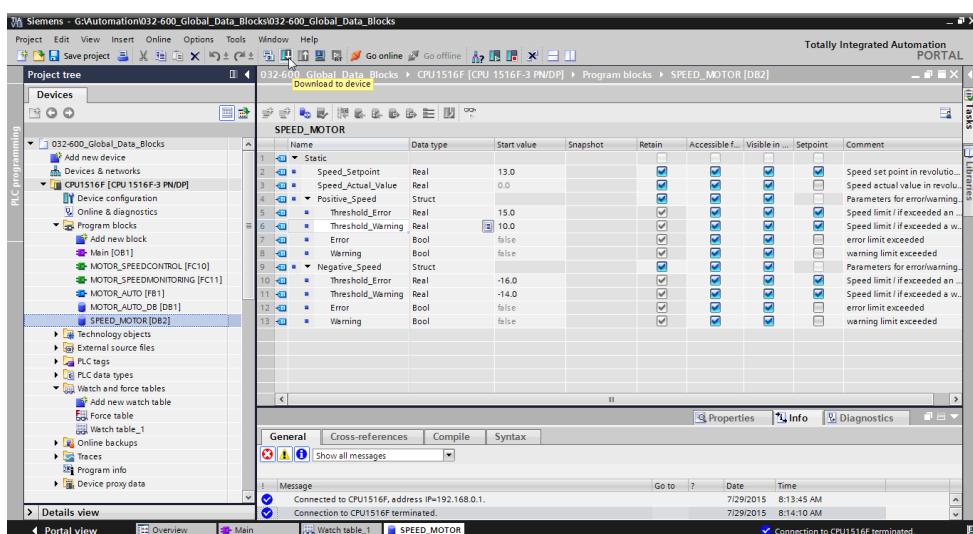
- Select the 'Optimized block access' check box in the properties under 'General', 'Attributes'.
 (→ General → Attributes → Optimized block access)



- Assign a 'Retentive memory reserve' to the data block for 'Download without reinitialization'.
 (→ Download without reinitialization → Retentive memory reserve → 10 bytes → OK)

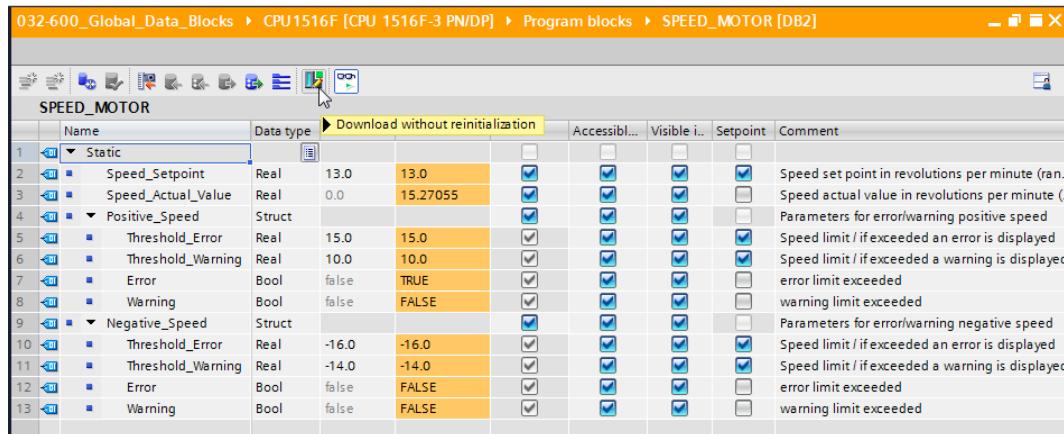


- Download your "SPEED_MOTOR" [DB] data block to the controller again and select ' Go online'.
- (→ SPEED_MOTOR [DB] →  → 

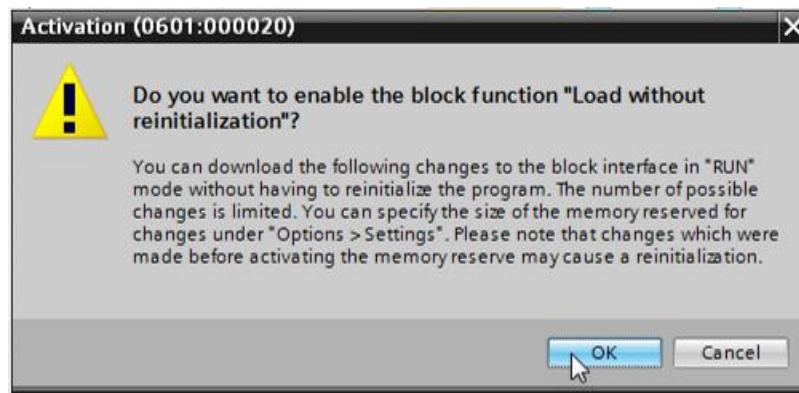


- Then click the '' icon to activate download without reinitialization and confirm the safety prompt with 'OK'.

(→  → OK)

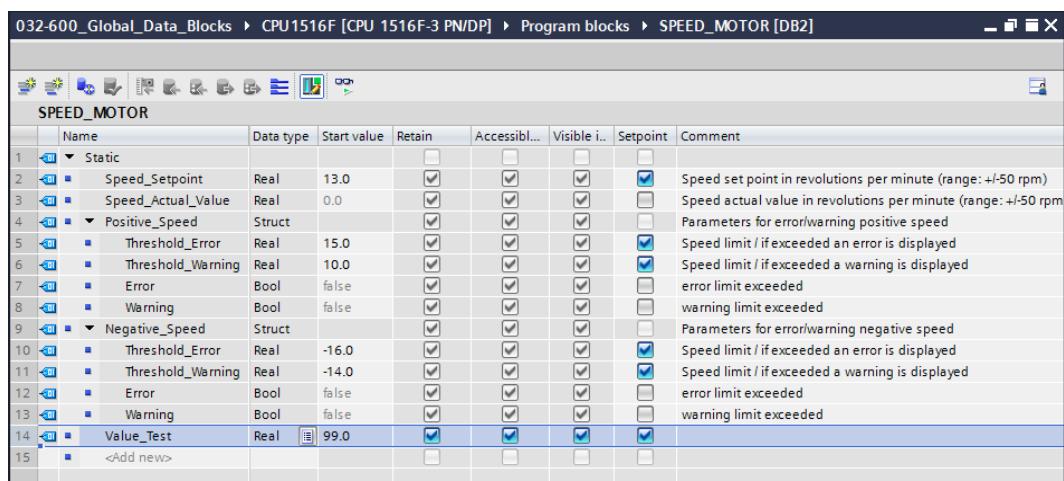


Name	Data type	Value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1 Static							
2 Speed_Setpoint	Real	13.0	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran.
3 Speed_Actual_Value	Real	0.0	15.27055	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (.
4 Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5 Threshold_Error	Real	15.0	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6 Threshold_Warning	Real	10.0	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7 Error	Bool	false	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8 Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9 Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning negative speed
10 Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11 Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12 Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13 Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
14 Value_Test	Real	99.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
15 <Add new>							



- Next add any tag in your data block

(→ Name: Value_test → Data type: Real → Start value: 99)



Name	Data type	Start value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1 Static							
2 Speed_Setpoint	Real	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
3 Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
4 Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning positive speed
5 Threshold_Error	Real	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6 Threshold_Warning	Real	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7 Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8 Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9 Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for error/warning negative speed
10 Threshold_Error	Real	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11 Threshold_Warning	Real	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12 Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13 Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
14 Value_Test	Real	99.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
15 <Add new>							

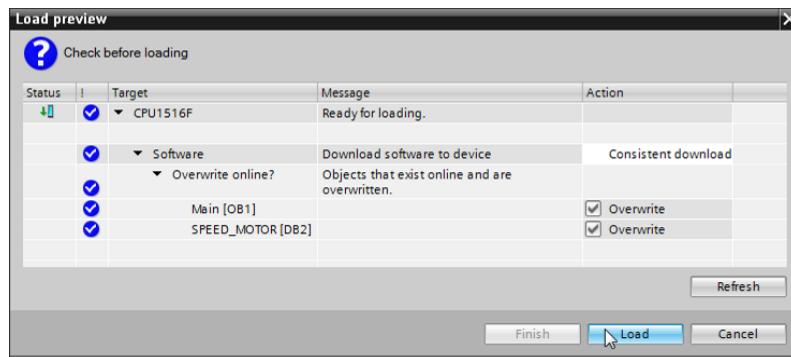
- Download your "SPEED_MOTOR" [DB] data block to the controller again.

(→ SPEED_MOTOR [DB] →  → Download)

The screenshot shows the SIMATIC Manager interface for a project titled "032-600_Global_Data_Blocks". The left sidebar is titled "PLC programming" and contains a tree view of the project structure, including "Program blocks" which is currently selected. The main workspace displays the "SPEED_MOTOR [DB2]" block under "Program blocks". The properties window at the bottom shows the following details:

Name	Data type	Start value	Retain	Accessible...	Visible i...	Setpoint	Comment
1 Static	Real	13.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (range: +/-50 rpm)
2 Speed_Setpoint	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (range: +/-50 rpm)
3 Speed_Actual_Value	Real	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for errorwarning positive speed
4 Positive_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
5 Threshold_Error	Real	15.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
6 Threshold_Warning	Real	10.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
7 Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
8 Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9 Negative_Speed	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for errorwarning negative speed
10 Threshold_Error	Real	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11 Threshold_Warning	Real	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12 Error	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13 Warning	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
14 Value_Test	Real	99.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<Add new>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

The status bar at the bottom right indicates "Connected to CPU1516F, address IP=192.168.0.1...".



- If you click  to monitor the block again, you will see that the monitored values have not been overwritten with the start values.



	Name	Data type	Start value	Monitor value	Retain	Accessibl...	Visible i...	Setpoint	Comment
1	Static								
2	Speed_Setpoint	Real	13.0	14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed set point in revolutions per minute (ran...
3	Speed_Actual_Value	Real	0.0	0.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed actual value in revolutions per minute (...
4	Positive_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for overwarning positive speed
5	Threshold_Error	Real	15.0	17.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
6	Threshold_Warning	Real	10.0	12.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
7	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
8	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
9	Negative_Speed	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Parameters for overwarning negative speed
10	Threshold_Error	Real	-16.0	-16.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded an error is displayed
11	Threshold_Warning	Real	-14.0	-14.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Speed limit / if exceeded a warning is displayed
12	Error	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	error limit exceeded
13	Warning	Bool	false	FALSE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	warning limit exceeded
14	Value_Test	Real	99.0	99.0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

8 Checklist

No.	Description	Completed
1	Data block SPEED_MOTOR [DB2] successfully created.	
2	Program changes made in Main [OB1].	
3	Compiling successful and without error message	
4	Download successful and without error message	
5	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
6	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
7	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
8	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
9	Manual mode (-S0 = 0) → -Q3 = 0	
10	Switch off station (-K0 = 0) → -Q3 = 0	
11	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
12	Speed > Motor_speed_monitoring_error_max → -Q3 = 0	
13	Speed < Motor_speed_monitoring_error_min → -Q3 = 0	
14	Project successfully archived	



Training Curriculum

TIA Portal Module 011
WinCC Advanced with
TP700 Comfort and SIMATIC S7-1500

Table of contents

1	Goal.....	338
2	Prerequisite.....	338
3	Required hardware and software.....	338
4	Theory	340
4.1	Process visualization	340
4.2	SIMATIC HMI Panel TP700 Comfort.....	340
4.2.1	Device description.....	340
4.2.2	Memory concept	342
4.2.3	Settings on the Touch Panel TP700 Comfort/Start Center.....	344
4.2.4	Setting the date and time	345
4.2.5	Setting transfer properties and assigning the IP address	345
4.2.6	Calibrating the touch panel and rebooting	347
4.3	WinCC Advanced V1X programming software (TIA Portal V1X).....	349
4.3.1	Project	349
4.3.2	Hardware configuration	349
4.3.3	Schedule the hardware	350
4.3.4	Planning the screen structure.....	351
4.3.5	Planning of the screen structure.....	351
4.3.6	Basic settings for WinCC Advanced in the TIA Portal	352
4.3.7	Resetting the SIMATIC HMI Panel TP700 Comfort and setting the IP address	353
4.3.8	WinCC user interface	356
4.3.9	Project tree	356
4.3.10	Details view	357
4.3.11	Menu bar and buttons	357
4.3.12	Work area.....	357
4.3.13	Tools	358
4.3.14	Properties window.....	358
4.3.15	Other tabs.....	359
5	Task.....	360
6	PlanningProcess visualization	360
6.1	Program description for the sorting station with speed control and speed monitoring of the motor	360
6.2	Technology diagram	361
6.3	Reference list	362
7	Structured step-by-step instructions.....	363
7.1	Retrieving an existing project	363
7.2	Adding a SIMATIC HMI Panel TP700 Comfort	364
7.3	HMI wizard for the TP700 Comfort Panel	366
7.4	Device configuration of the TP700 Comfort Panel	370

7.4.1	Setting the IP address.....	371
7.5	Compiling the CPU and panel and saving the project.....	372
7.6	Configuring the graphic display	373
7.7	Displaying a process value in an IO field.....	376
7.8	Visualizing binary signals with animated rectangles	379
7.9	Symbol library.....	382
7.10	Connections and HMI tags.....	385
7.11	.Downloading the CPU and panel	387
7.12	Process visualization in the simulation	390
7.13	Switches and buttons for the process operation	391
7.14	Adapting the headers in the template	405
7.15	Bar graph	415
7.16	Alarms.....	421
7.16.1	General alarm settings.....	421
7.16.2	Alarm window	422
7.16.3	Alarm indicator.....	424
7.16.4	Alarms for system diagnostics of CPU 1516F.....	425
7.16.5	Alarm class settings	427
7.16.6	System events	427
7.16.7	Controller alarms.....	428
7.16.8	Analog alarms	428
7.16.9	Discrete alarms.....	429
7.17	Remote control of the TP700 Comfort Panel	433
7.17.1	Activating web services for Runtime	433
7.17.2	WinCC Internet settings in the TP700 Comfort Panel	434
7.17.3	Starting remote access to the TP700 Comfort Panel	436
8	Checklist	438

PROCESS VISUALIZATION WITH TP 700

1 Goal

In this chapter, you will become acquainted with the basics of process visualization and the use of a SIMATIC HMI Panel TP700 Comfort together with the SIMATIC S7-1500 and the TIA Portal programming tool.

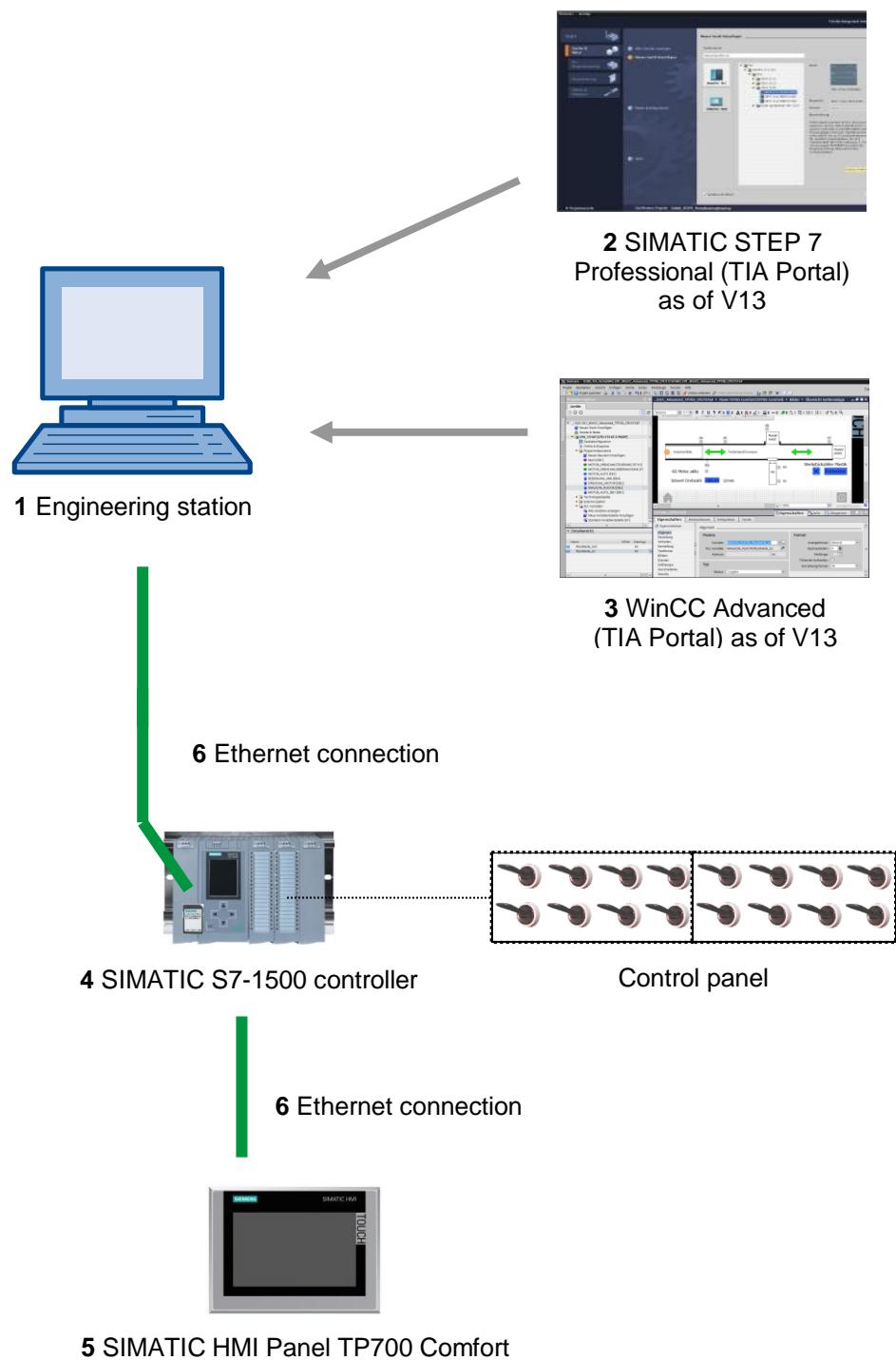
The module explains the configuring of a SIMATIC HMI Panel TP700 Comfort, the creating of the connection to the SIMATIC S7-1500 and the read and write access to CPU data from the SIMATIC HMI Panel TP700 Comfort.

2 Prerequisite

This chapter builds on the chapter "Global data blocks with SIMATIC S7 CPU 1516F-3 PN/DP". You can use the following project for this chapter, for example: Data Block.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 WinCC Advanced software in TIA Portal – as of V1X
- 4 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs and analog inputs and outputs should be fed out to a panel.
- 5 SIMATIC HMI Panel TP700 Comfort
- 6 Ethernet connection between engineering station and controller and
between controller and TP700 Comfort control panel



4 Theory

4.1 Process visualization

Production processes are becoming more and more complex and greater demands are being placed on the functionality of machinery and plants. For this reason, the operator needs a high-performance tool to control and monitor production plants. An HMI (Human Machine Interface) system represents the interface between the human being (operator) and the process (machine/plant). The controller has the actual control over the process. There is therefore an interface between the operator and WinCC (on the HMI panel) and an interface between WinCC and the controller.

SIMATIC HMI Comfort Panels and WinCC undertake the following tasks:

- **Representing processes with a clear screen structure**

The process is represented on the HMI device. If, for example, a state changes in the process, the display is updated on the HMI device. The process can be represented clearly structured on multiple screens.

- **Operating processes**

The operator can operate the process using the graphical user interface. The operator can, for example, enter a setpoint for the controller or start a motor.

- **Outputting alarms**

If critical process states occur in the process, an alarm is triggered automatically; for example, when a specified limit is exceeded.

- **Logging process values and alarms**

The HMI system can log alarms and process values. You can document the process history in this way. As a result, you still have access to older production data even at a later time.

- **Documenting process values and alarms**

The HMI system can print out alarms and process values as reports. For example, you can output the production data at the end of a shift.

- **Managing process parameters and machine parameters in recipes**

The HMI system can store parameters for processes and machines in recipes. You can transfer these parameters, for example, from the HMI device to the controller in a single step in order to change production to another product type.

- **User management**

Certain rights can be granted to the devices and the possible operator inputs can be limited for certain users.

4.2 SIMATIC HMI Panel TP700 Comfort

4.2.1 Device description

The SIMATIC HMI Comfort Panels product line includes touch panels (operated by a touch screen), key panels (operated by a keyboard) and key & touch panels (operated by a keyboard and touch screen).

The SIMATIC HMI Comfort Panels cover all requirements described in Chapter 3.1. The following is also optionally possible:

- Support for operation through help texts
- User-specific expansion of functionality through VBScript

- Microsoft Excel/Word/PDF Viewer for displaying documents
- Remote access to the user interface of the Comfort Panel via Ethernet from the web browser of another HMI device or any PC through the WinCC/Sm@rtServer option
- Recording of operations in an audit trail with electronic signature through the WinCC/Audit option
- Uninterruptible power supply (UPS) with USB support

This document explains these HMI devices using the TP700 Comfort as an example.



Figure 1: TP700 Comfort

The WinCC Advanced V1X (TIA Portal V1X) software is required for configuration and programming. This software is included in the product package of the SCE Trainer Packet "**SIMATIC HMI TP700 COMFORT PANEL**"!

Notes:

Because all devices in this series have similar functions, it would also be possible to work through the chapters of this document with a different device version in this series.

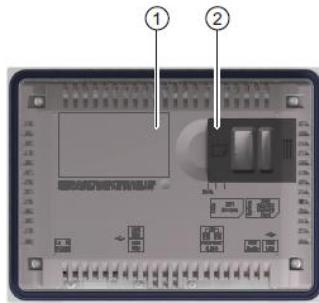
WinCC Advanced Runtime Simulation can also be used to represent the TP700 Touch Panel on the PC (included in the product package of the SCE Trainer Package SIMATIC HMI TP700 COMFORT PANEL).

Front view of the TP700 Comfort



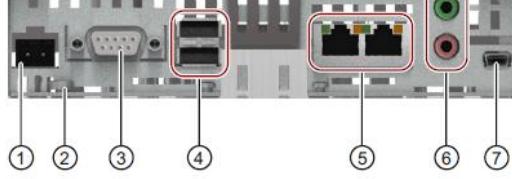
③ Touchscreen display

Rear view of the TP700 Comfort



- ① Rating plate
 ② Slots for SD memory cards

Interfaces of the TP700 Comfort



- ① Connection for power supply
 ② Connection for equipotential bonding (ground)
 ③ PROFIBUS (Sub-D RS422/485)
 ④ 2x USB type A
 ⑤ PROFINET (LAN), 10/100 Mbit, 2 ports
 ⑥ Audio Line IN/OUT
 ⑦ USB type Mini-B

4.2.2 Memory concept

The HMI devices can use the following types memory:

- Internal memory
- System memory card
- Memory card
- USB mass storage connected to the USB port

Internal memory

The following data is stored here:

- Operating system
- Project file
- License keys
- User management
- Recipes

Note: Cyclic write access is not permitted for the internal memory because this reduces the life expectancy of the internal memory and thus the service life of the HMI device.

You should preferably use external memory cards, such as the SIMATIC HMI Memory Card, if possible, in order to prolong the service life of the HMI device for the storage of data records and for logs.

Memory card

The following data is stored here:

- Logs
- Backups
- Recipes

You can use commercially available memory cards in "SD(IO / HC)" or "MMC" format as the memory card. For reasons of data consistency, use of a SIMATIC HMI Memory Card as the memory card is recommended.

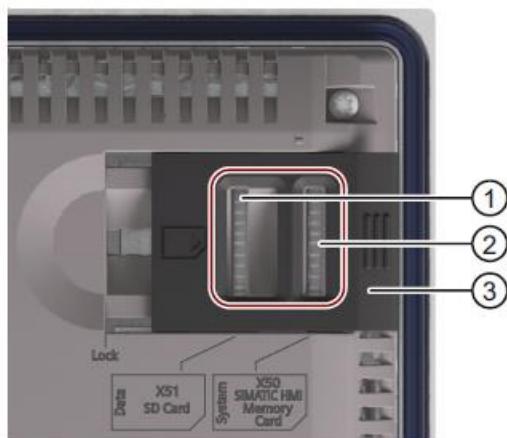
System memory card

The system memory card is part of the service concept of the HMI devices and helps to reduce downtimes.

When you activate the service concept, all data is transferred from the internal memory of the HMI device to the system memory card. In this way, if the HMI device fails, the system memory card can simply be inserted into the replacement device.

Only the SIMATIC HMI Memory Card with 2 GB or more memory is permitted as the system memory card. Other memory cards are not recognized by the HMI device as a system memory card.

Slots for memory card and system memory card



① Slot for memory card in "SD(IO / HC)" or "MMC" format

② Slot for system memory card

③ Locking slide

4.2.3 Settings on the Touch Panel TP700 Comfort/Start Center

A few important settings must be made directly on the Touch Panel TP700 Comfort.

The Touch Panel TP700 runs on the Windows CE operating system and can be operated, like all touch panels, directly on the screen. For better operation, you should use a touch pen or connect a mouse to the panel's USB port.

After the panel starts, the desktop containing the standard icons and the **'Start Center'** window appear. The version of the images present on the panel is also displayed in this window.

Buttons in the Start Center:

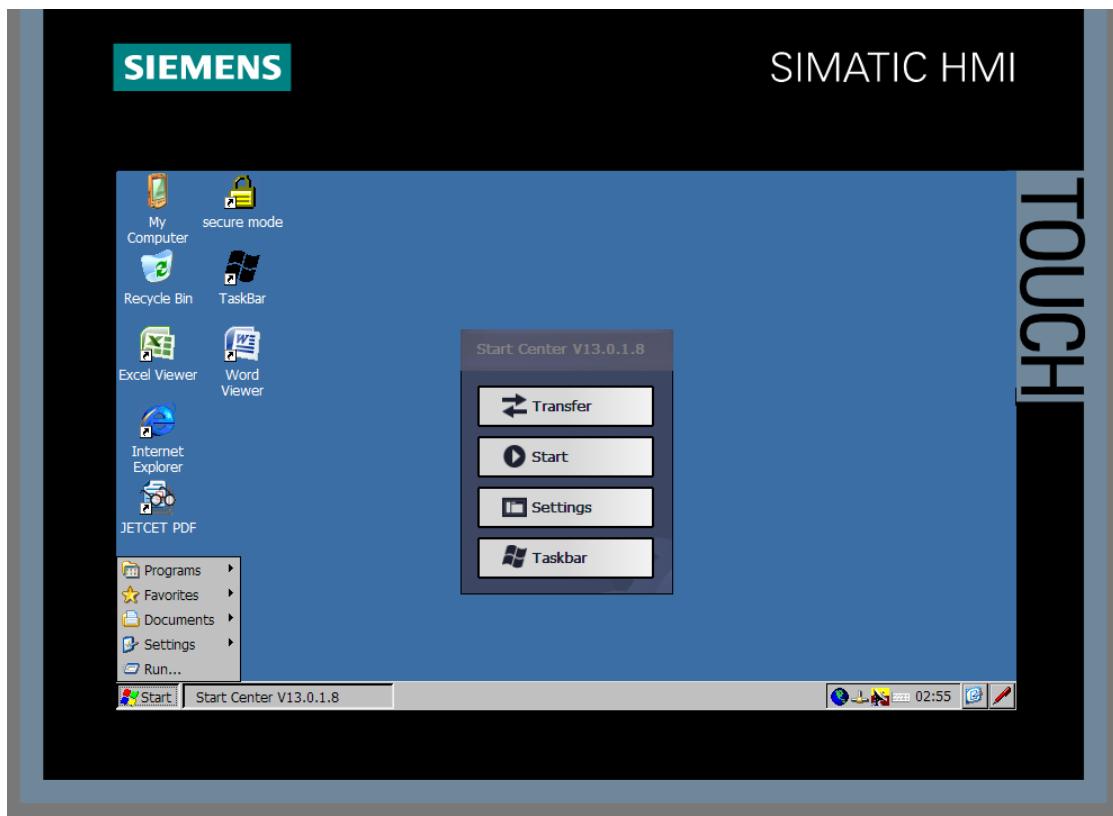
Transfer: The data transfer becomes activate (Connecting to host ...) and the panel waits for the configuration data to download from WinCC Advanced to the PC.

Start: Runtime is started and the process visualization appears on the panel. The panel is often set up so that the start occurs automatically after a few seconds.

Settings: The Windows CE settings dialog opens. Settings for the panel can be made here.

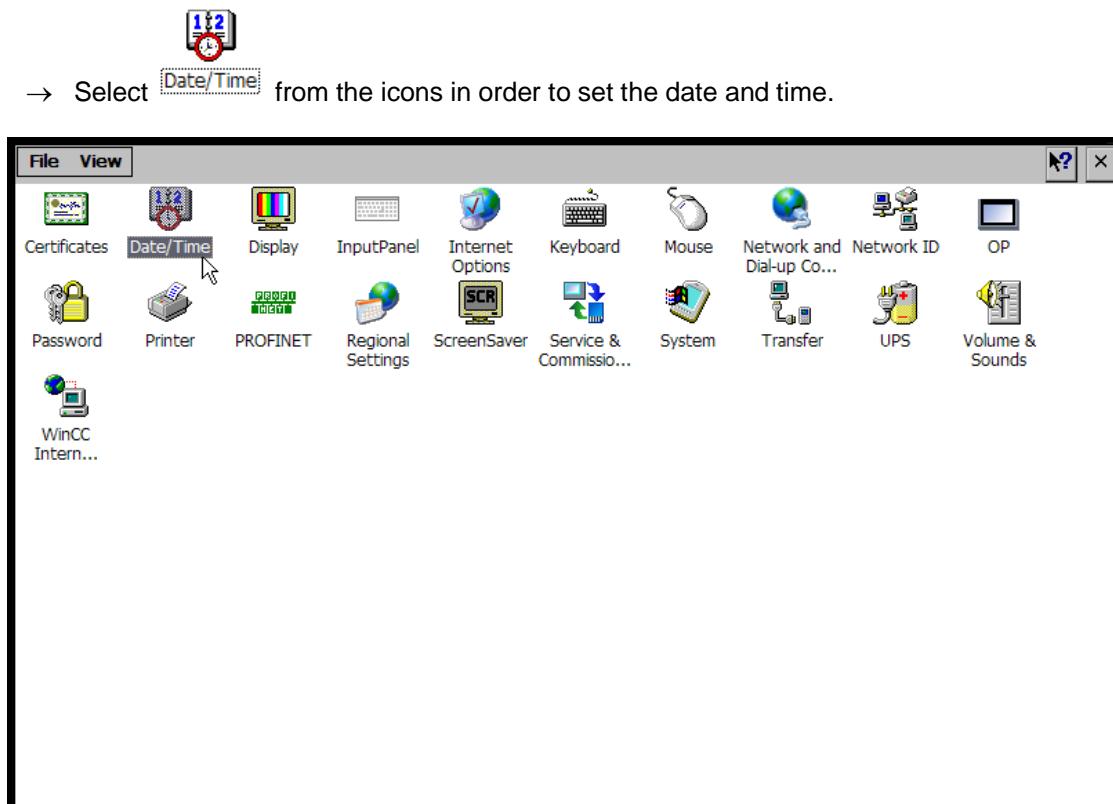
Taskbar: The Start bar of Windows CE opens.

- Select → "Settings" in the "Start Center" directly after switching on the voltage supply and the start of the panel.

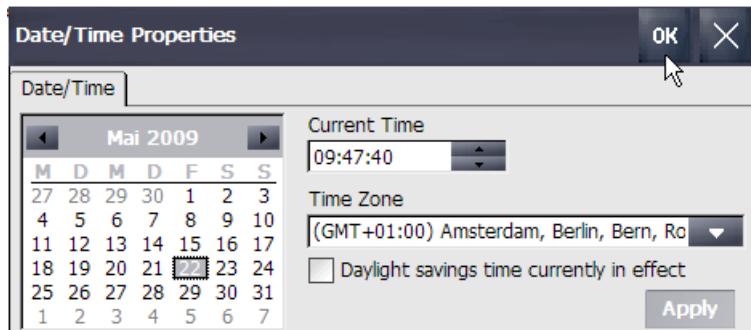


Note: The selection of "Settings" must occur fast enough, and before the automatic "Start" of Runtime.

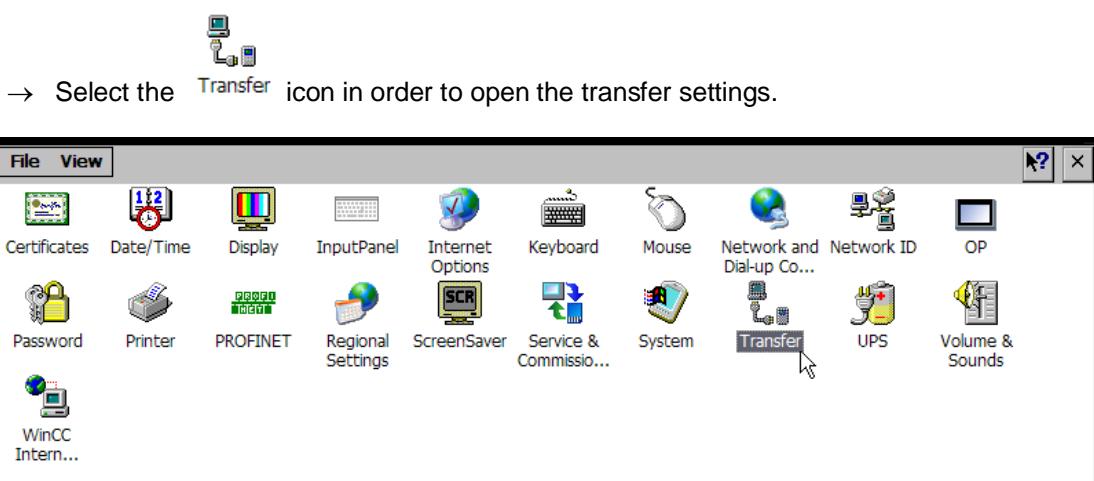
4.2.4 Setting the date and time



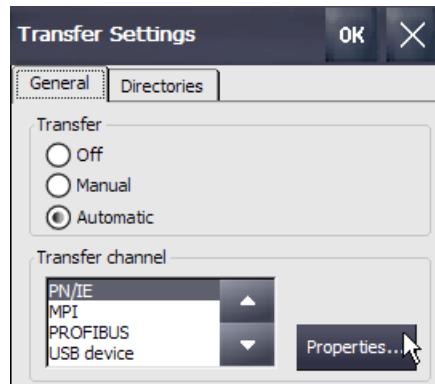
→ Select **Date/Time** from the icons in order to set the date and time.



4.2.5 Setting transfer properties and assigning the IP address



- Select "Automatic". The transfer will then be started from the WinCC Advanced software.
- Select Ethernet "PN/IE" as the transfer channel. Press the "Properties..." button → to set the IP address on the panel and to make or check the network settings.



Notes:

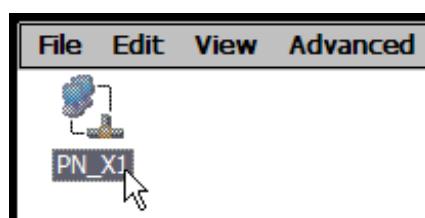
To set the MPI or PROFIBUS address on the panel, select the MPI or PROFIBUS interface and then press the Properties button to make or check the bus settings.

The settings in the Transfer Settings window have nothing to do with the connection settings in the project. This means, for example, that data can be transferred between the TP700 panel and WinCC Advanced using the Ethernet interface, and communication between the panel and the SIMATIC S7 controller can take place via the PROFIBUS interface.

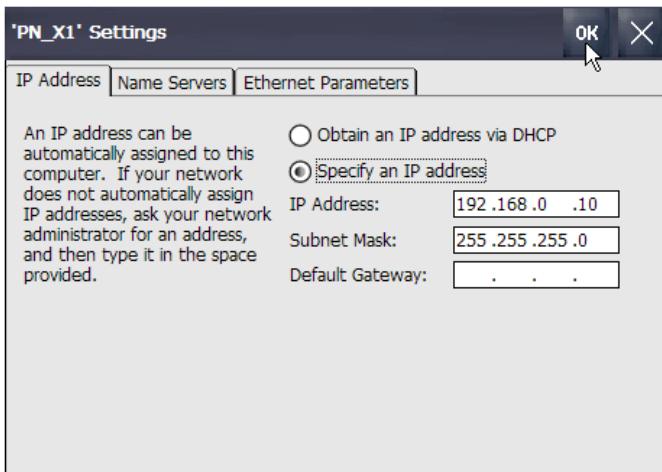
In our example, data is transferred between the TP700 panel and WinCC Advanced and communication takes place between the panel and the SIMATIC S7 controller via the Ethernet interface.



- Double-click the Ethernet interface **PN_X1** in the network settings to select it.



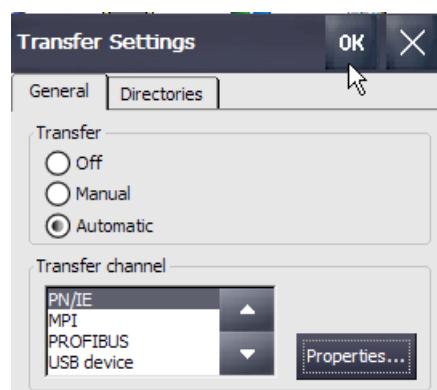
- In the "IP Address" tab, select the "Specify an IP address" option and set the IP address and subnet mask. → Close and confirm the dialog with "OK".



→ Click to close the network settings.



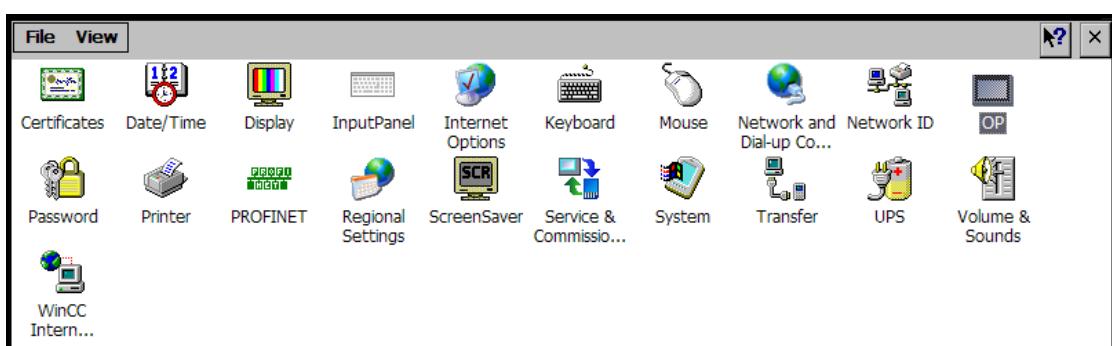
→ Click "OK" to accept the transfer settings.



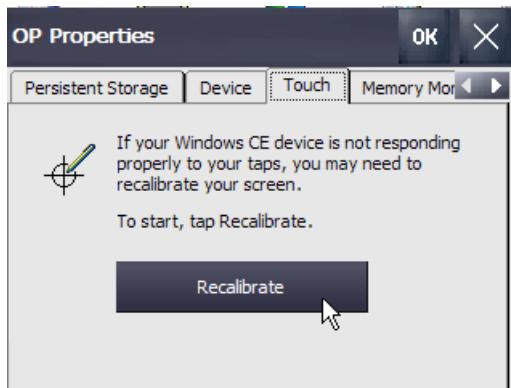
4.2.6 Calibrating the touch panel and rebooting



→ Select the icon in order to open the basic settings of the touch panel.



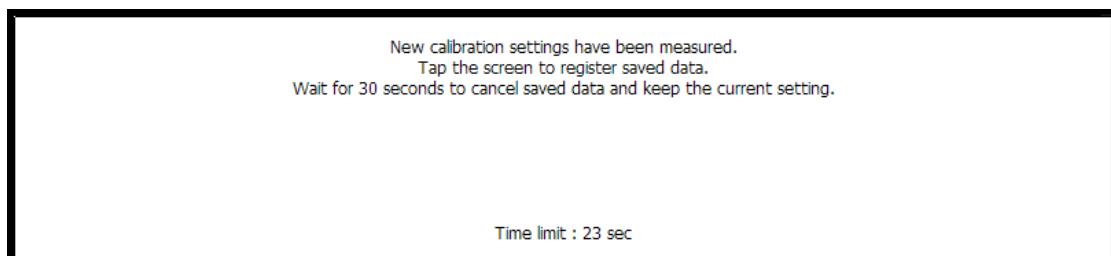
→ Select the "Touch" tab → Start the calibration there with "Recalibrate".



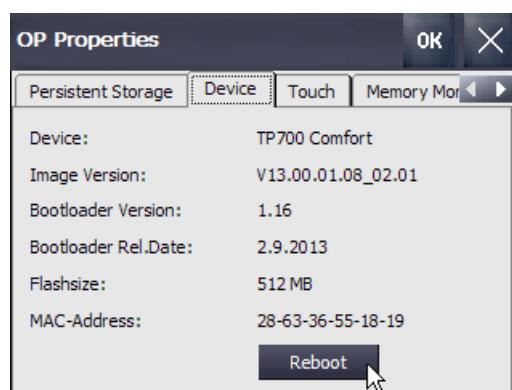
- Follow the instructions on the touch panel and press on the center of the target as accurately as possible.



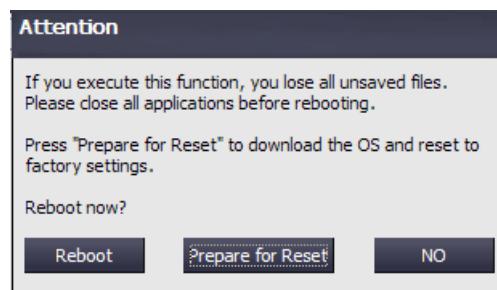
- To conclude the calibration operation, tap on the touch display of the panel again.



- Now select the "Reboot" option in the "Device" tab.



- Press the "Reboot" button.



4.3 WinCC Advanced V1X programming software (TIA Portal V1X)

The WinCC Advanced V1X software (TIA Portal V1X) is the programming tool for the following visualization systems:

- WinCC Runtime Advanced for PC-based single-user systems with licenses for 128, 512, 2k, 4k as well as 8k Power Tags (tags with a process interface)
- SIMATIC Comfort Panels
- SIMATIC Mobile Panels
- SIMATIC Multi Panels
- SIMATIC Panels of predecessor generations (Series 70, 170, 270)
- SIMATIC Basic Panels

With WinCC Advanced V1X you have the following functions for creating HMI (Human Machine Interface) systems:

- Configuration and parameter assignment of the hardware
- Specification of communication and creation of a connection to a PLC
- Creation and design of screens with hierarchical structure
- Creation of internal and external variables
- Creation of alarms and alarm views
- Creation of logs and display of logs in the form of trends and tables
- Creation of recipes and recipe views
- Creation and printing of reports
- Creation of user-defined functions with Visual Basic Scripting
- Testing, commissioning and servicing with operational/diagnostic functions
- Remote operation via the optional WinCC Smart Server
- Recording of operations in an audit trail with electronic signature through the WinCC/Audit option
- Documentation

Support is provided for all functions through detailed online help.

4.3.1 Project

To implement a solution for an automation and visualization task, you create a project in the TIA Portal. A project in the TIA Portal contains the configuration data for the configuration and networking of devices as well as the programs and the configuration of the visualization.

4.3.2 Hardware configuration

The *hardware configuration* contains the configuration of the devices consisting of the automation system hardware, the intelligent field devices and the visualization hardware, e.g.

panels as HMI (Human Machine Interface) systems. The configuration of the networks specifies the communication between the various hardware components. The individual hardware components are inserted in the hardware configuration from catalogs.

The hardware of automation systems comprises controllers (CPUs) with signal modules for input and output signals (SMs) and communication and interface modules (CP, IM). Other power supply modules (PS) and power modules (PM) are also available for supplying the modules with energy.

The signal modules and intelligent field devices connect the input and output data of the process to be automated and visualized to the automation system.

The visualization, in turn, accesses the data in the automation system.

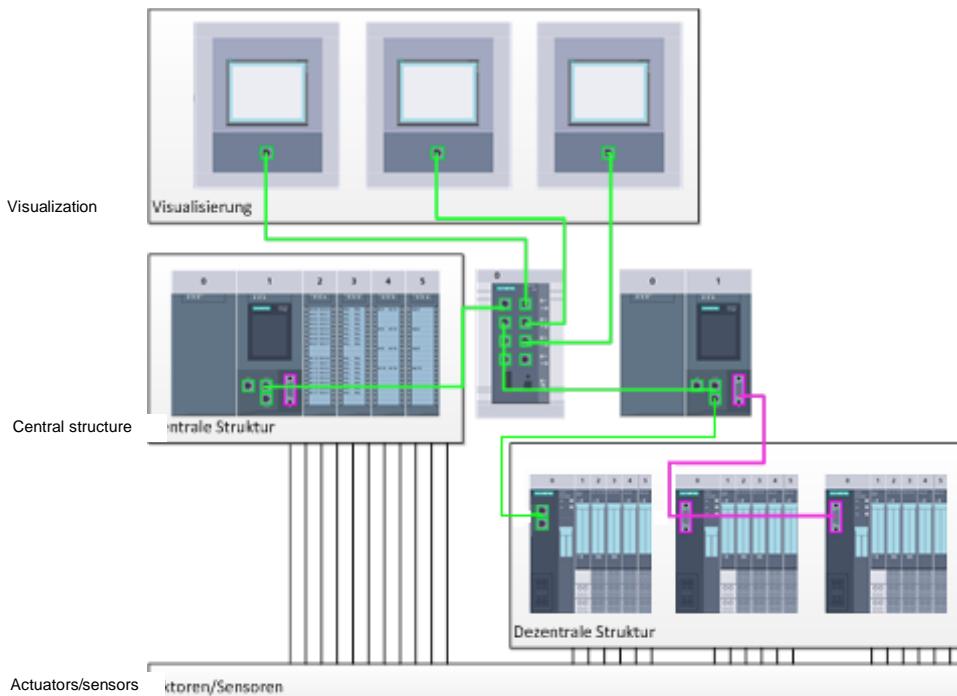


Figure 2: Example of hardware configuration with centralized and distributed structures and process visualization

The hardware configuration enables the downloading of automation and visualization solutions to the automation system and access to the connected signal modules by the controller.

4.3.3 Schedule the hardware

Before you can configure the hardware, you must plan it (hardware planning). In general, you begin by selecting which controllers are needed and how many. Next you select the communication modules and signal modules. The selection of signal modules is based on the number and type of inputs and outputs needed. As the final step, a power supply that ensures that the necessary power is supplied must be selected for each controller or field device.

The functionality required and the ambient conditions are of vital importance for planning the hardware configuration. For example, the temperature range in the application area sometimes limits the devices available for selection. Fail-safe operation might be another requirement, for example.

The [TIA Selection Tool](#) (automation technology → select TIA Selection Tool and follow the instructions) provides you support.

Notes:

- TIA Selection Tool requires Java.
- Online research: If more than one manual is available, you should look for the description "Device Manual", "Product Manual" or simply "Manual" (as opposed to

"Function Manual", "List Manual", "System Manual", etc.) in order to find the device specifications.

Figure 2 (see above) shows an automation structure that contains both centralized and distributed structures.

Centralized and distributed uses are also possible for the visualization. Panels are often used for distributed local operation. These can communicate with the controller via Ethernet, WLAN or fieldbus. For centralized operation and monitoring, PCs can also be used. They are usually connected to the controller via Ethernet.

The [TIA Selection Tool](#) (automation technology → select TIA Selection Tool and follow the instructions, requires Java) also supports you when selecting the panels.

4.3.4 Planning the screen structure

After selection of a device for the visualization, the screen structure must be planned. For this purpose, you should collect, group and structure the information to be displayed. A screen structure, such as shown by way of example in Figure 3, should be derived from this. The point of entry into the screen structure is always guaranteed by a so-called root screen.

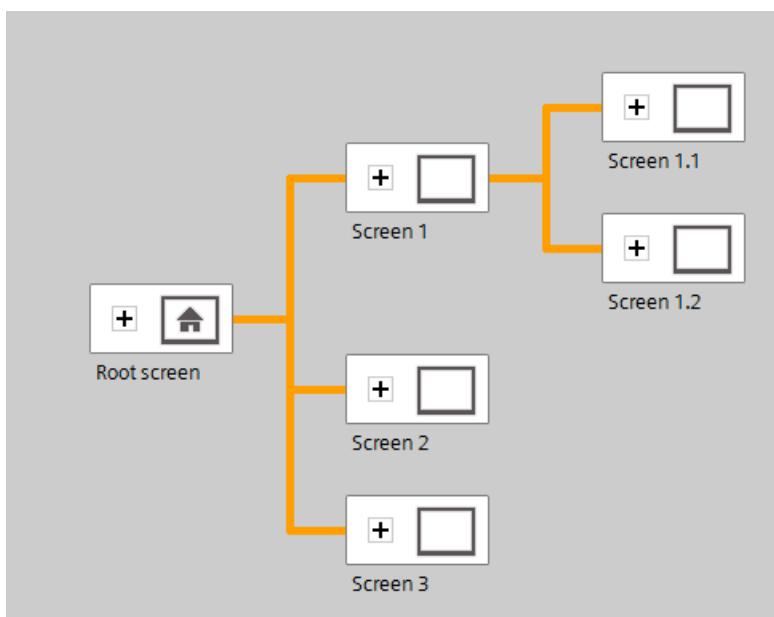


Figure 3: Example of the screen structure

The screen structure should be chosen to optimally support navigation by the user through the information distributed among the screens for operation and monitoring of the process.

The following questions can aid you in this:

Which mental model of the process should be observed for the information display?

Which data belong together?

Which data belong in which order?

Which data belong to which actions/processes?

Is there cross-action data and the like?

Which data are key data, which are supplementary data?

4.3.5 Planning of the screen structure

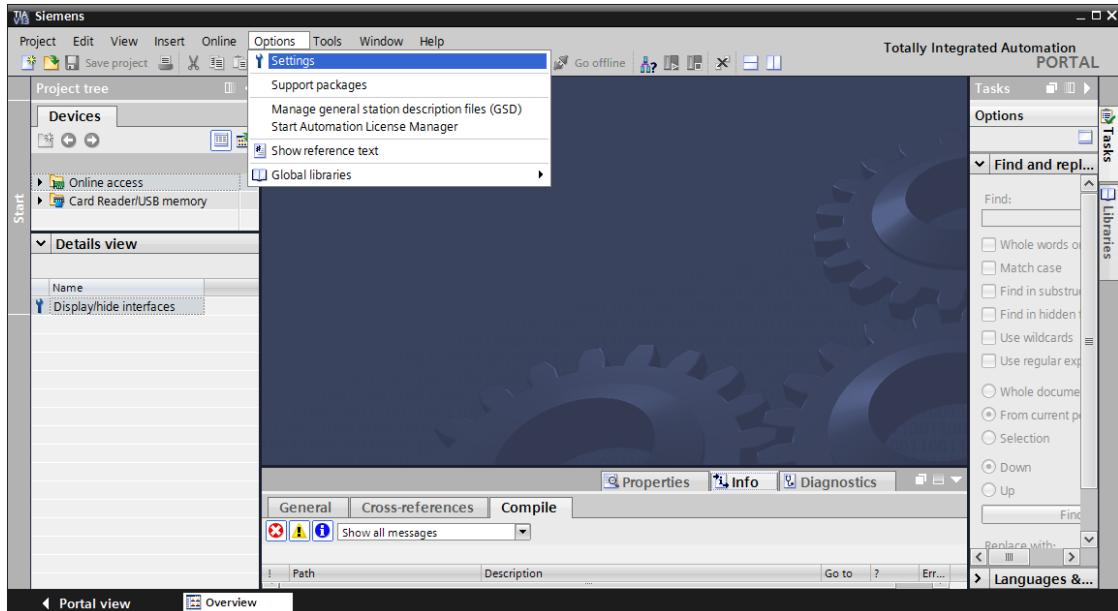
Every individual screen must be planned. For the information display, its use by people must also be considered. It is helpful here to observe design principles such as the law of proximity, law of similarity and law of symmetry. The following rules of thumb derived from the design principles can also aid in structuring screens:

- Form groups of data blocks
- Uniform division of the overall screen into work information, status or system information and controller information
- Observe the average distribution of attention on the screen as a function of reading direction.
- Use alignment as a design principle (align numbers, column headings same as column content)
- Make effective use of a maximum of 30 – 40% of the available space: place as little information as possible and as much as necessary
- Use codings sparingly (for example, color, bold text, brightness, shape, outline, pattern, flashing)
- Subdivide numbers: Subdivide numbers with more than 4 digits in groups of 2, 3 or 4 (for example, 66 234)
- Select numbers preferentially when listing objects, properties, etc.
- Use and position designations uniformly, use short words if possible

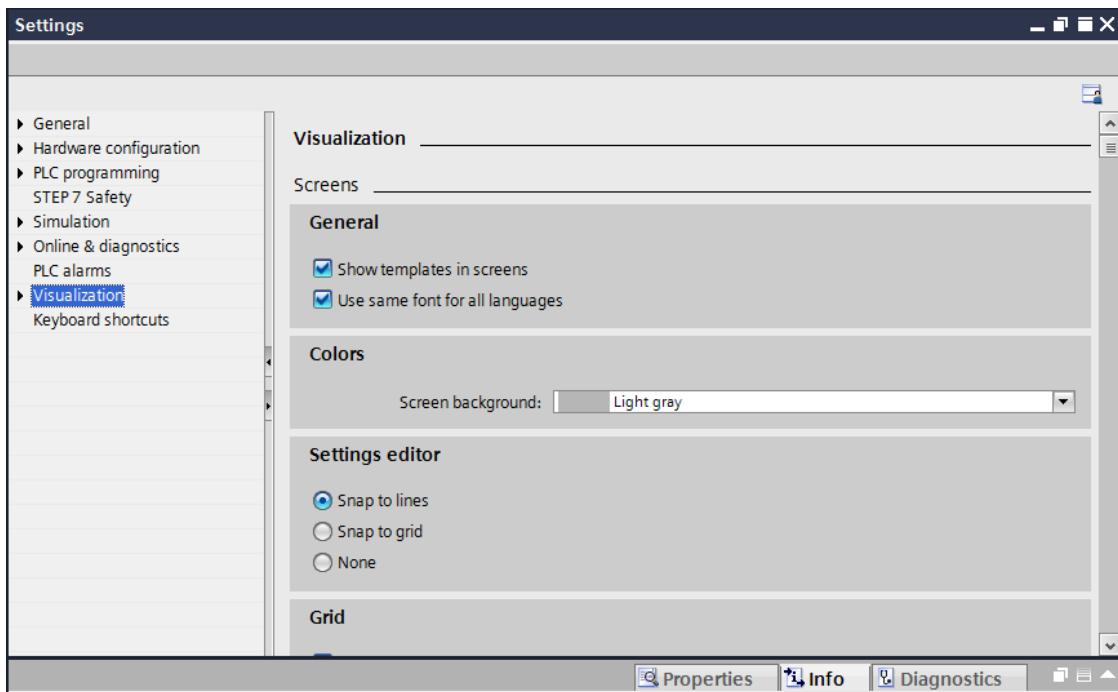
4.3.6 Basic settings for WinCC Advanced in the TIA Portal

Users can specify their own default settings for certain settings in the TIA Portal. The path to the settings for the visualization is shown here.

- In the project view, select the → "Options" menu and then → "Settings".



- Under → "Visualization" in "Settings", select the desired default settings for the appearance of the user interface.



Note: Keep the default settings for the visualization here.

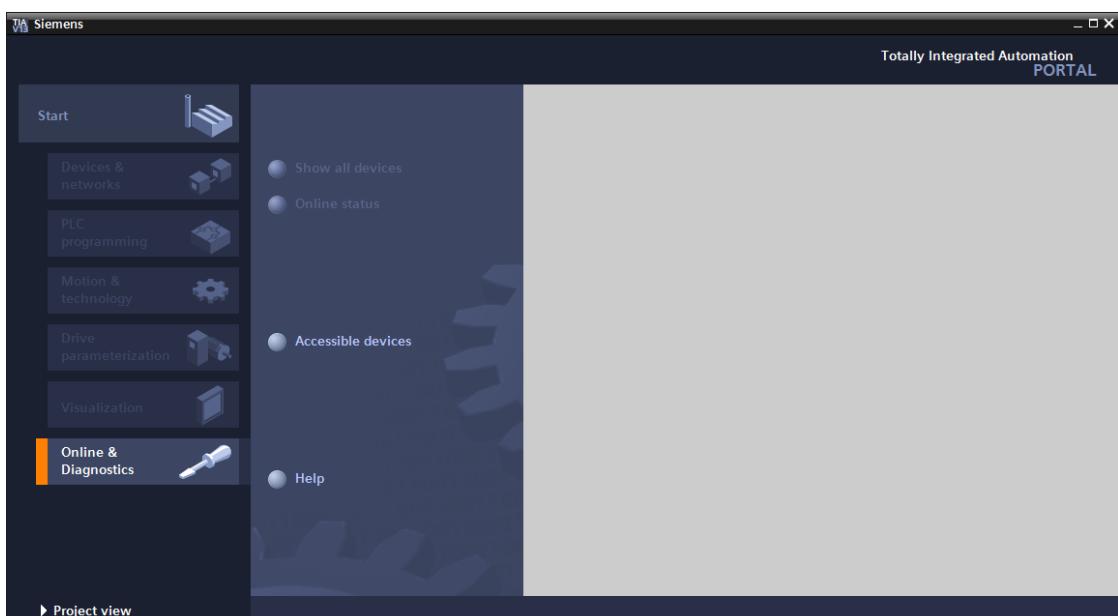
4.3.7 Resetting the SIMATIC HMI Panel TP700 Comfort and setting the IP address

The SIMATIC HMI Panel TP700 Comfort can be reset directly in the TIA Portal. A new IP address can also be assigned to the panel here.

To do so, select the Totally Integrated Automation Portal, which is opened here with a double-click.

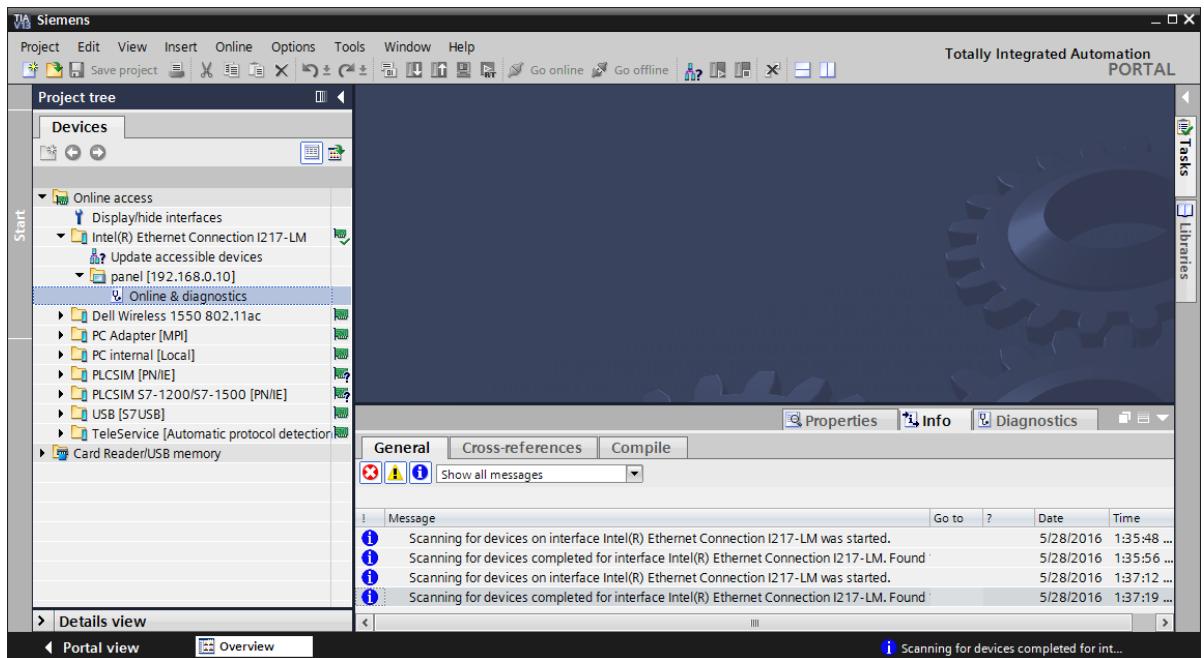
(→ TIA Portal V1X)

→ Then, select →"Online & Diagnostics" and open →"Project View".

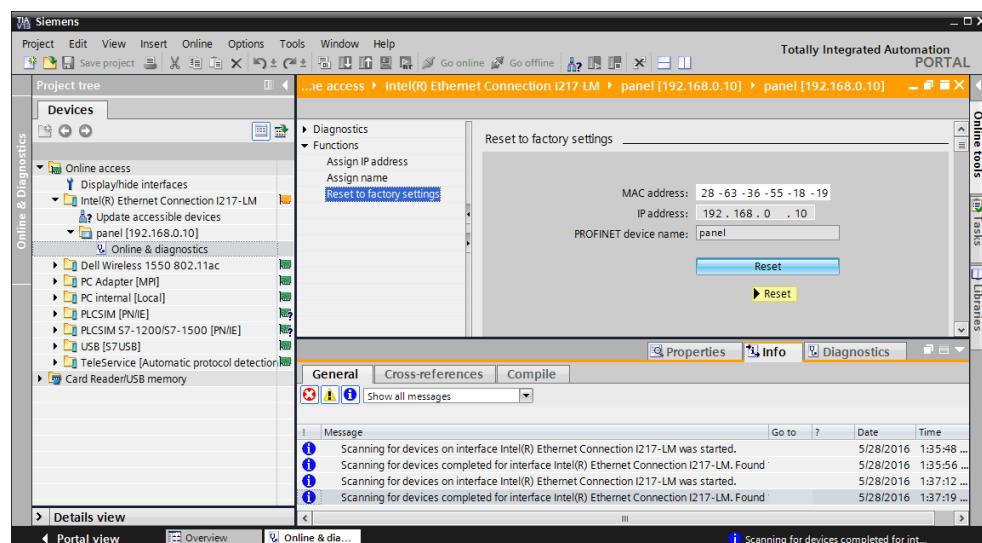


→ In the project tree, select the network adapter of your computer under → "Online access". If you click → "Update accessible devices" here, you will see the IP address (if

previously set) or the MAC address (if IP address not yet assigned) of the connected SIMATIC HMI Comfort Panel → Select → "Online & Diagnostics" here.

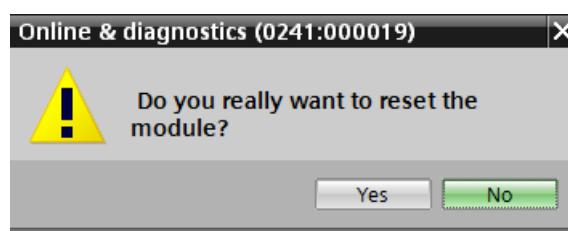


→ To reset the panel, select the → "Reset to factory settings" function and click → "Reset".

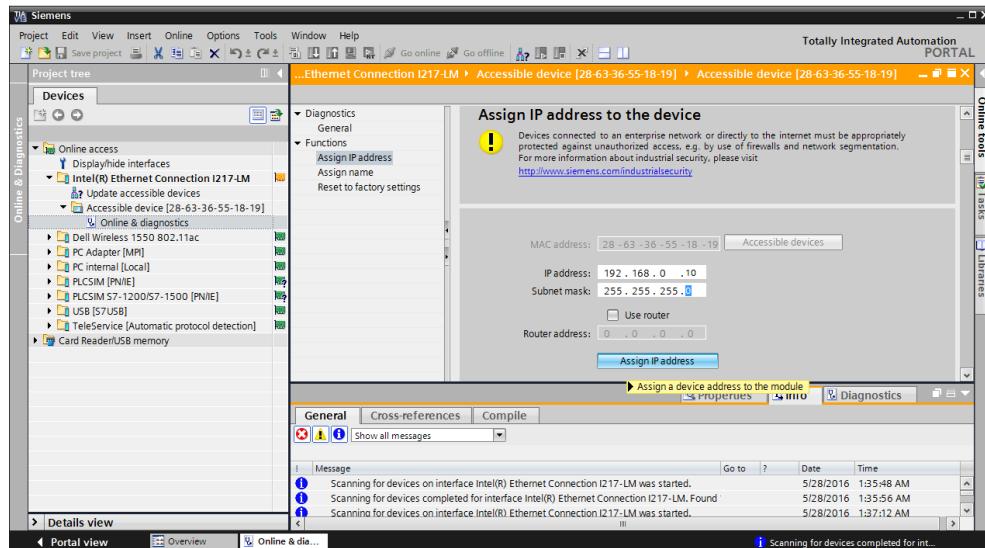


Note: For resetting to factory settings, "Runtime" should not yet be started in the panel and the "Start Center" should be displayed.

→ Confirm the prompt asking if you really want to reset the module with → "Yes".



- Before you can assign the IP address, you must wait until the panel has finished resetting. Afterwards, you must select → "Update accessible devices" and → "Online & diagnostics" of your panel again. To assign the IP address, select the → "Assign IP address" function here. Enter the following IP address here, for example: → IP address: 192.168.0.10 → Subnet mask 255.255.255.0. Click → "Assign IP address" and this new address will be assigned to your SIMATIC HMI Panel TP700 Comfort.

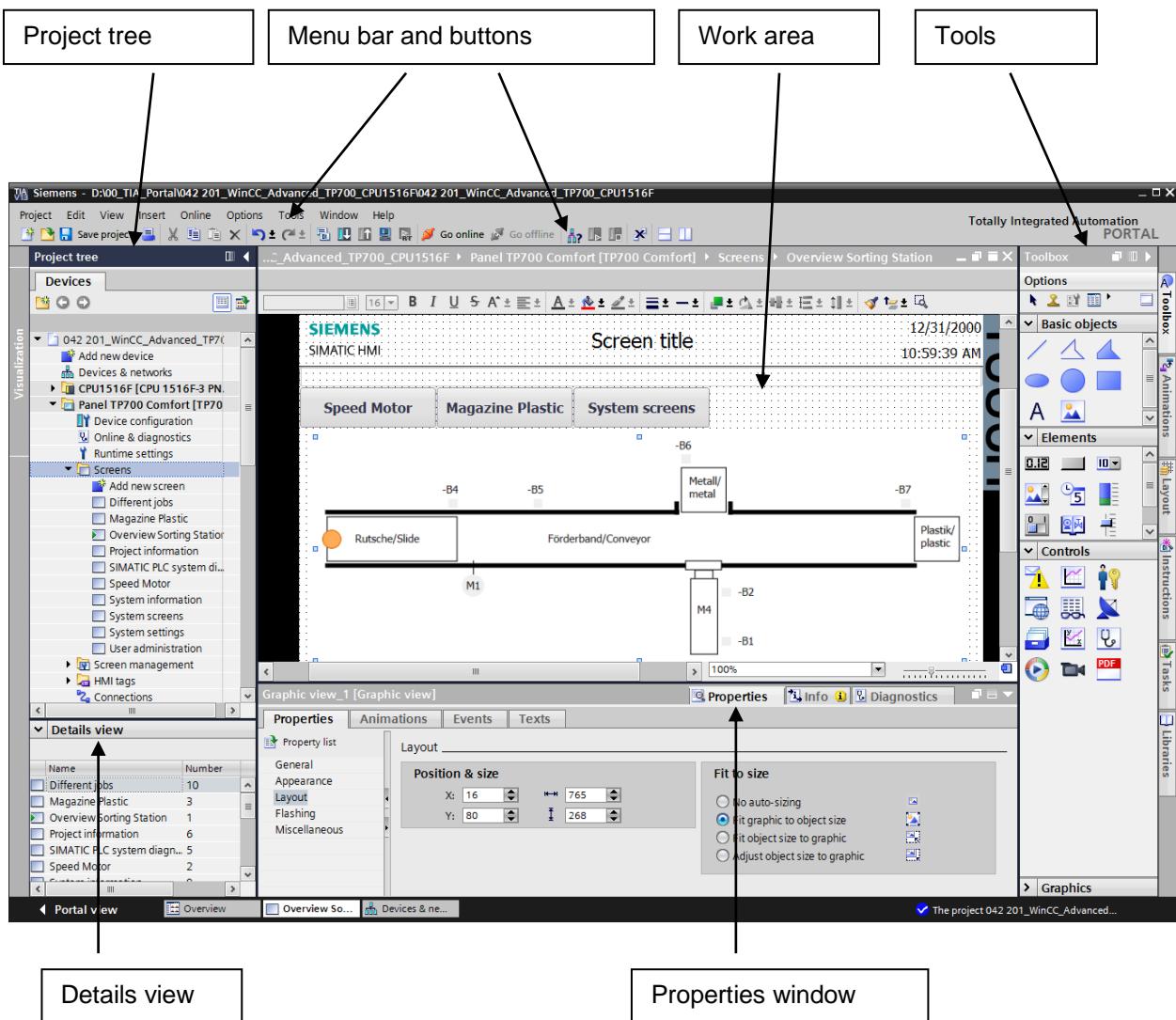


- If the IP address was not successfully assigned, you will receive a message in the → "Info" window under → "General".



Note: The IP address of the SIMATIC HMI Panel TP700 Comfort can also be set using the Windows CE operating system of the panel.

4.3.8 WinCC user interface

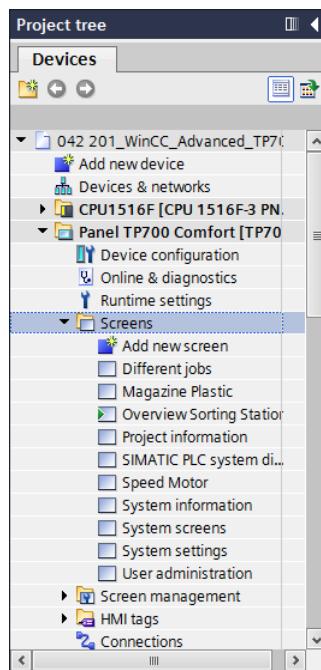


4.3.9 Project tree

The project tree window is the central control point for project editing. All components and all available editors of a project are displayed in a tree structure in the project window and can be opened there.

Each editor is assigned a symbol that enables you to identify the associated objects. Only elements supported by the selected HMI device are displayed in the project window.

You can access the HMI device settings in the project window.



4.3.10 Details view

The details view displays the contents or additional information on the objects selected in the project tree.

Details view	
Name	Number
Different jobs	10
Magazine Plastic	3
Overview Sorting Station	1
Project information	6
SIMATIC PLC system diag...	5
Speed Motor	2
System information	0

4.3.11 Menu bar and buttons

In the menus and toolbars, you will find frequently used functions you need to configure your HMI device. If a corresponding editor is active, editor-specific menu commands and toolbars are displayed.

If you point to a command with the mouse, a corresponding tooltip is provided for each function.

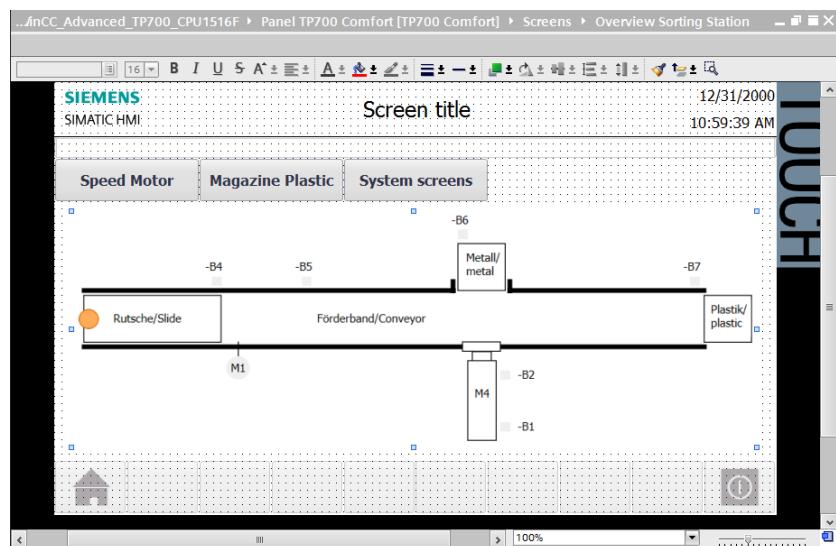


4.3.12 Work area

You edit the objects of the project in the work area. All other WinCC elements are arranged around the working area.

In the work area, you edit the project data either in tabular form (for example, tags), or graphically (for example, process screens).

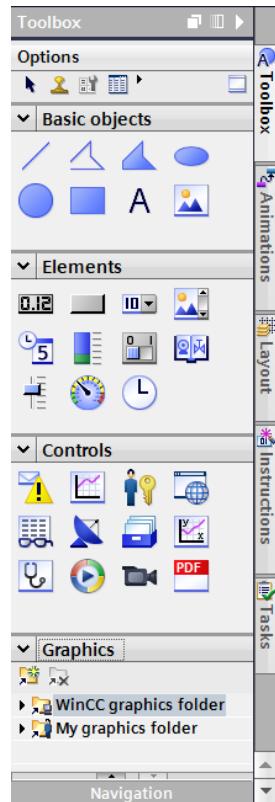
A toolbar is located in the upper part of the work area. Here, you can select, for example, the font, font color and functions such as rotate, align, etc.



4.3.13 Tools

The Tools window contains a selection of objects is provided that you can insert into your screens, such as graphic objects and operator controls. In addition, the Tools window contains graphics with preassembled graphic objects and collections of faceplates.

Objects are moved to the working area via "drag and drop".

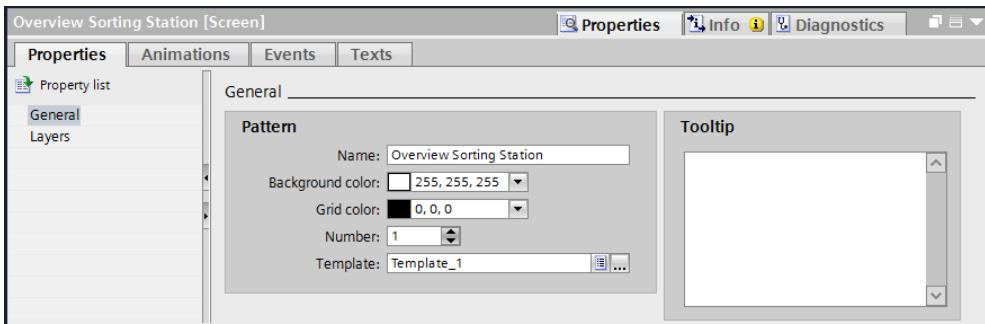


4.3.14 Properties window

In the properties window, you edit the properties of the objects selected in the work area (e.g. the color of screen objects). The window is available only in certain editors.

The properties of the selected object are displayed arranged by category in the properties window. As soon as you exit an input field, the value changes take effect. If you enter an invalid value, it is highlighted in color. The tool tip gives you information about the valid value range, for example.

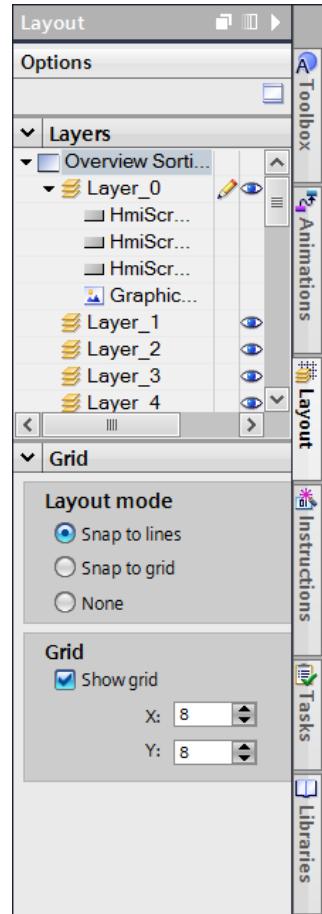
In addition, animations (e.g. change of color at a signal state change in the PLC) and events (e.g. change of screen when a button is released) are configured for a selected object in the properties window.



4.3.15 Other tabs

Settings for the work area such as the level selection and grid functions can be made in the layout window.

Animations, instructions, tasks and libraries of the select object can be selected using additional tabs.



5 Task

In this chapter, process visualization will be added to the program from chapter Data_Blocks. This will enable you to better monitor the process flow and operate the process more effectively.

6 Planning Process visualization

A Touch Panel TP700 Comfort is to be used for the process visualization.

The programming device, a SIMATIC S7-1500 controller and the Touch Panel TP700 Comfort are interconnected via the **Ethernet interface**.

The basic configuration is to be performed using the wizard in the TIA Portal V1X. All **system screens** are to be created in the process.

In an overview screen "**Overview of sorting station**", the process will be represented by the conveyor and sensors. Conveyor speed and the count of plastic workpieces will also be displayed here.

In this screen, it is also to be possible to select the operating mode, start and stop in automatic mode and reset the counter.

The actual speed of the motor will be graphically displayed in another screen titled "**Motor Speed**". The setpoint speed can also be specified here.

The "**Plastic Magazine**" will be created in a first step.

An alarm line, date/time and the plant states "Emergency Stop ok/triggered", "Main switch ON/OFF" and "Automatic started/stopped" are to be displayed in the **header** of all screens.

The **footer** contains a button that can be used to return to the start screen and a button to exit Runtime.

The **alarm system** is also to be configured.

System events are to be displayed on the panel and the CPU and limit violations of the motor speed and the main switch monitored.

The alarms will be displayed in the alarm line of the header and shown automatically in alarm windows when errors/warnings occur.

Remote control of the Touch Panel TP700 Comfort is activated via the **web server**.

6.1 Program description for the sorting station with speed control and speed monitoring of the motor

The "**MOTOR_AUTO**" [FB1] function block controls a **conveyor in automatic mode**.

The **Memory_automatic_start_stop** is latched with the **Start_command** but only if the **reset** conditions are not present.

The **Memory_automatic_start_stop** is to be reset if a **Stop_command** is pending or safety shutoff is active or automatic mode is not activated from the visualization.

The **Conveyor_motor_automatic_mode** output is activated when **Memory_automatic_start_stop** is set, the enable conditions are met and **Memory_conveyor_start_stop** is set.

To save energy, the conveyor should only run when a part is present. For this reason, the **Memory_conveyor_start_stop** is set when **Sensor_chute_occupied** signals a part and reset when **Sensor_end_of_conveyor** produces a negative edge or safety shutoff is active or automatic mode is not activated (manual mode).

Because the Sensor_end_of_conveyor is not able to be mounted directly at the end of the conveyor, signal stretching of the Sensor_end_of_conveyor signal is programmed.

The magazine for plastic holds only five parts. For this reason, the parts will be counted at the end of the conveyor. When the magazine contains five parts, automatic mode is to be stopped. After the magazine has been emptied, automatic mode will be restarted with the Start_command once the counter has been reset from the visualization.

The **speed setting** is made at an input of the "SPEED_MOTORCONTROL" [FC10] function in revolutions per minute (range: +/- 50 rpm).

First, the function first checks for correct entry of the speed setpoint in the range +/- 50 rpm.

If the speed setpoint is outside the range +/- 50 rpm, the value 0 will be output at the speed manipulated value output. The return value of the function (Ret_Val) will be assigned the value TRUE (1).

If the speed setting is within the range +/- 50 rpm, this value will first be normalized to the range 0...1 and then scaled to +/- 27648 with data type 16-bit integer (Int) for output as the speed manipulated value at the analog output.

In the "SPEED_MOTORMONITORING" [FC11] function, the actual value will be made available to B8 as an analog value and queried at an input of the "SPEED_MOTORMONITORING" [FC11] function.

The actual speed value will be scaled to revolutions per minute (range: +/- 50 rpm) and made available at an output.

The following four limits can be specified for the block inputs in order to monitor them in the function:

Speed > SPEED_MOTOR_monitoring_error_max

Speed > SPEED_MOTOR_monitoring_warning_max

Speed < SPEED_MOTOR_monitoring_warning_min

Speed < SPEED_MOTOR_monitoring_error_min

If a high or low limit is violated, the value TRUE (1) is assigned to the corresponding output bit.

If a fault is present, the protective tripping of the "MOTOR_AUTO" [FB1] function block is to be tripped.

The speed setpoint and the actual speed value as well as the positive and negative error and warning limits are created in the data block "SPEED_MOTOR" [DB2], as are the error and warning bits.

The setpoint and actual value of the counter for the plastic parts will be specified and displayed in the global data block "MAGAZINE_PLASTIC" [DB3]. These values are interconnected with the "MOTOR_AUTO" [FB1] function block using an input for the setpoint setting and an output for the actual value display.

6.2 Technology diagram

Here, you see the technology diagram of the plant for the task.

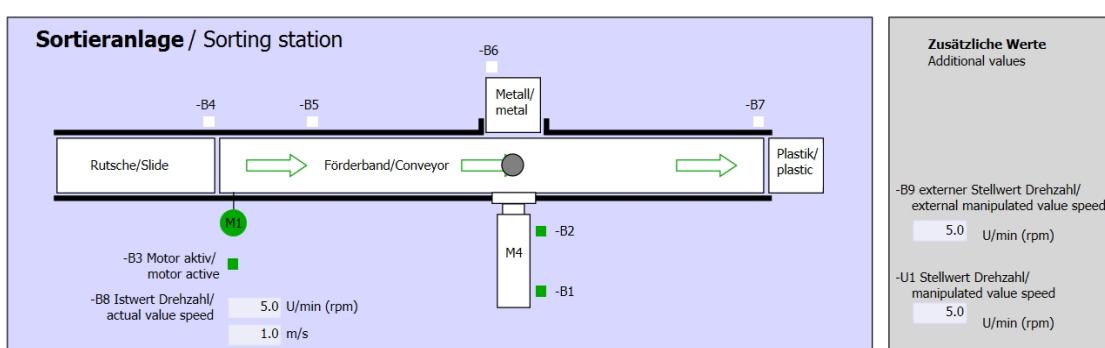


Figure 4: Technology diagram



Figure 5: Control panel

6.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton "Automatic Start"	NO
I 0.4	BOOL	-S2	Pushbutton "Automatic Stop"	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
IW64	BOOL	-B8	Sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm	

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in two directions +/-10V corresponds to +/-50 rpm	

Legend for reference list

DI	Digital input	DO	Digital output
AI	Analog input	AO	Analog output
I	Input	O	Output
NC	Normally Closed		
NO	NO Normally Open		

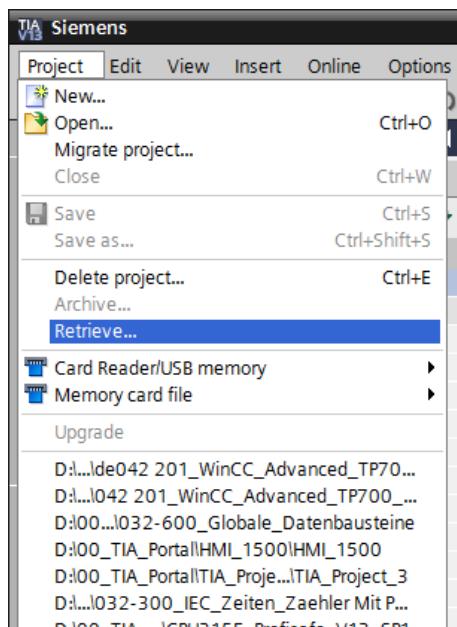
7 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, follow the individual instructions in the steps below.

7.1 Retrieving an existing project

- Before you can expand the "SCE_EN_032-600_Global_Data_Blocks_R1508.zap13" project from chapter "SCE_EN_032-600_Global_Data_Blocks", you must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with "Open".

(→ Project → Retrieve → Select a .zap archive ... → Open)

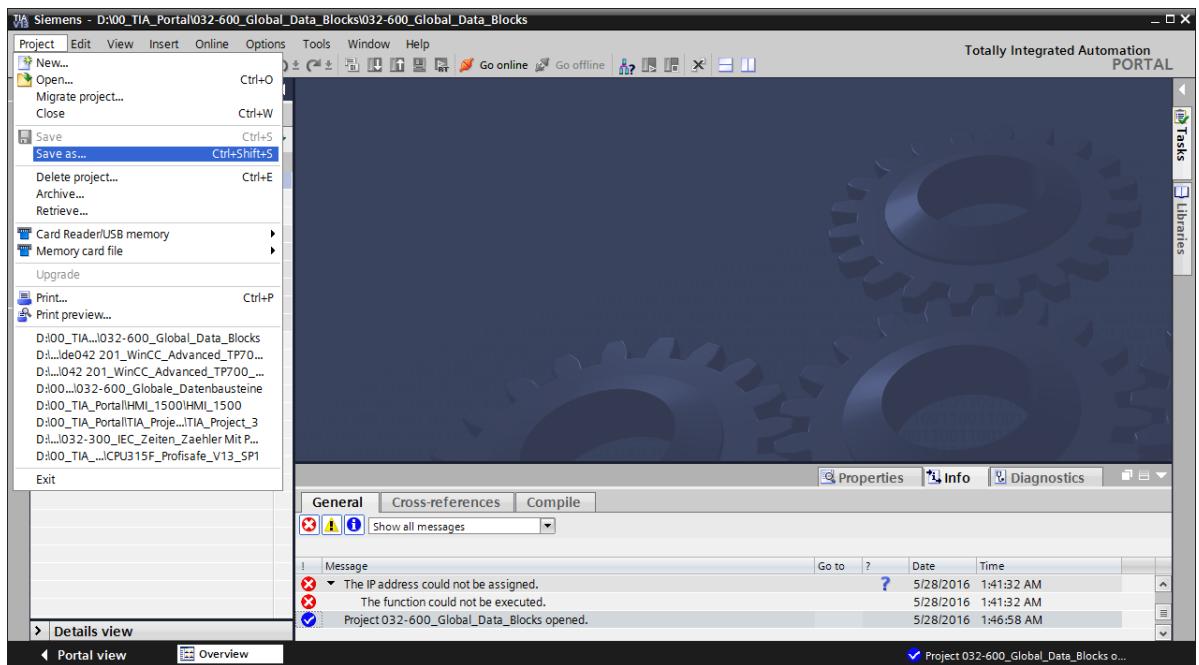


- The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK".

(→ Target directory ... → OK)

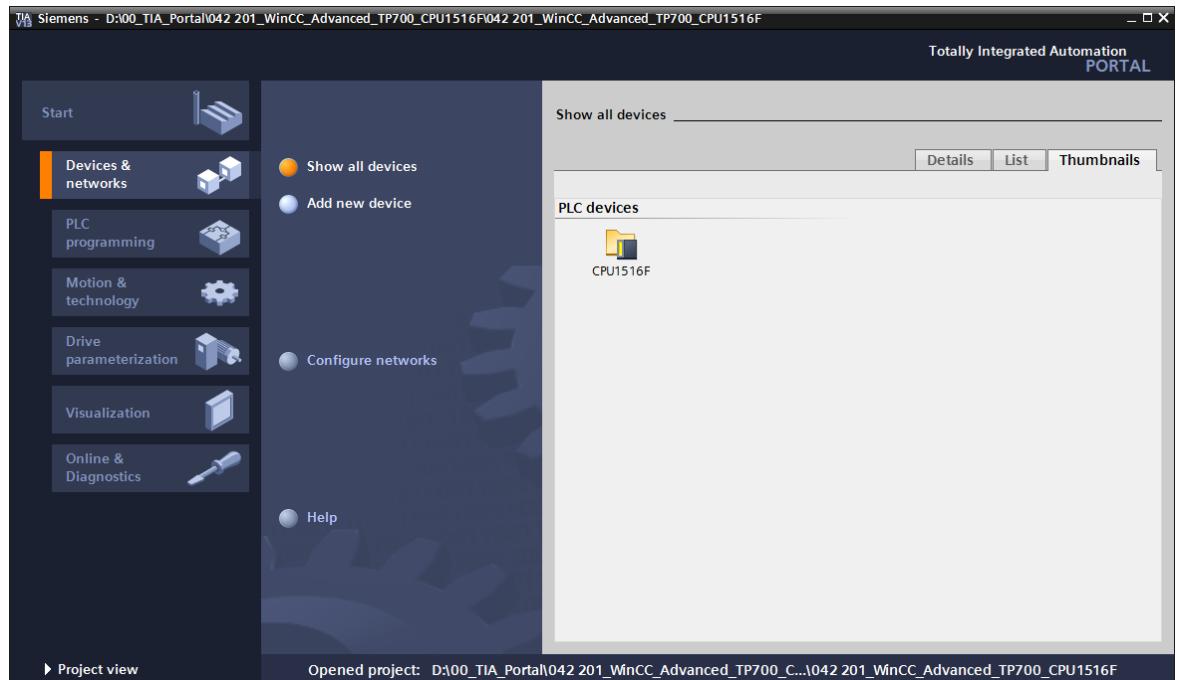
- Save the opened project under the name 042-201_WinCC_Advanced_TP700_CPU 1516F.

(→ Project → Save as ... → 042-201_WinCC_Advanced_TP700_CPU 1516F → Save)

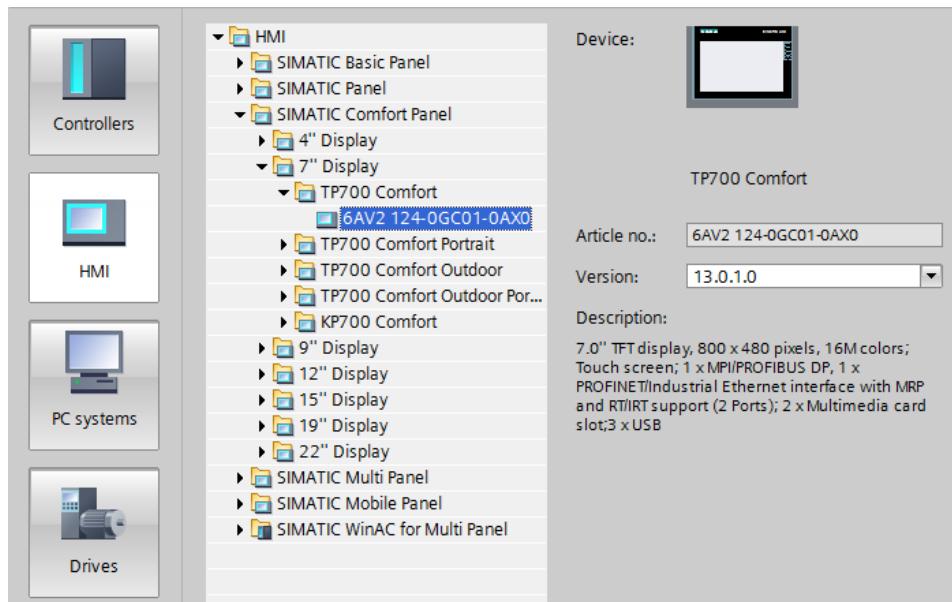


7.2 Adding a SIMATIC HMI Panel TP700 Comfort

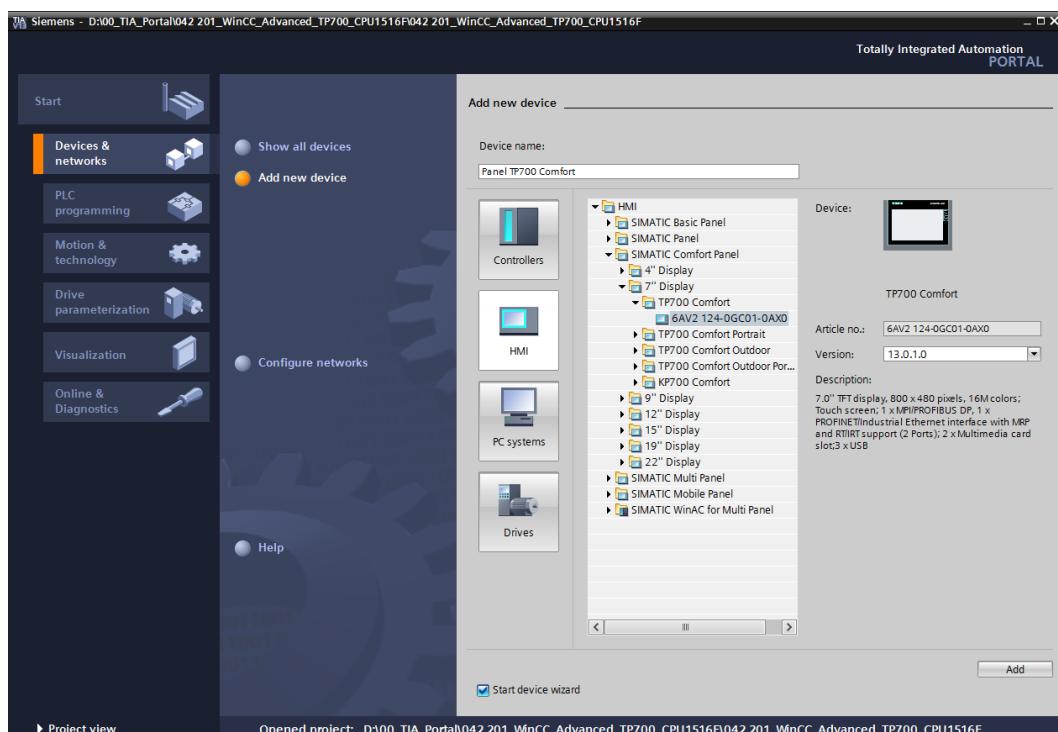
- To create a new panel in the project, go to the portal view. In the portal, select the menu command → "Devices & networks" and → "Add new device".



- Now, select the device version as follows: → "HMI" → "SIMATIC Comfort Panel" → "7" Display" → "TP700 Comfort" and the correct order number of your panel, here, for example, → 6AV2 124-0GC01-0AX0



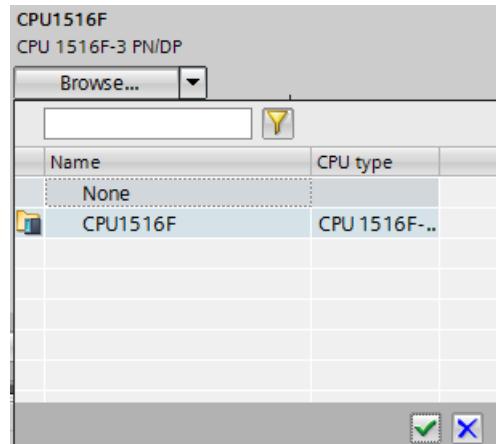
- Enter "Panel TP700 Comfort" as the device name and → select the check box "Start device wizard". Click the **Add** button here.



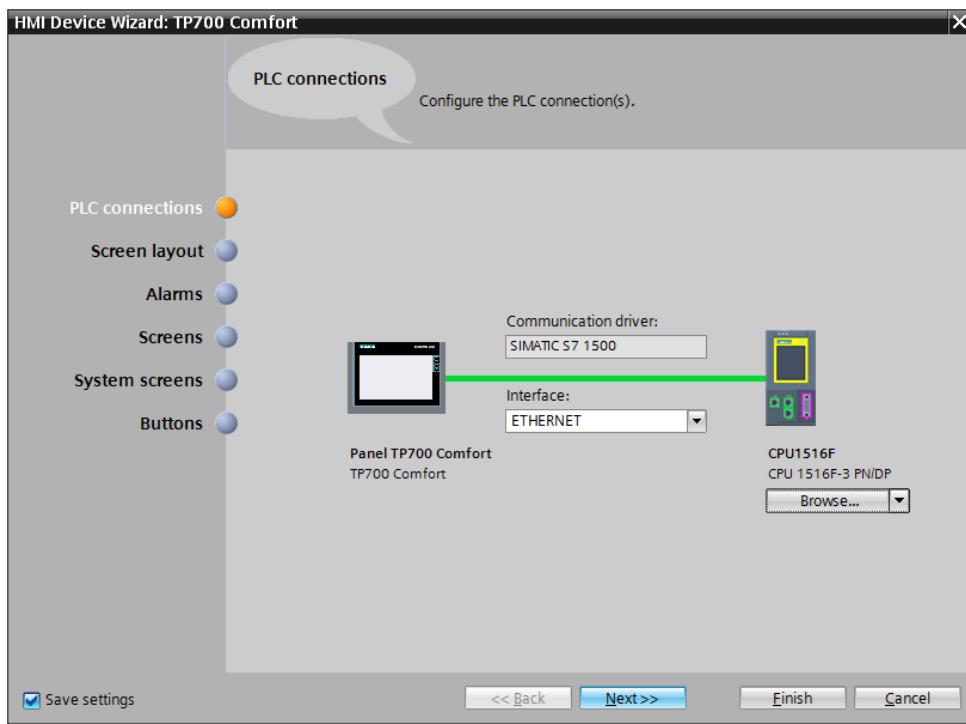
7.3 HMI wizard for the TP700 Comfort Panel

The TIA Portal now creates the desired panel and automatically starts the HMI wizard for the TP700 Comfort Panel. This helps in specifying a few basic settings for the panel.

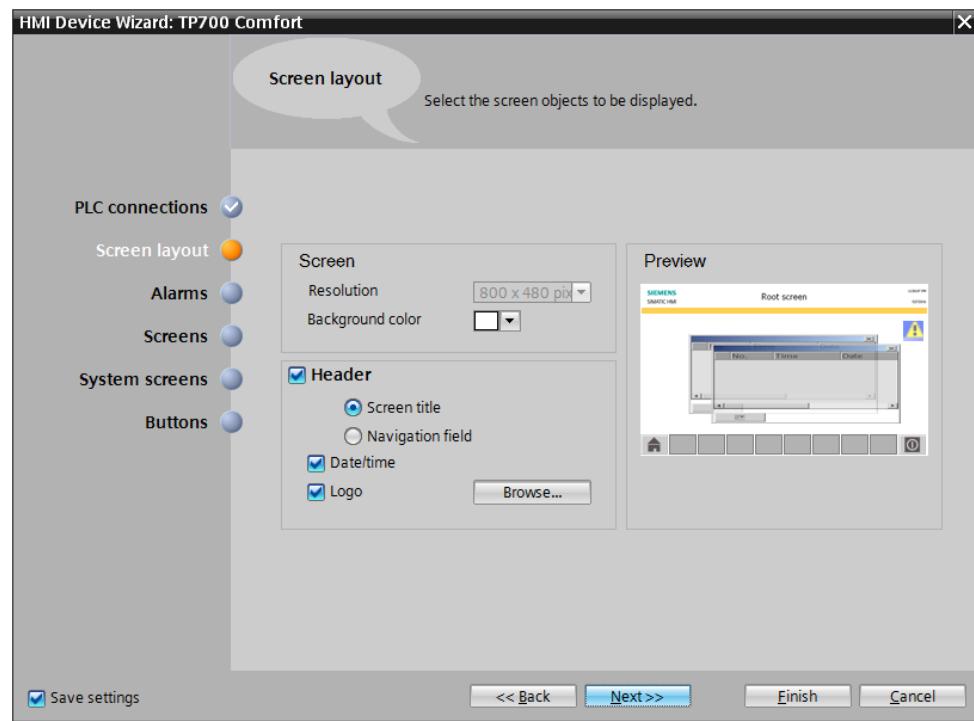
- A prompt for the PLC connections appears first. Choose your previously configured CPU 1516F as the communication partner.



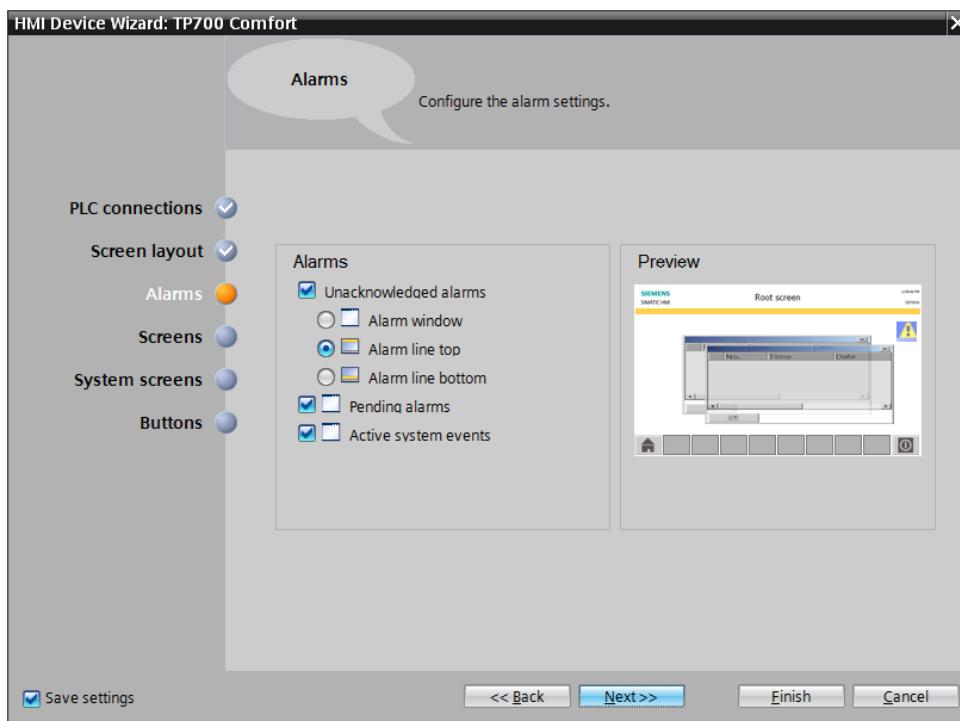
- In order to connect your panel to the CPU, select the "Ethernet" interface. → Confirm your selection by clicking "[Next >>](#)"



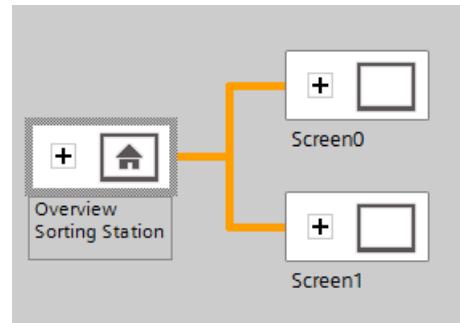
- Under "Screen layout" you can change the default background color of your panel. → Select the "Page header" check box with screen title "Date/time" and "Logo". → Confirm your selection by clicking "[Next >>](#)".



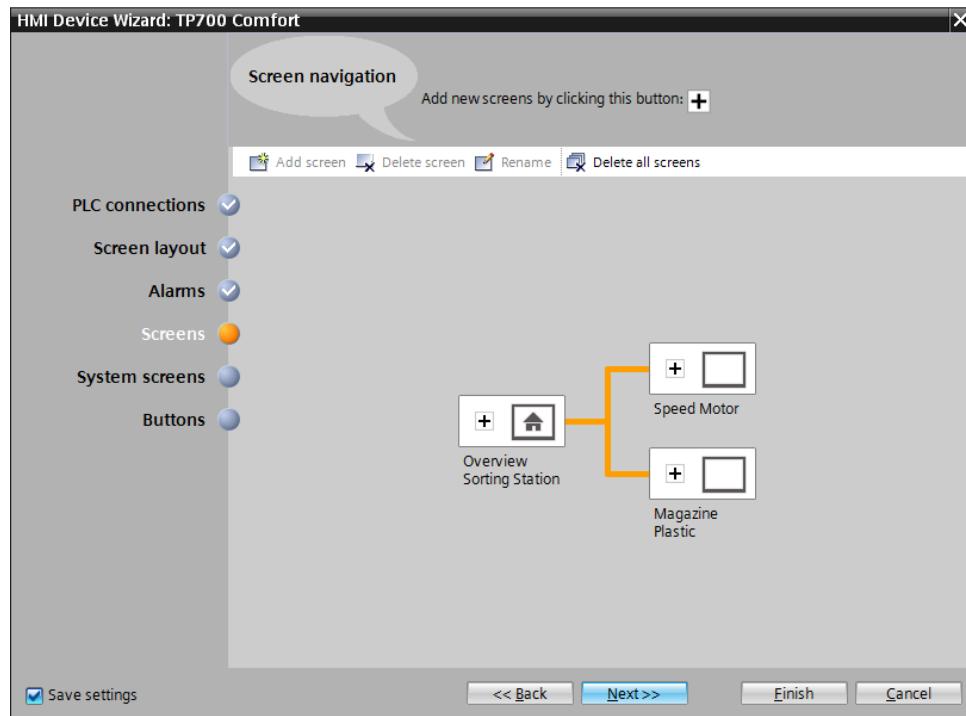
- In the "Alarms" section, you can specify the alarms that are to be displayed in a window. Select all 3 alarm types (with the option "Top alarm line" for unacknowledged alarms) → Confirm your selection by clicking "[Next >>](#)".



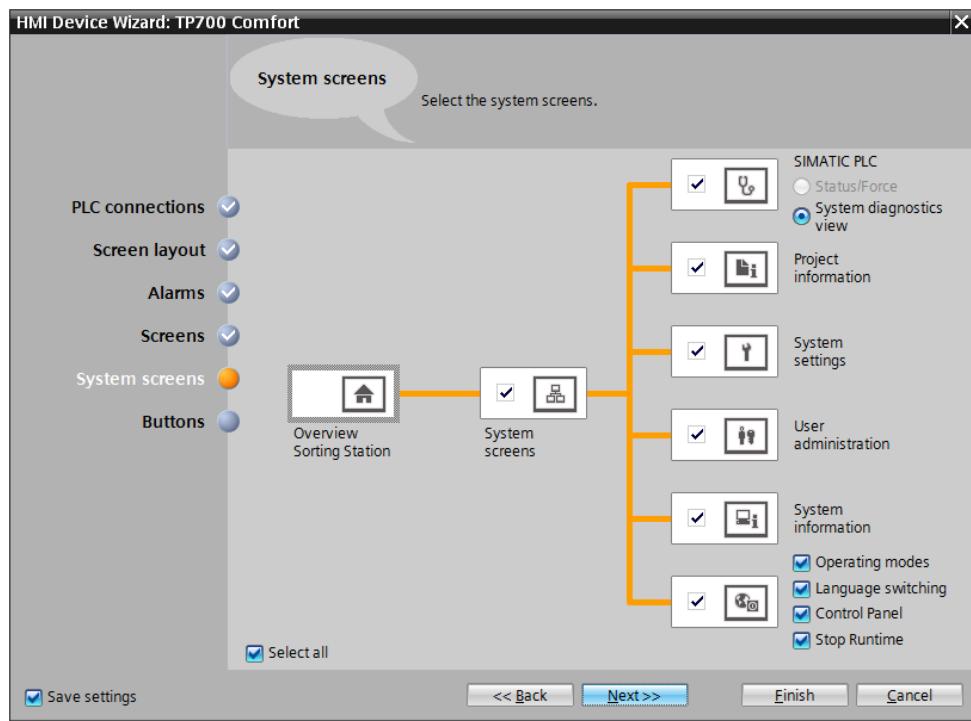
- The "Screen navigation" section shows the screen structure with the screen names of the last created project, in which the start screen is the first screen shown on the left.
- By clicking a screen name, you can easily assign a new name → By clicking **+**, you can insert new pictures in the hierarchy → By clicking "**Delete screen**", you can delete selected screens.



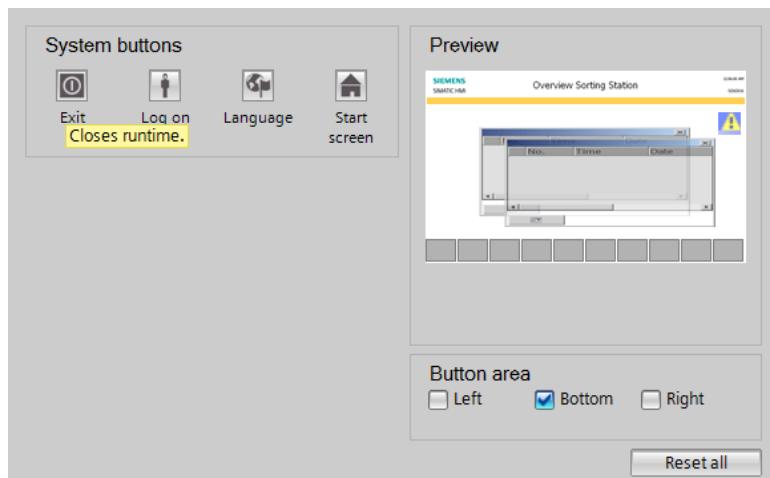
- Use this method to create the screen structure shown below with the corresponding screen names. → Confirm your selection by clicking "**Next >>**".



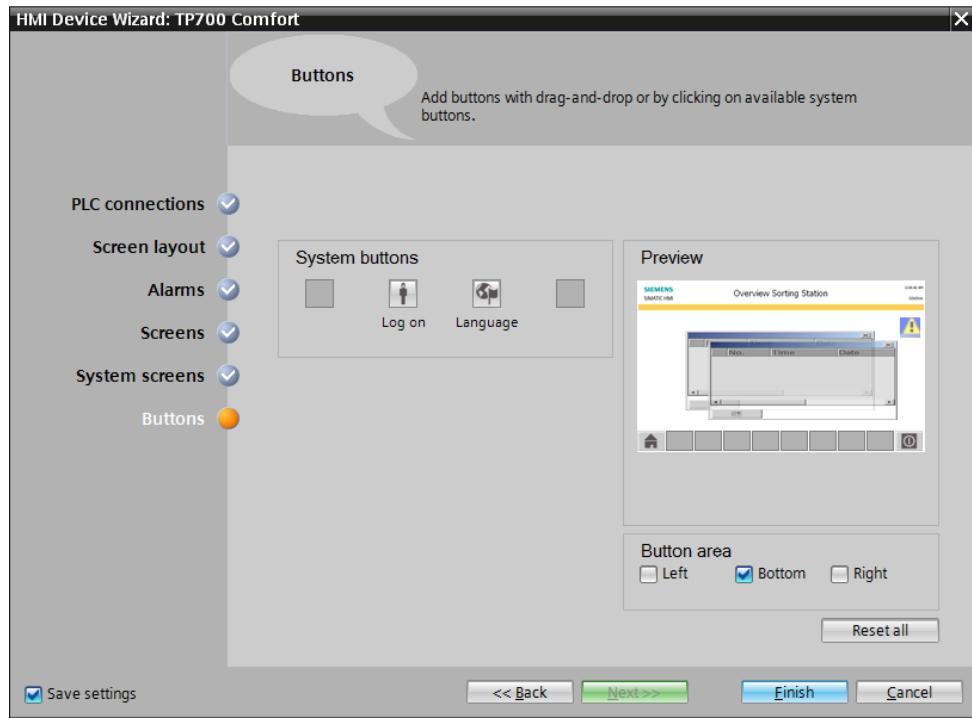
- In the "System screens" section, you can select previously preset views for system functions and automatically insert them. → Enable all system screens with "Select all"
- Confirm your selection by clicking "**Next >>**".



- In the "System buttons" section, you will find the four freely selectable buttons for Exit , Log on , Language , and Start screen . You can place these buttons on the provided button areas "Left", "Bottom" or "Right" using drag-and-drop.

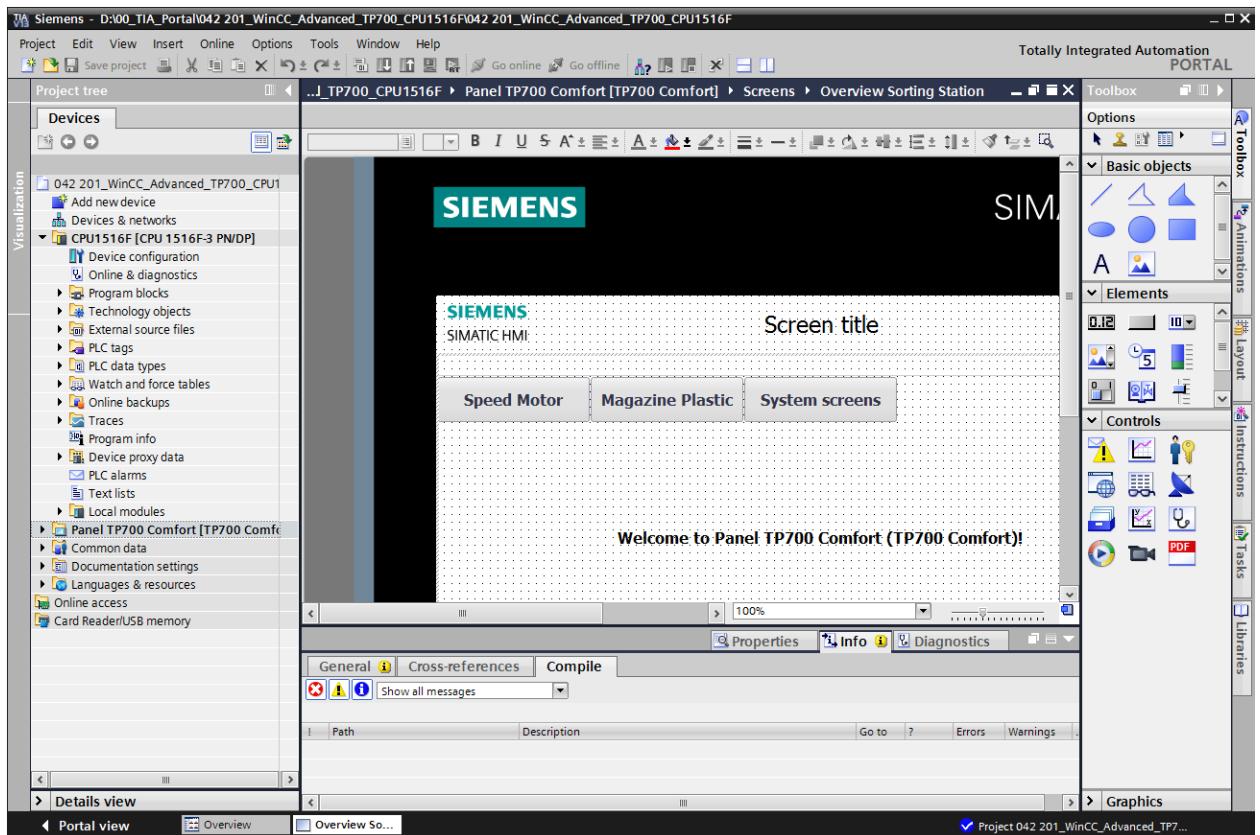


- Enable only the "Button area" "Bottom". → Insert the button for the "Start screen" , on the left and the button for Runtime "Exit" , on the right. → Confirm your selection by clicking ""

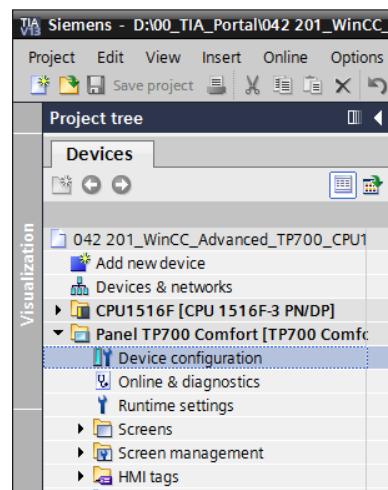


7.4 Device configuration of the TP700 Comfort Panel

- The TIA Portal now switches automatically to the project view and displays the start screen of our visualization there.

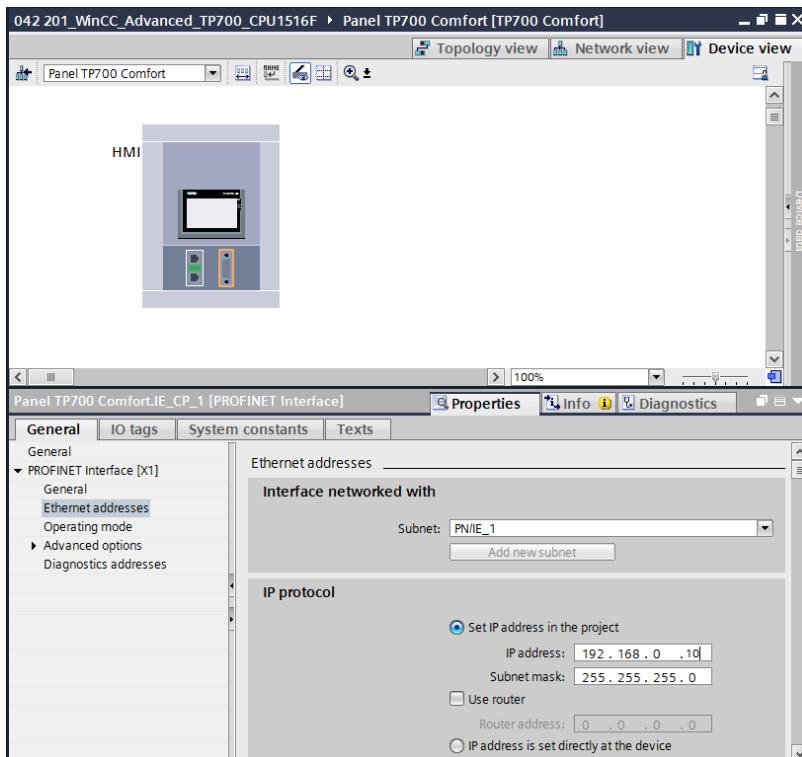


- In order to configure our panel, select "Panel TP700 Comfort" in the project tree and double-click it to open its "Device configuration".



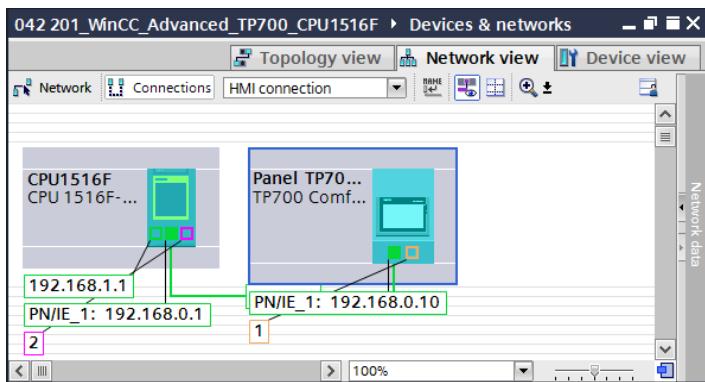
7.4.1 Setting the IP address

- Double-click the Ethernet interface of the panel in the Device view.
- In the → "Properties" under General, open the → "PROFINET interface [X1]" menu command and select the → "Ethernet addresses" entry there.
- Set the IP address 192.168.0.10 under IP protocol.



Note: The subnet mask was previously set in the settings of the CPU 1516F and is applied automatically to the panel.

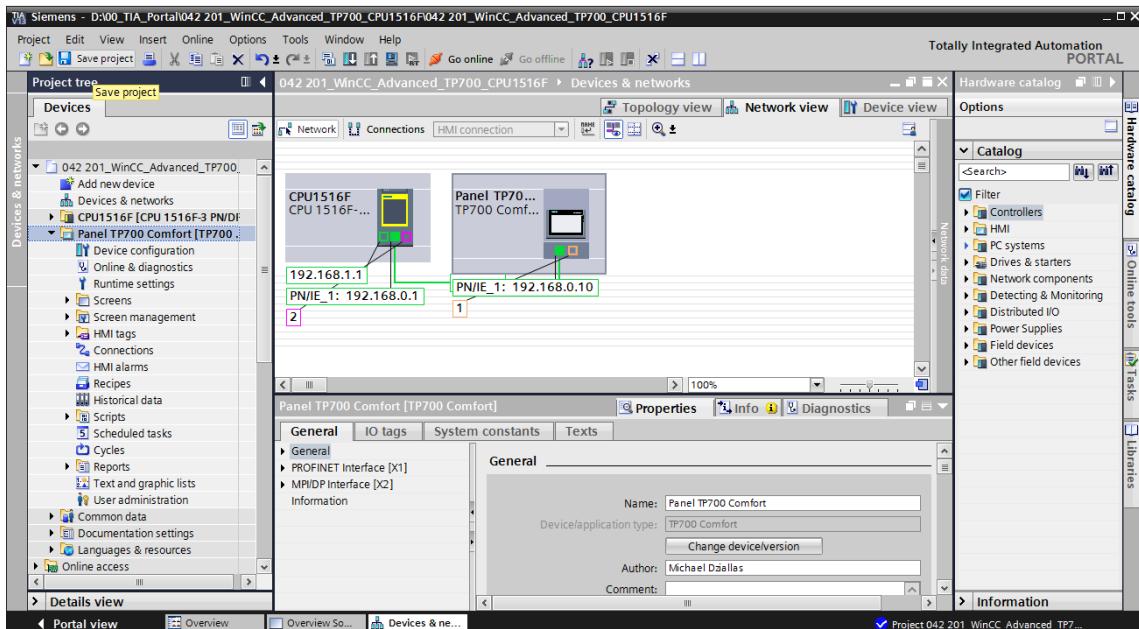
- To obtain an overview of the assigned addresses within a project, you can click the → " icon in the → "Network view". If you click → Connections, the "HMI connection" between the CPU and panel that you previously created in the wizard will be displayed.



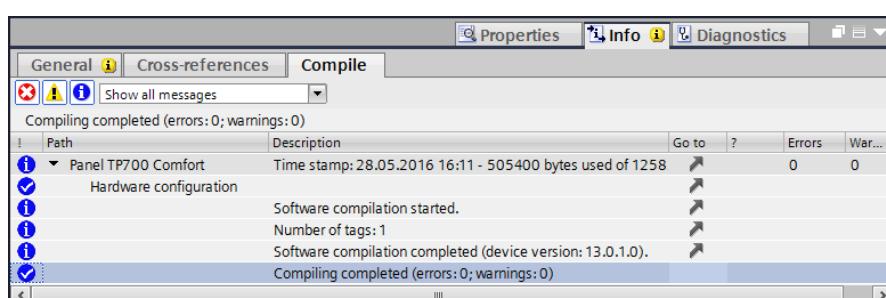
7.5 Compiling the CPU and panel and saving the project

- To compile all blocks, click the "CPU_1516F" folder and select the icon for compiling in the menu. To compile the panel, click the "Panel TP700 Comfort" folder and select the icon for compiling in the menu. You can save your project by clicking the button in the menu.

(→ CPU_1516F → → Panel TP700 Comfort → →

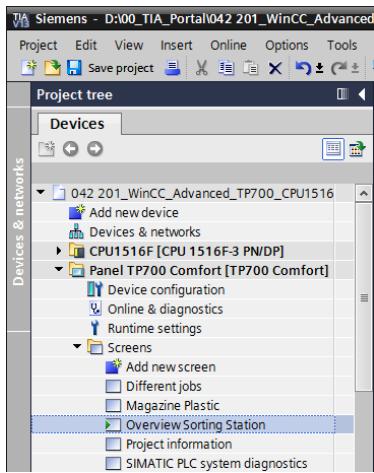


- The 'Info' area then indicates whether compiling was successful or whether warnings or errors occurred.

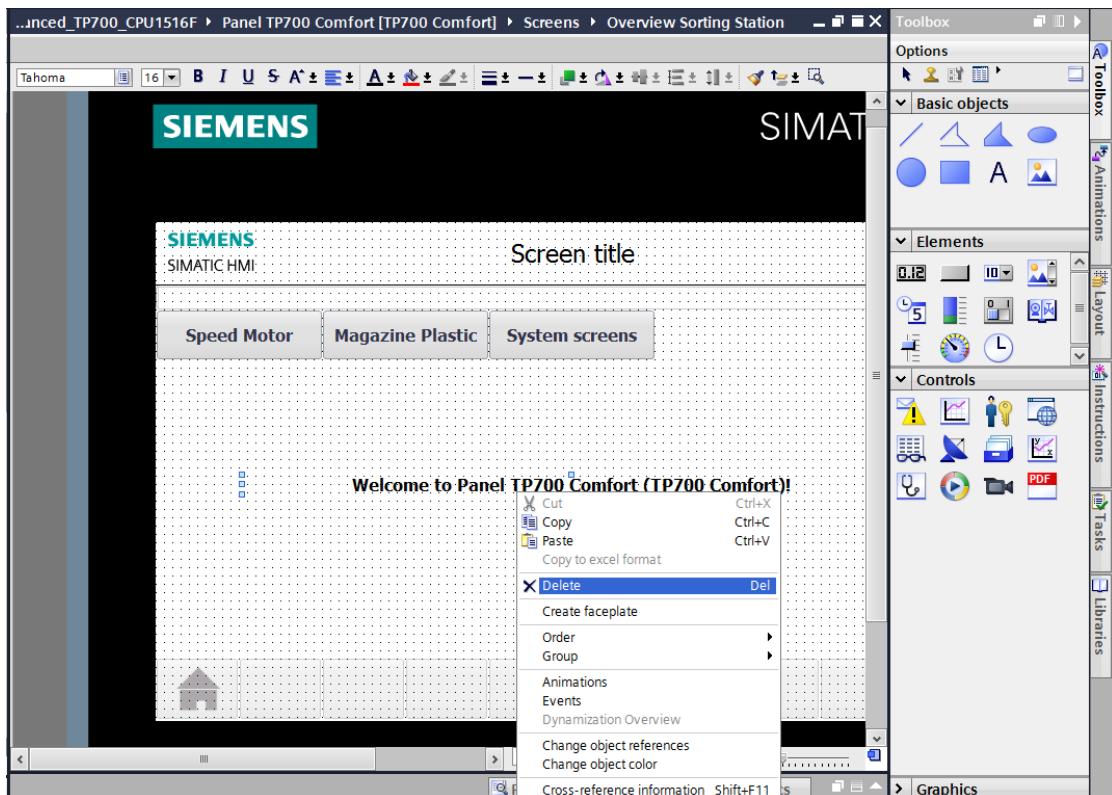


7.6 Configuring the graphic display

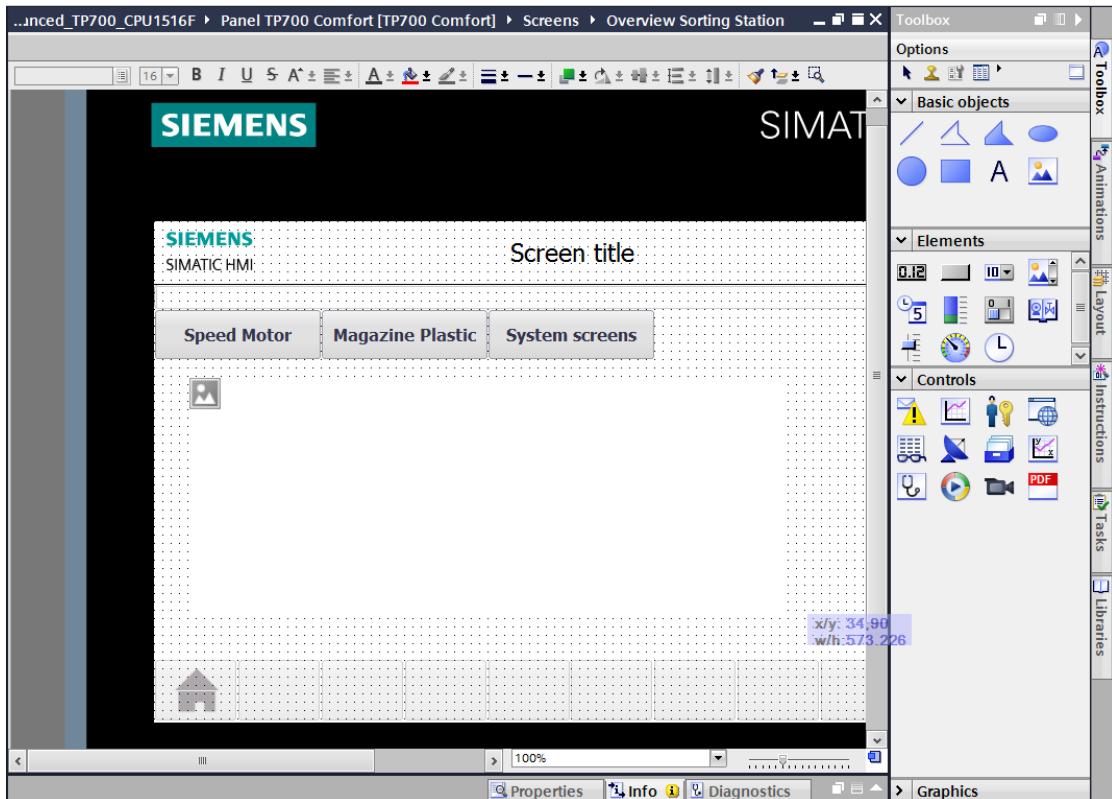
- After successful compilation, you want to lay out the first screen for the visualization. To do this, first open the → "Overview of sorting system" screen by double-clicking it.



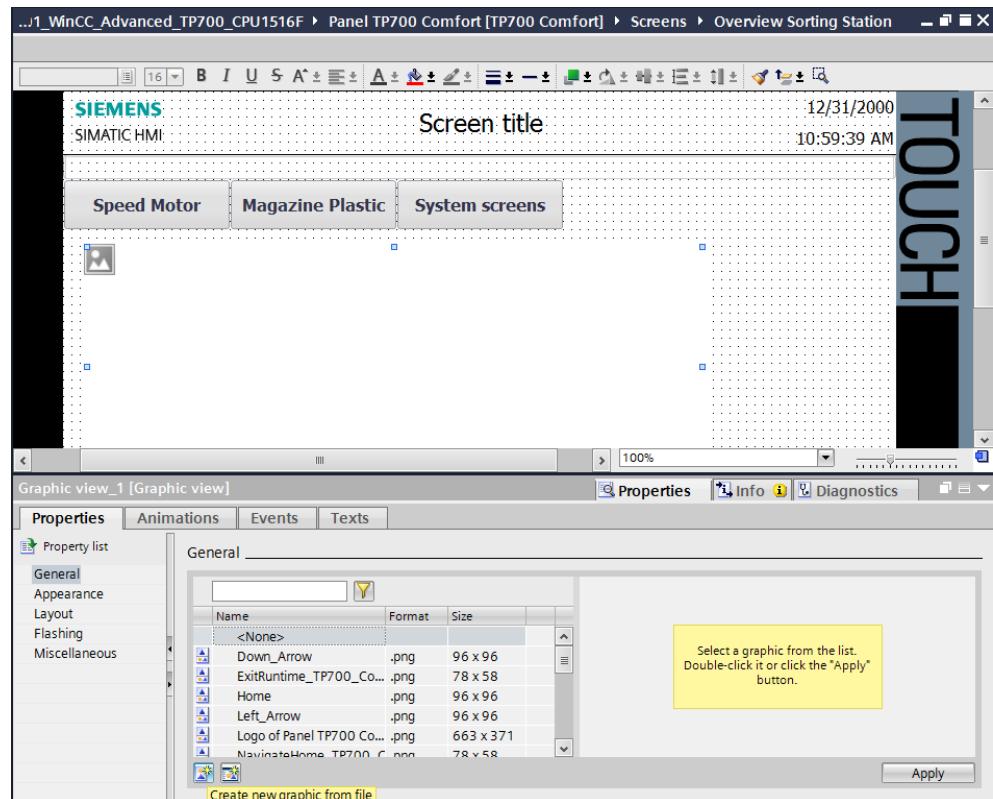
- Many objects such as the screen change buttons were already created by the wizard. The text filed in the center of the screen is to be removed by right-clicking it and selecting → "Delete" in the displayed dialog.



- From Tools, select → "Graphic display"  under → "Basic objects". The mouse pointer changes so that you can now draw an area for displaying a graphic in the work area.



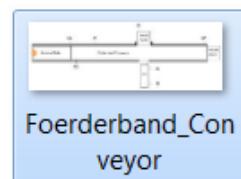
- You can double-click the graphic display area to have its properties displayed. Select the
→ "Create graphic from file" icon →  in the → "General" sub-item.



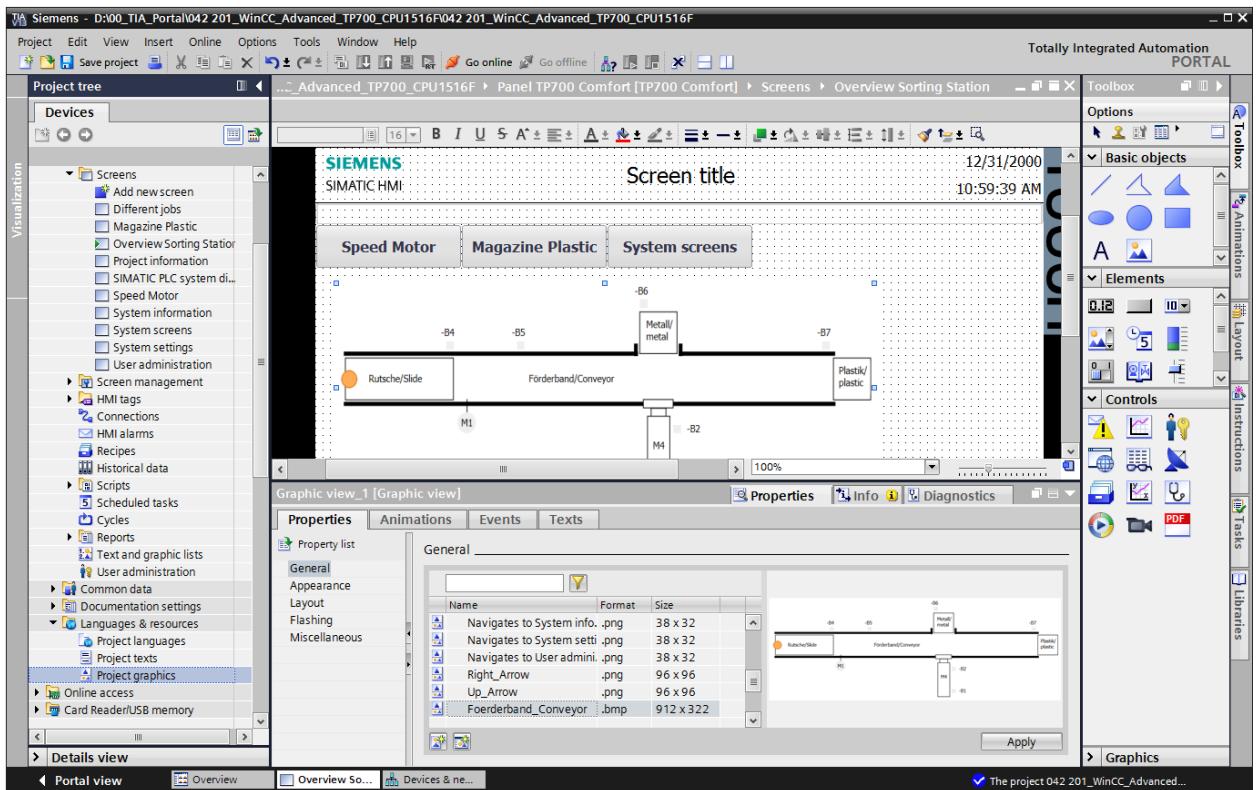
Note: There are four sub-items for the properties of the objects.

- Properties for static settings of the object.
- Animation for dynamic settings of the object.
- Events if actions are to be triggered from objects.
- Texts for the multilingual display within an object.

→ In the displayed dialog, select the Picture → "Open"

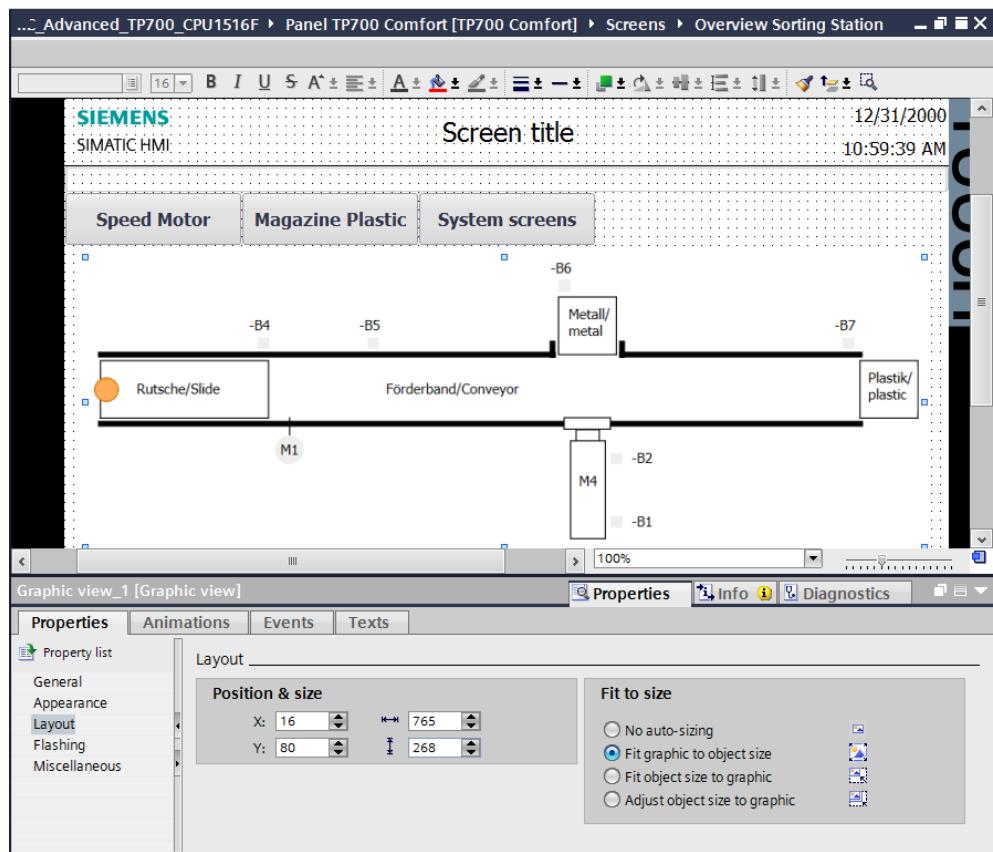


→ Select the "Foerderband_Conveyor.bmp" graphic for the display and click → "Apply".



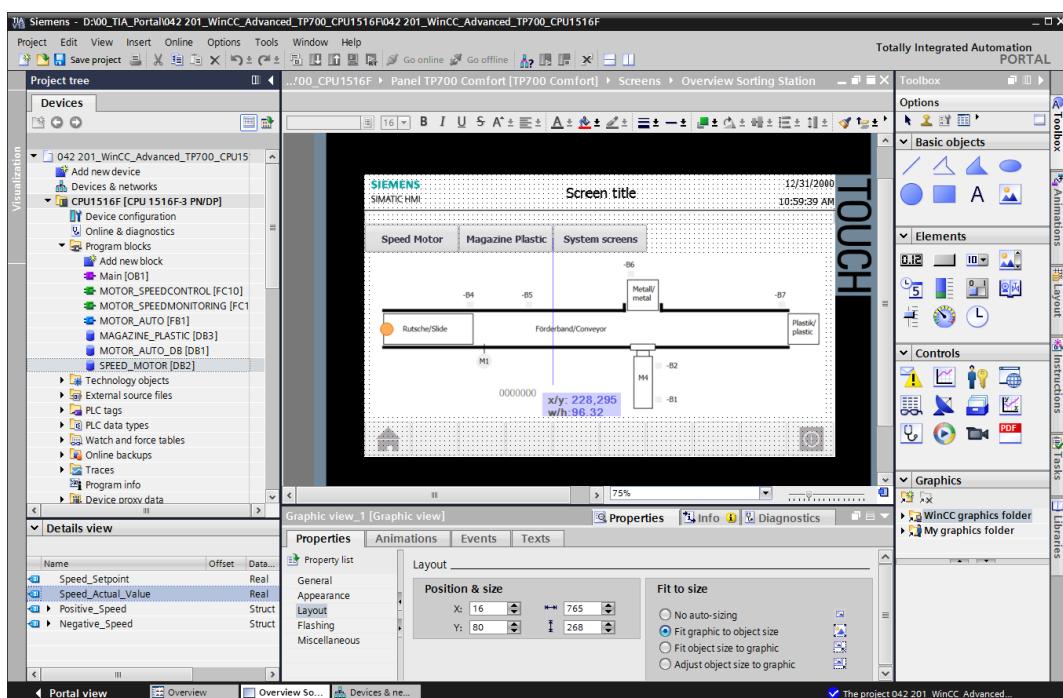
Note: The generated graphic is stored in the "Languages & Resources" path under "Project graphics".

→ Position the graphic with the mouse in such a way that the positions and sizes shown below are entered → under "Layout" → in the properties. Select the → "Fit graphic to object size" option to adapt the size.

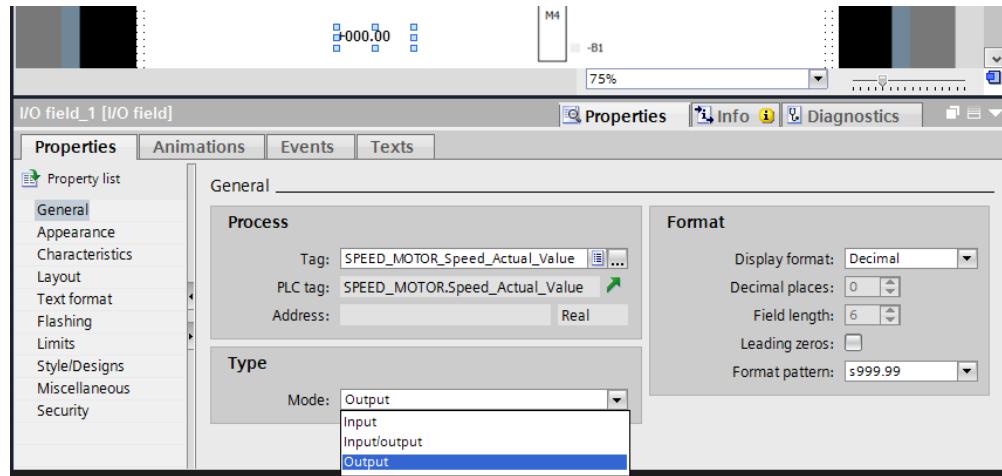


7.7 Displaying a process value in an IO field

- You first want to insert a display of the actual value of the current speed below the conveyor motor. To do this, select the → "Program blocks" of the → "CPU_1516F" and the → "SPEED_MOTOR[DB2]" data block there. Then drag the → "Actual speed value" tag from the → "Detail view" to our "Overview of sorting station" screen.

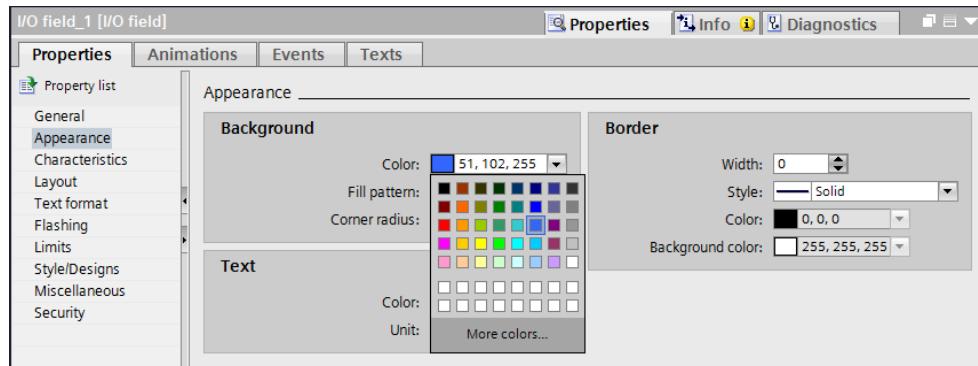


- The connection to the tag in the PLC has already been created in the "Properties" of the IO field under "General", "Process". The "Display format" is set to "Decimal". Now, all you have to do is change the "Format pattern" to → "s999.99" and the "Type" of the field to → "Output".

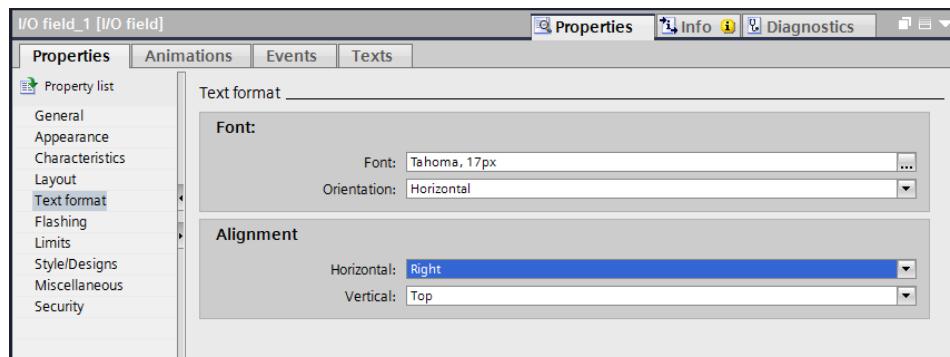


Note: The format pattern s999.99 means that the display will include three places in front of the decimal point two places after the decimal point and a sign.

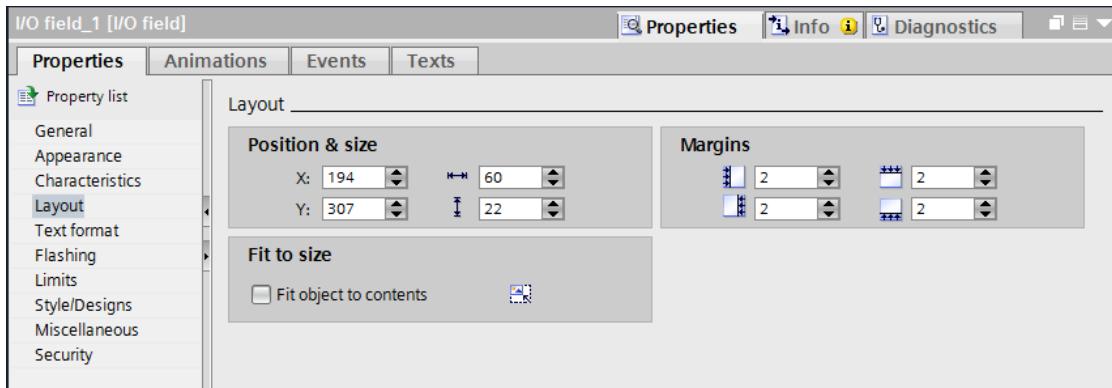
- In the "Properties" under "Appearance", we change the "Color" of the "Background" to → "Blue".



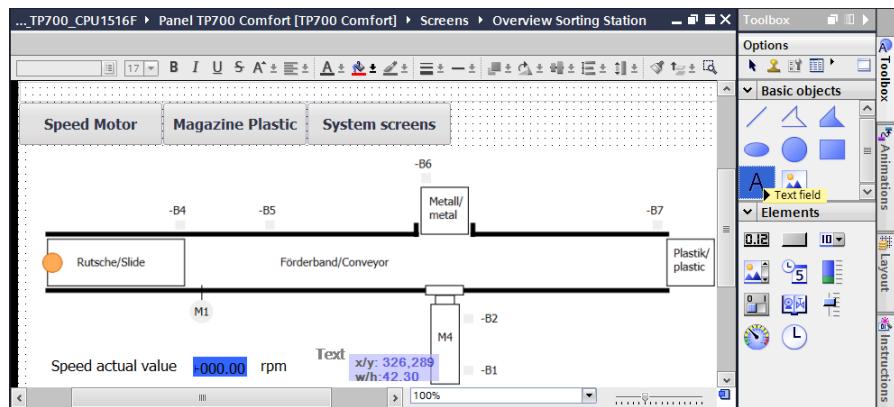
- Under "Text format" in "Properties", change the "Horizontal" setting of "Alignment" to → "Right".



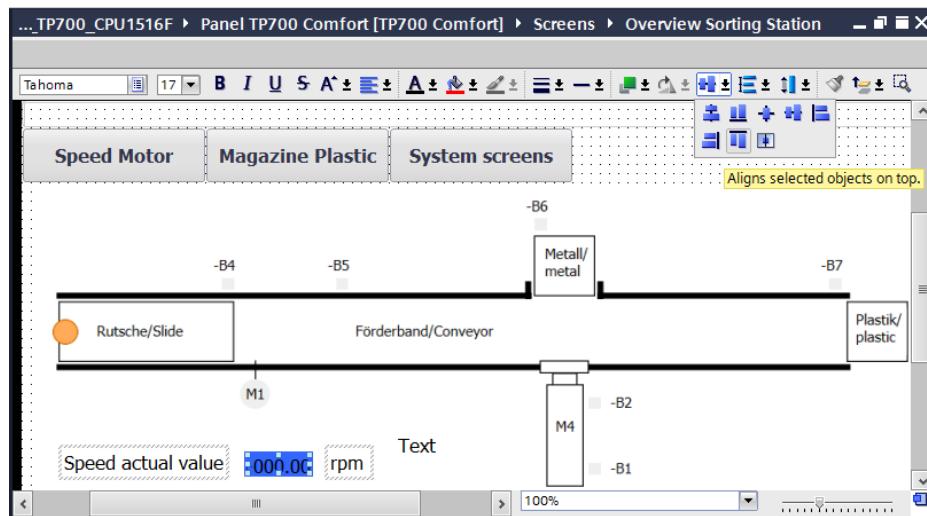
- Under "Layout" in the "Properties", change → "Position & Size" as illustrated here so that the IO field will be displayed below the conveyor motor.



- For the description, use drag-and-drop to insert a → "Text field" before and after the IO field from → "Basic objects" in "Tools". Type → "Actual speed value" and → "rpm" there.

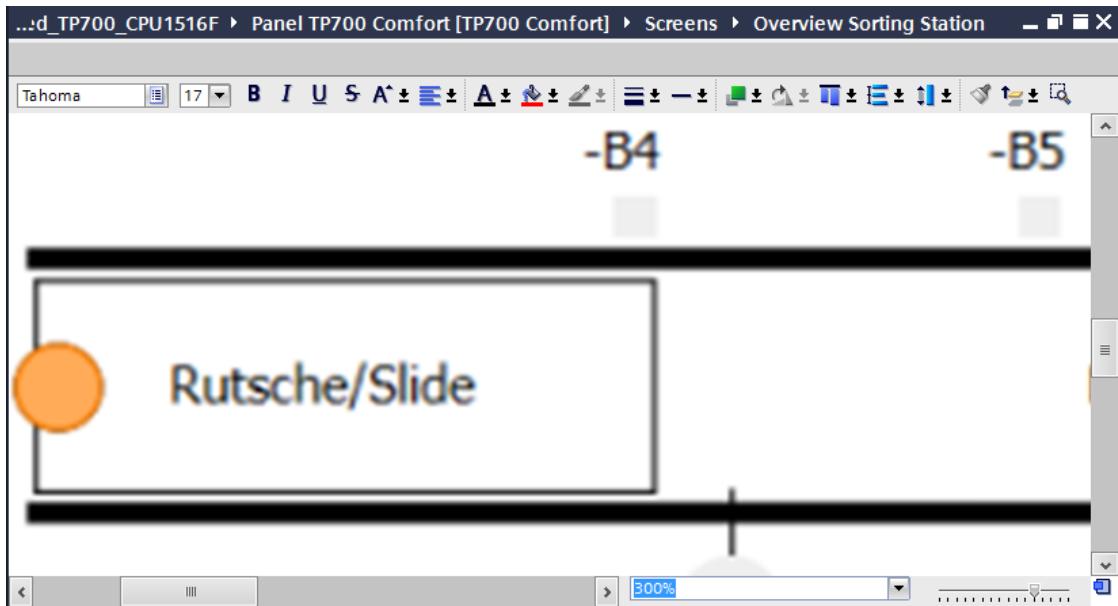


- Next, select the three objects → IO field → text field "Actual speed value" → and text field "rpm" in that order and then click on the → "Align selected objects on top" function in the toolbar of the work area. Save your project by clicking .

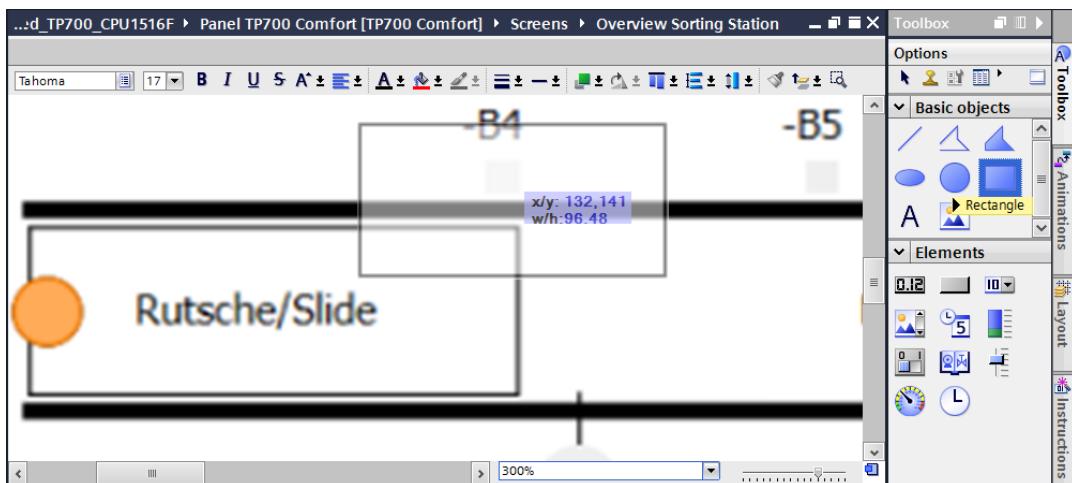


7.8 Visualizing binary signals with animated rectangles

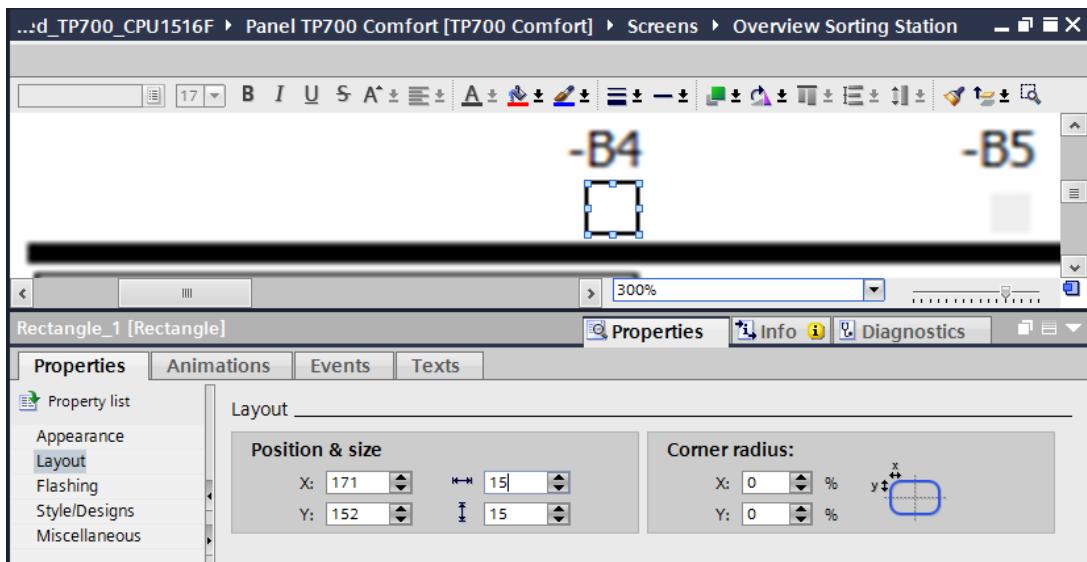
- For the visualization of the sensors, you want to start with sensor "-B4" at the slide. To allow you to draw and position the rectangle better, first change the zoom factor to → "300%".



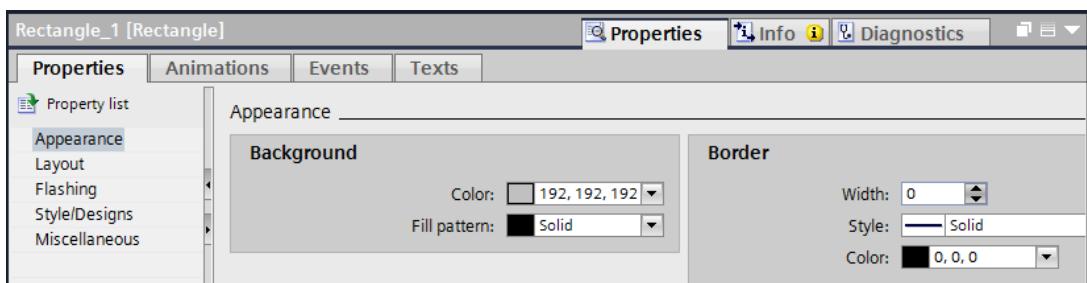
- Next, use "drag-and-drop" to move a rectangle from → "Basic objects" in Tools to the position of sensor "-B4".



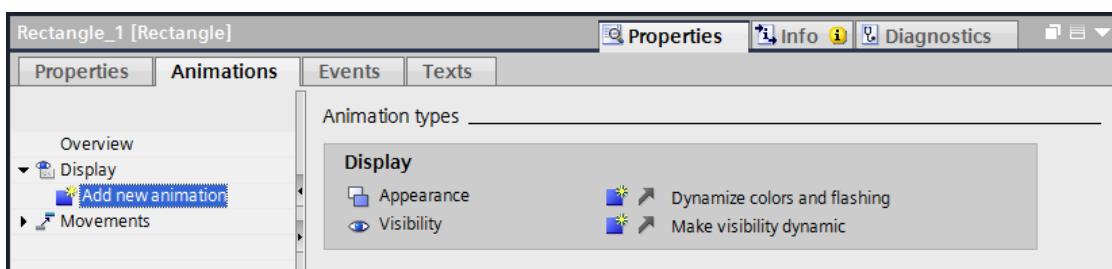
- Draw the correctly sized and positioned rectangle with the mouse or set the → "Position & Size" under "Layout" in "Properties" as shown here. The sensor will then be displayed below the designation "-B4".



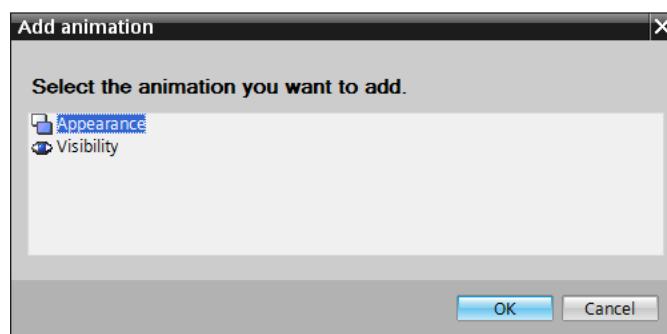
- In "Properties" under "Appearance", change the "Color" of the "Background" to → "Gray" and the "Width" of the "Border" to → "0".



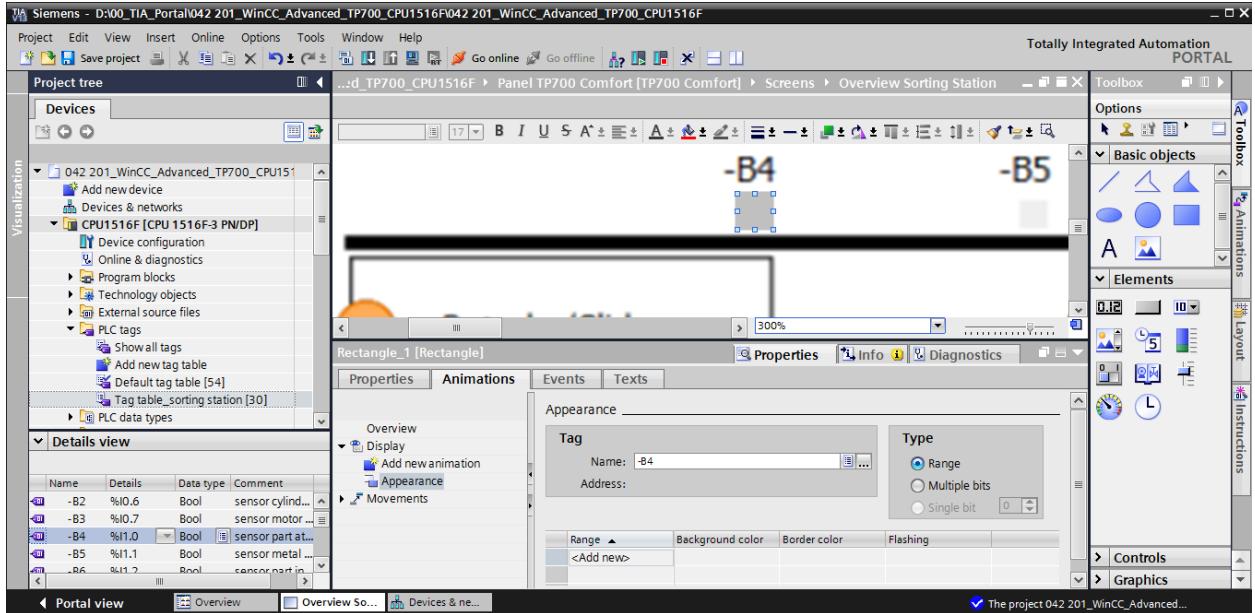
- Now change to the "Animation" tab and select "Display" and click
→ "Add new animation"



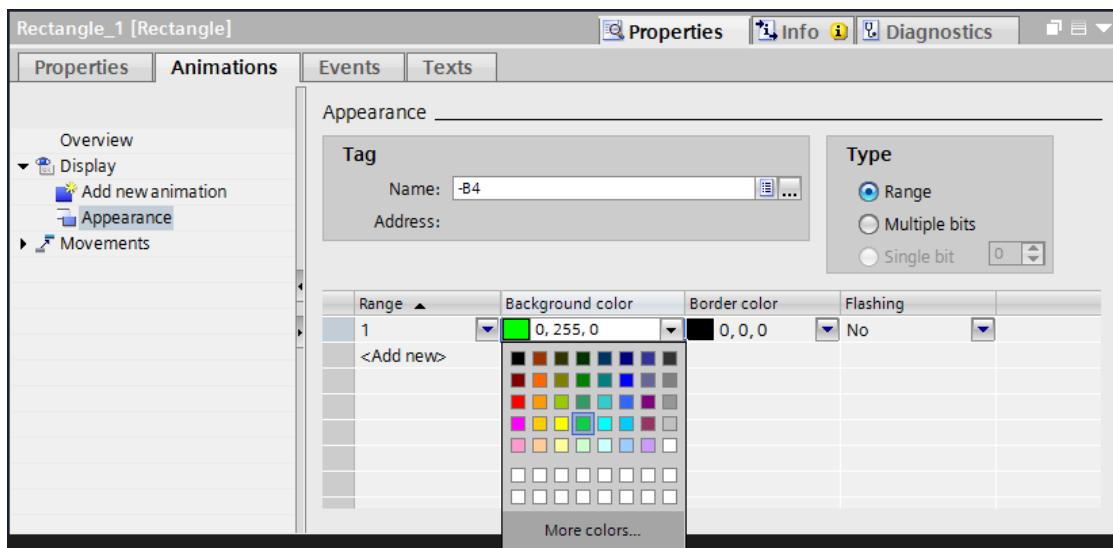
- In the displayed dialog, select → "Appearance" and click → "OK".



- To establish the connection to the global tag in the CPU, select → "PLC tags" under → "CPU_1516F" and below that the → "Tag_table_sorting_station". Next, drag the "-B4" tag from the → "Detail view" to the "Name" field for the "Tag".

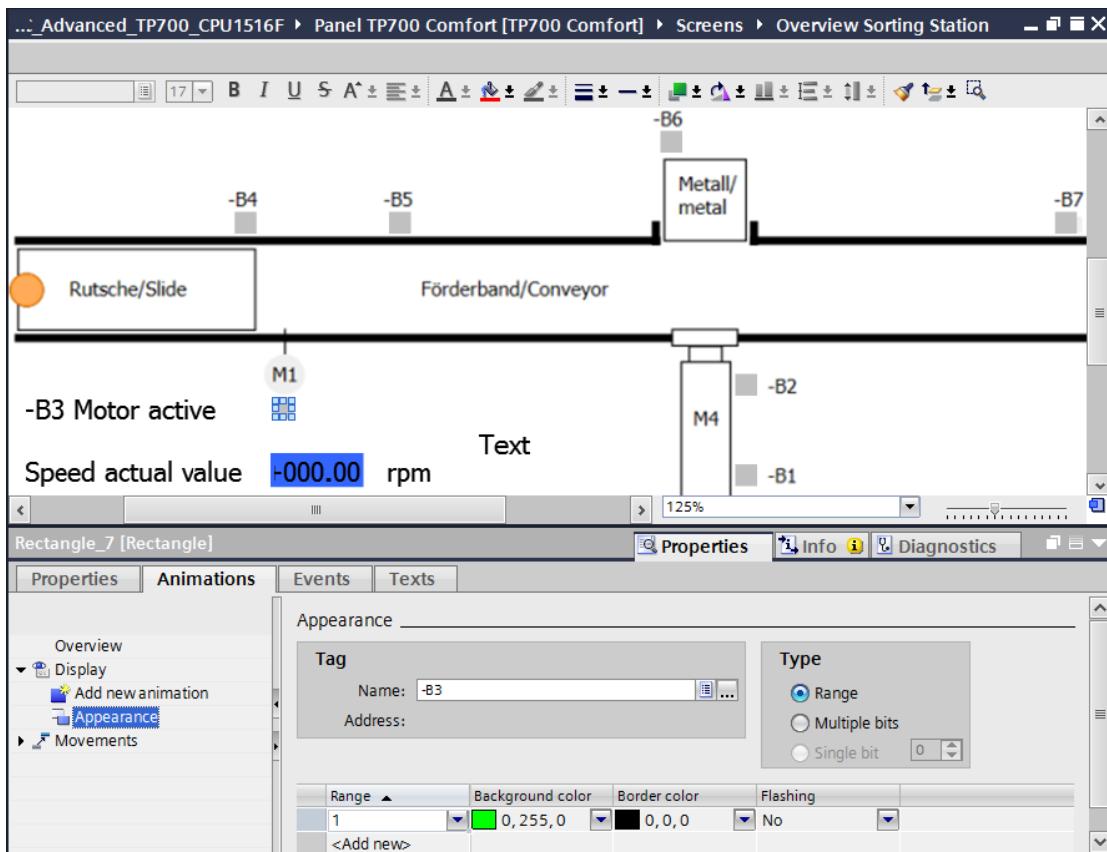


- In the "Appearance" of the "Display", add an area with the value → "1" (signal state "High") and change the "Background color" to → "Green".



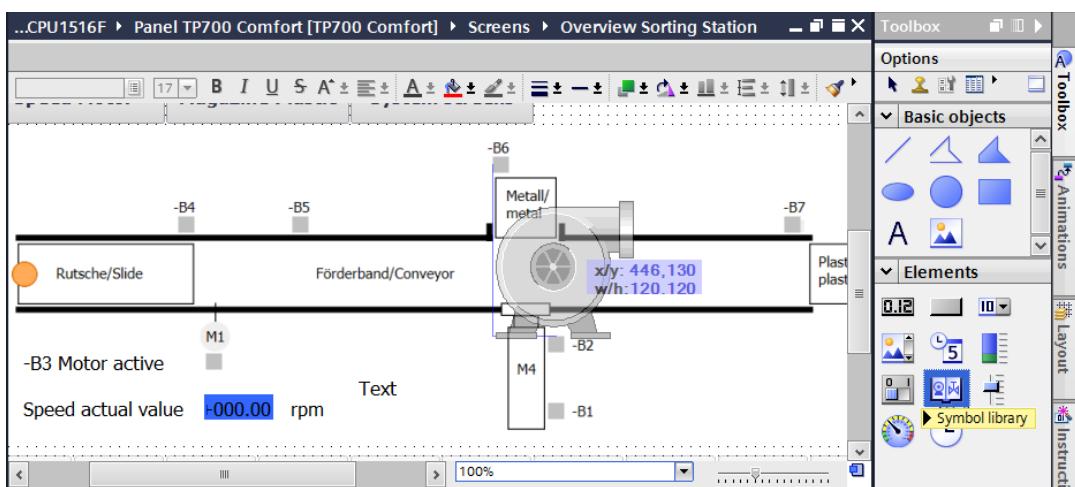
As shown in the previous steps, create a display for each of the following sensors:

- "-B1", → "-B2", → "-B5", → "-B6" and → "-B7".
- Also insert an additional binary display below the motor M1 and connect it to the global tag → "-B3". For the description, insert a text field → "-B3 Motor active" in front of it.

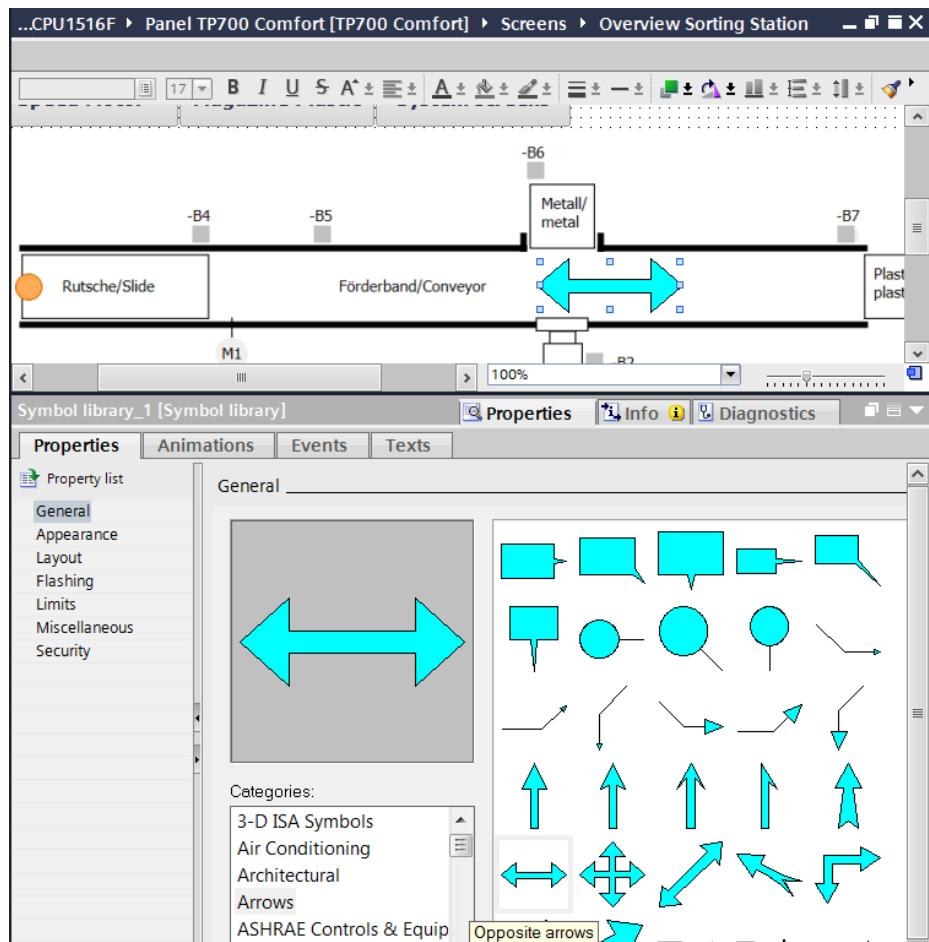


7.9 Symbol library

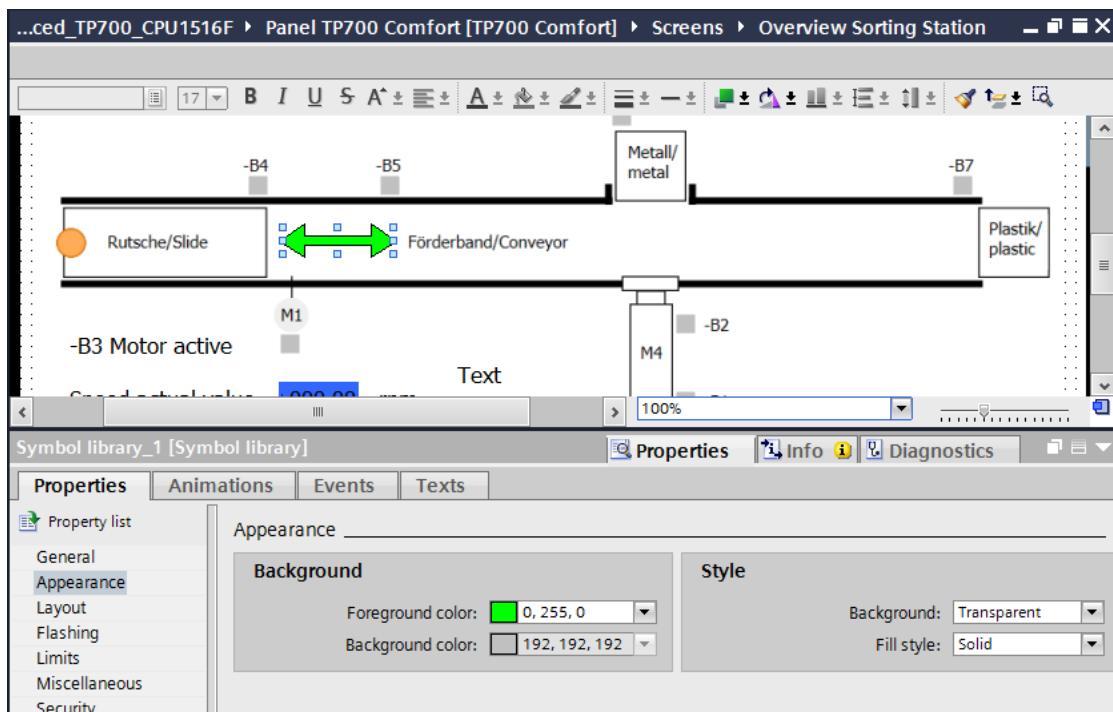
- In order to show that the conveyor is being controlled, use drag-and-drop to move the "Symbol library" object  from → "Elements" in Tools onto the conveyor.



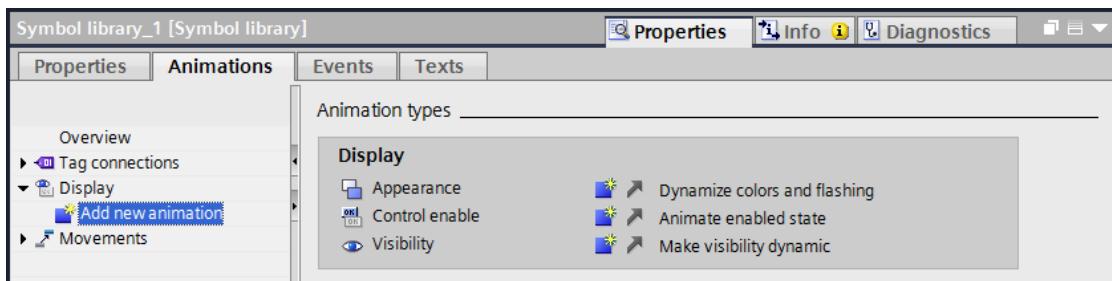
- The symbol library contains a number of simple and more complex graphic elements. Under "General" in "Properties", select the "Category" → "Arrow". Select an arrow pointing in the opposing direction here → .



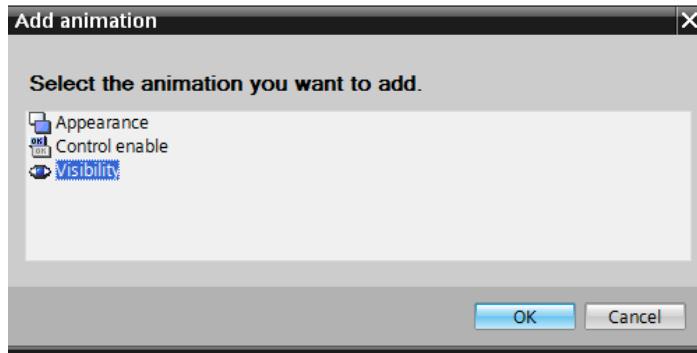
- Under "Appearance" in "Properties", change the "Style" of the arrow to → "Solid" and the "Color" of the "Foreground" to → "Green".



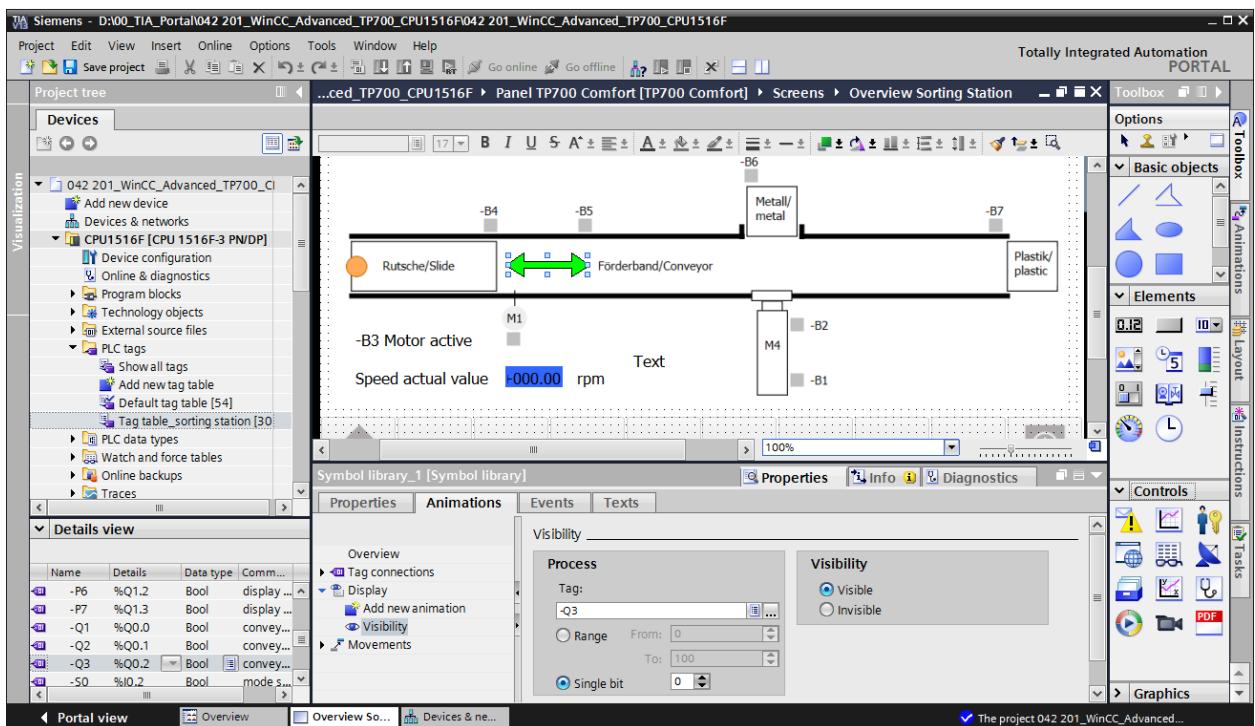
- Now change to the "Animation" tab and select "Display" and click → "Add new animation" .



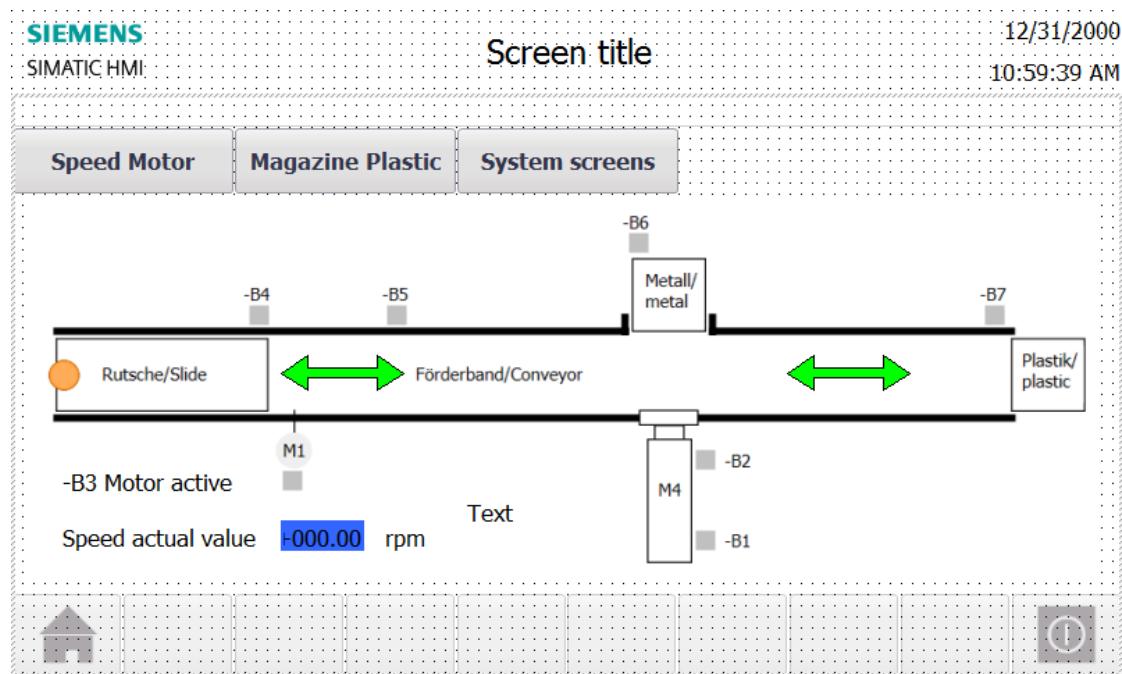
→ In the displayed dialog, select → "Visibility" and click → "OK".



→ To establish the connection to the global tag in the CPU, select → "PLC tags" under → "CPU_1516F" and below that → "Tag_table_sorting_station". Next, drag the "-Q3" tag from the → "Detail view" to inside the "Tag" field. In addition, select → "Single bit" as the type of evaluation.

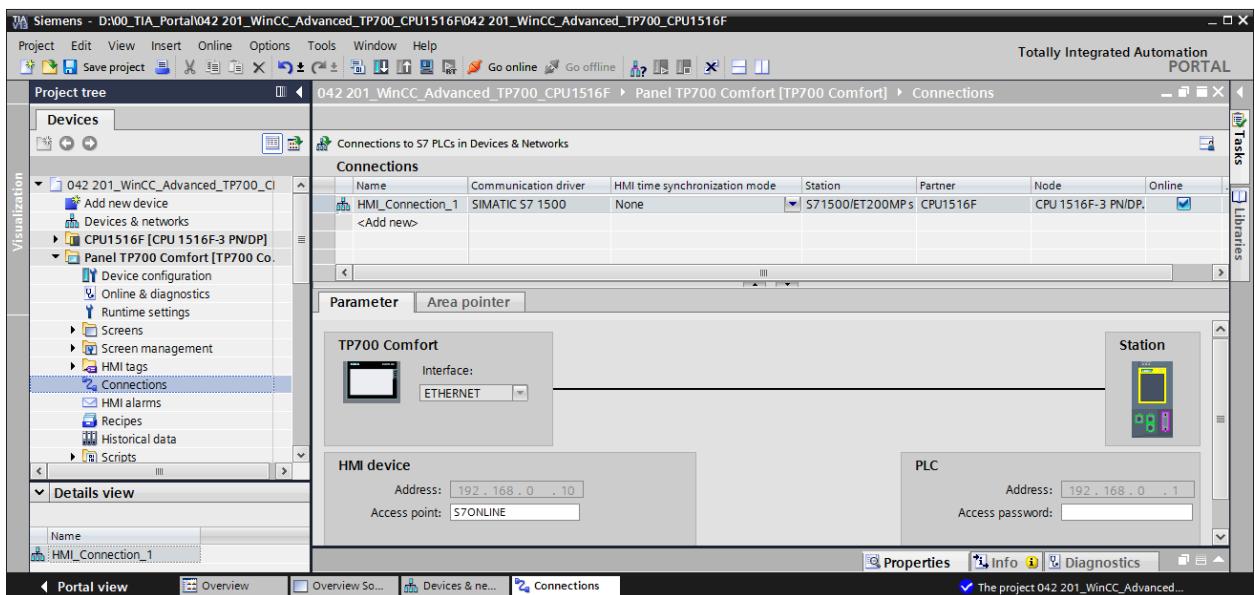


→ Next, duplicate the arrow from the symbol library with all its properties using → "Copy" and "Paste".



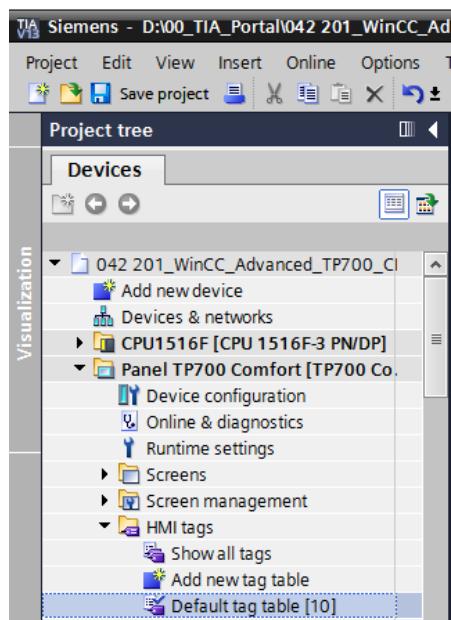
7.10 Connections and HMI tags

- Before you download the configuration to the TP700 Comfort Panel, you should check the connection to the CPU 1516F. To do so, double-click → "Connections" in → "Panel TP700 Comfort" to select it. In the displayed view, you can check the IP addresses and the connection settings. It is important that the check box for Online is selected.



Note: If access protection has been activated for the CPU 1516F, the access password for the panel can also be entered here.

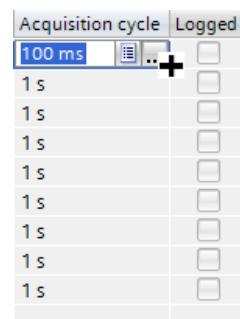
- To access the HMI tags, you must double-click the → "Default tag table" in the → "HMI tags" folder in → "Panel TP700 Comfort" to open it. All tags created by drag-and-drop have been entered here.

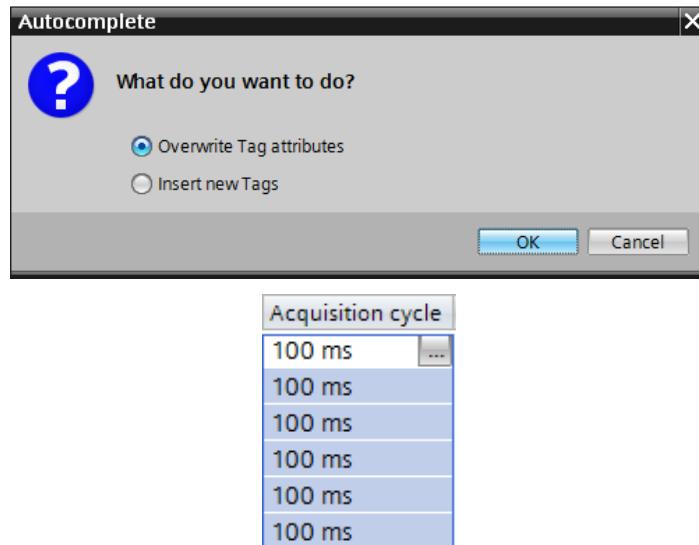


- In the default tag table, you can check which tags in the CPU 1516F will be accessed. You can also make additional settings. The "Acquisition cycle" of our tags is to be accelerated from 1 second to 100 milliseconds here. To do so, click the → selection field and select a new acquisition cycle → "100 ms" by double-clicking it.

The screenshot shows the 'Default tag table' dialog for the '042 201_WinCC_Advanced_TP700_CI' project. The main table lists tags such as -B1 through -B7, Q3, and SPEED_MOTOR_Speed_Actual_Value, each with its data type (Bool or Real), connection (HMI_Conne.. or HMI_Conne..), PLC name (CPU1516F), PLC tag, address, access mode (<symbolic access>), and acquisition cycle (set to 1 s). Below the table are tabs for 'Discrete alarms', 'Analog alarms', and 'Logging tags'. On the right, a context menu for the 'Cycles' entry in the 'Panel TP700 Comfort' tree shows a list of acquisition cycle names: None, 100 ms, 500 ms, 1 s, 2 s, 5 s, 10 s, 1 min, 5 min, and 10 min. The '100 ms' option is highlighted.

- You can make the settings of other tags using the "Autocomplete" function by selecting the lower right corner of the first entry with the mouse and dragging over the other entries.





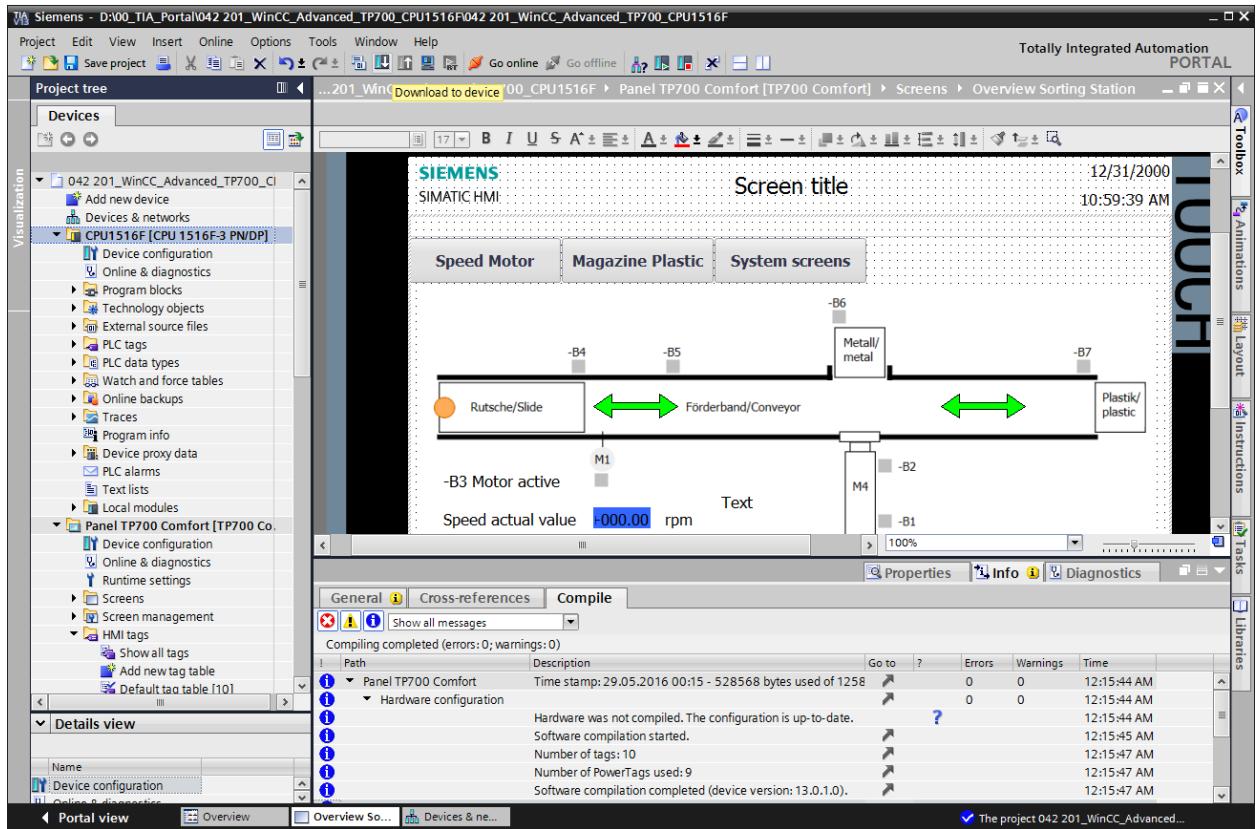
7.11 .Downloading the CPU and panel

- Before the project is downloaded to the CPU and the panel, compile the CPU and panel again and save the project.

(→ CPU_1516F → → Panel TP700 Comfort → →

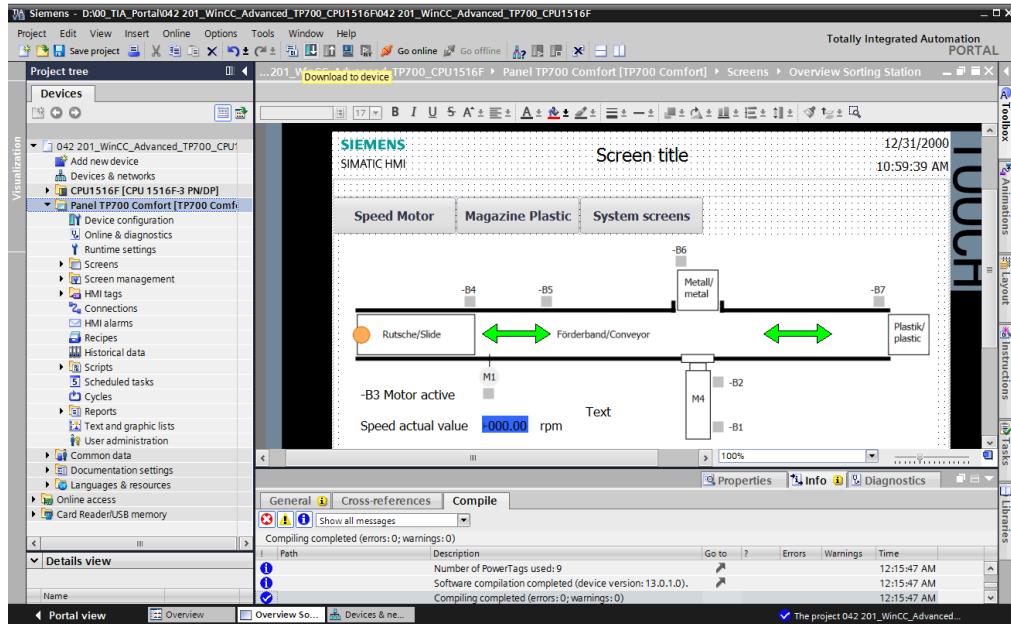
- After successful compilation, the complete controller with the created program including the hardware configuration can be downloaded, as described in the previous modules.

(→



- To download the visualization to the panel, proceed in a similar way. Select the → "Panel TP700 Comfort [TP700 Comfort]" folder and click the icon

→  "Download to device".



- The manager for configuring the connection properties (extended download) opens.

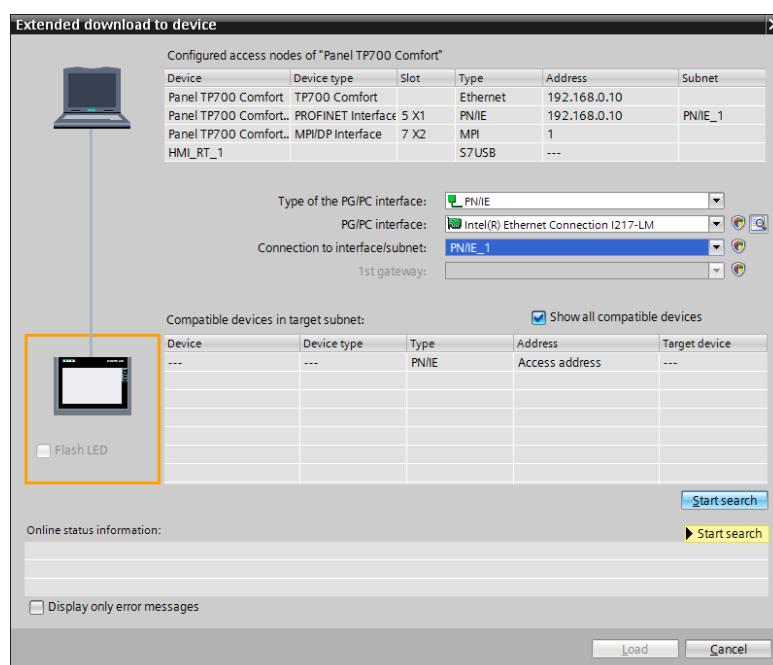
First, the interface must be correctly selected. This happens in three steps.

→ Type of the PG/PC interface → PN/IE

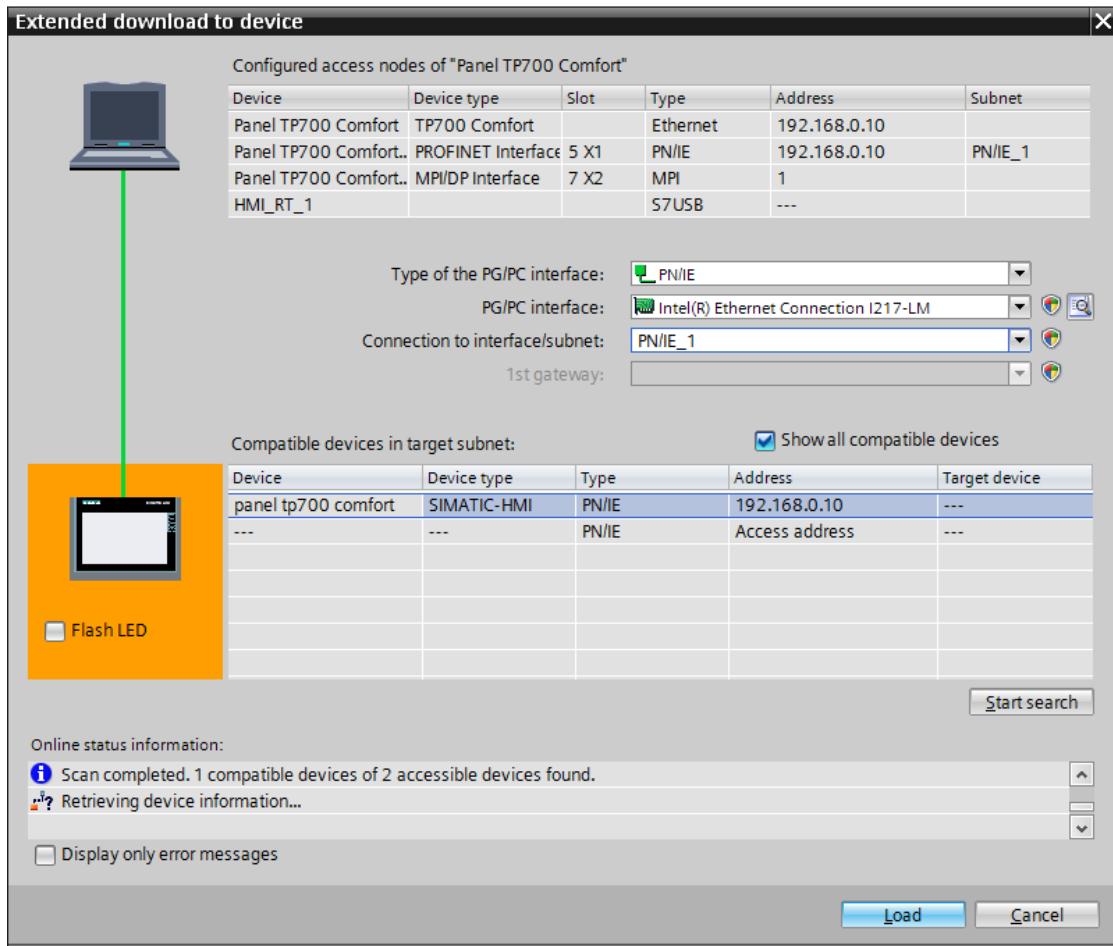
→ PG/PC interface → here, for example: Intel(R) Ethernet Connection I217-LM

→ Connection to interface/subnet → "PN/IE_1"

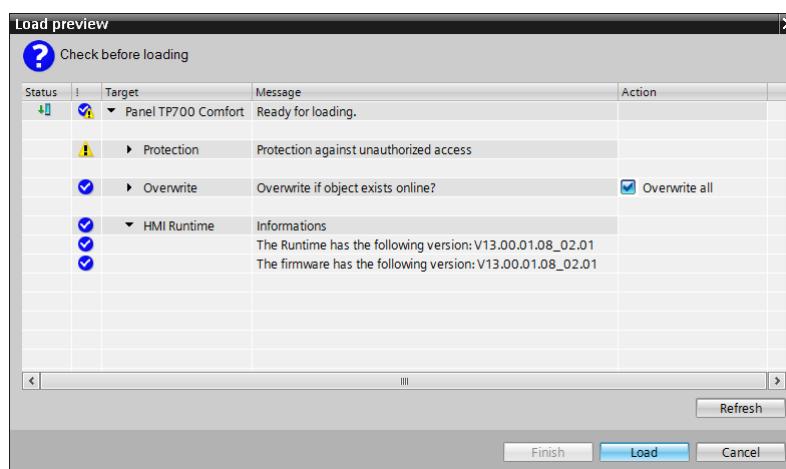
The → "Show all compatible devices" check box must then be selected. The search for devices in the network is started by clicking the → **Start search** button.



- If your panel is shown in the "Compatible devices in target subnet" list, it must be selected and the download started.
 (→ Panel TP700 Comfort → "Download")



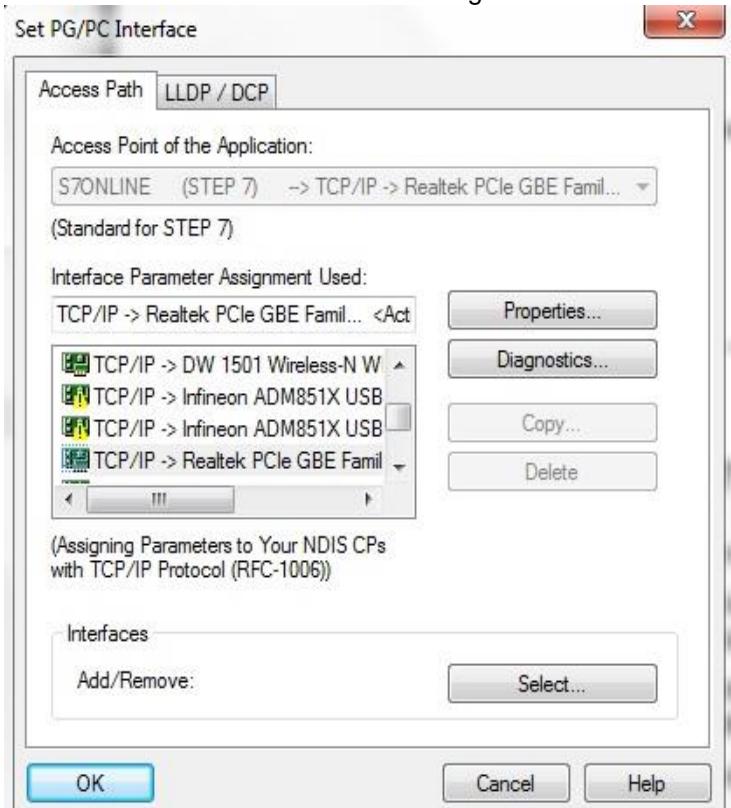
- You first obtain a preview. Confirm the prompt → "Overwrite all" and continue with → "Load".



Note: The ! symbol should be visible in every line of the "Load preview". You can find additional information in the "Message" column.

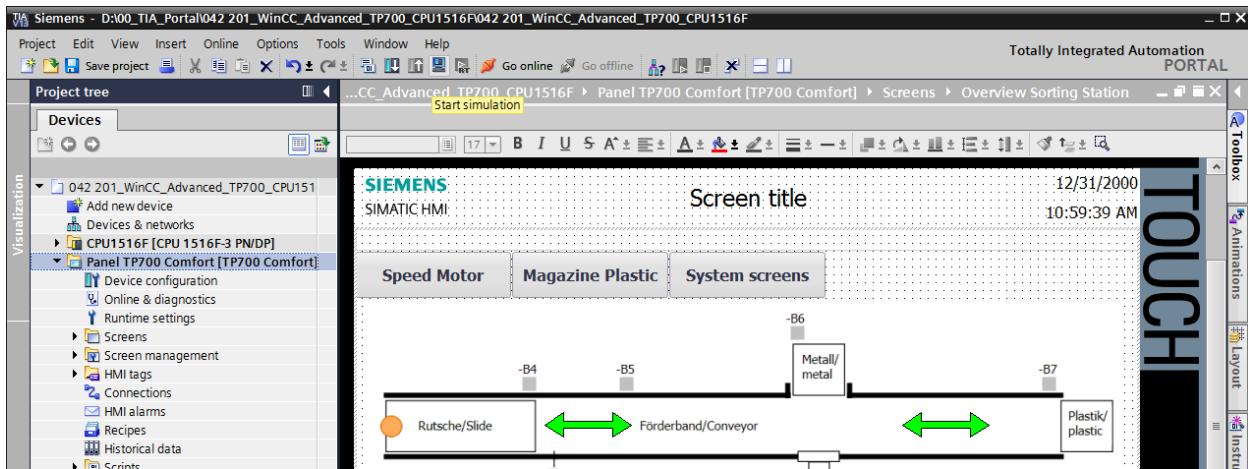
7.12 Process visualization in the simulation

For a connection to be set up between the runtime simulation on the PG/PC and the S7-1500 CPU, first the PG/PC interface has to be set to TCP/IP.

No.	Procedure:
1	<p>Open the Control Panel</p> <ul style="list-style-type: none"> with "Start > Control Panel" (Start menu for easy access to programs under Windows XP), or via "Start > Settings > Control Panel" (with class Start menu as is the case in previous Windows versions).
2	<p>In the Control Panel, double-click the icon "Set PG/PC Interface".</p>  <p>PG/PC-Schnitt stelle einstellen</p>
3	<p>Set the following parameters in the "Access Path" tab:</p> <ol style="list-style-type: none"> For the access point of the application, select "S7ONLINE [STEP 7]" from the drop-down list. From the list of interface parameter assignment used, select the interface "TCP/IP(Auto) -> with your network adapter that is connected directly with the Panel and the controller, e.g., 3Com EtherLink XL. Then click OK and confirm the next message with OK. 

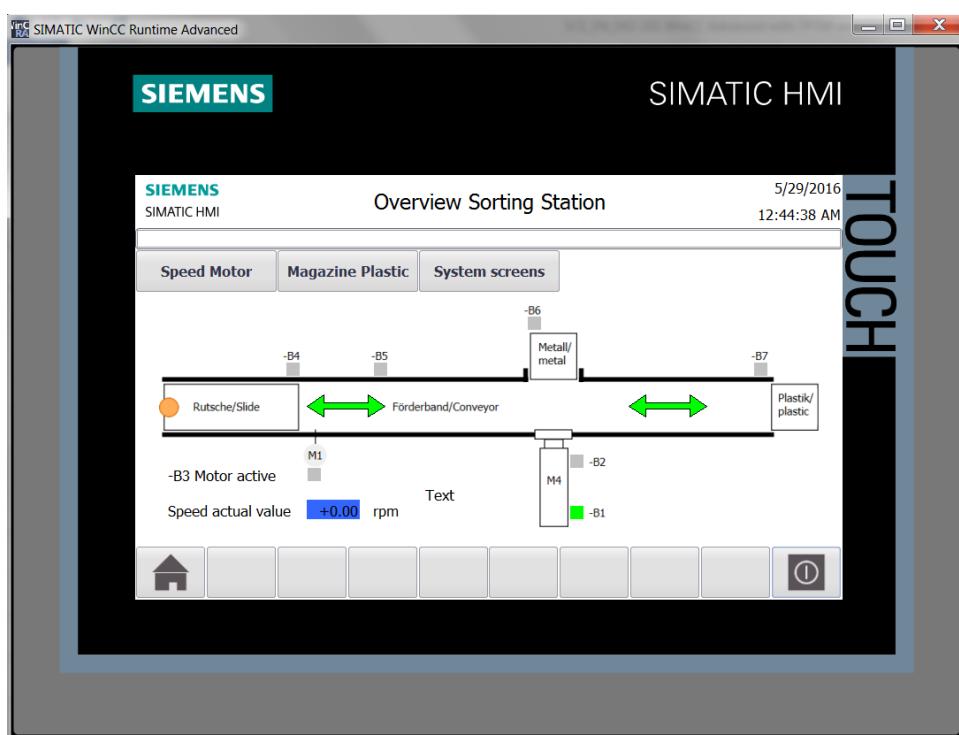
→ Select "Panel TP700 Comfort" and click the

→ "Start simulation" button.



→ The process visualization will be implemented completely on the PC with a connection to

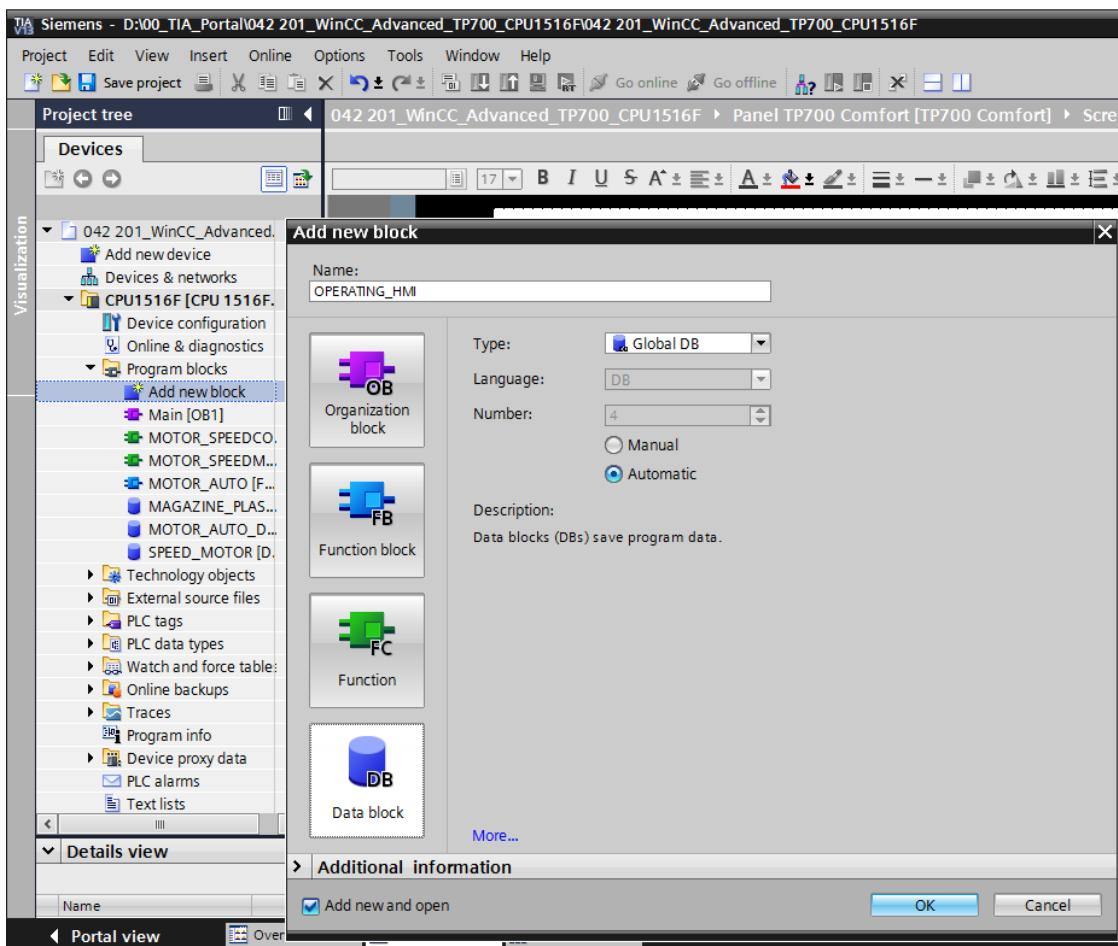
the process data in the CPU 1516F. To close the simulation, you can select the → button for "Exit Runtime" in the application or simply close the window by clicking
→ ".



7.13 Switches and buttons for the process operation

→ To have an interface to the process operation available in the PLC, select → "Add new block" in the "Program blocks" folder in "CPU_1516F" and create a global data block

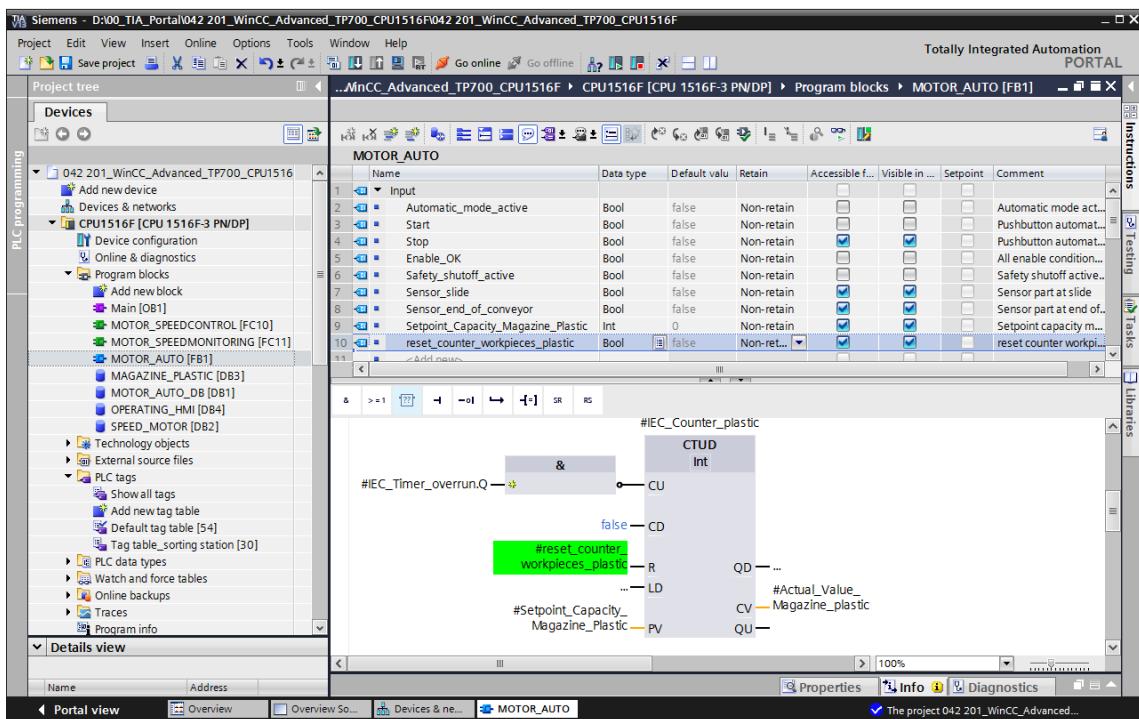
"OPERATION_HMI".



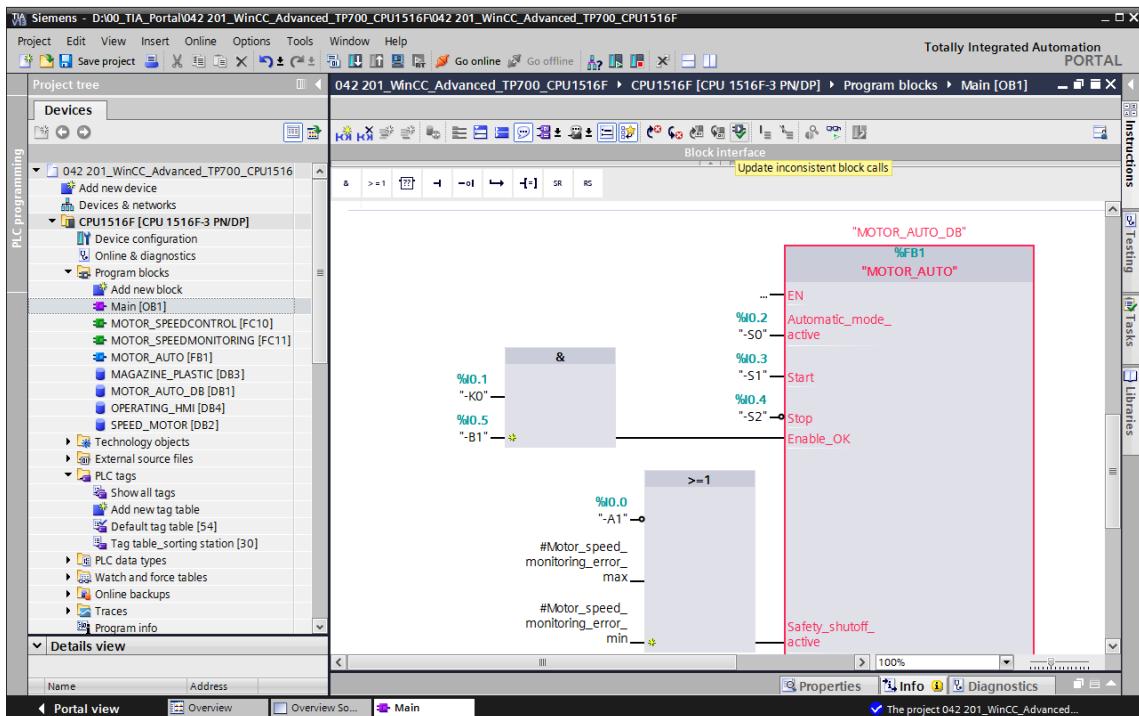
- In the "OPERATION_HMI" data block, create the following four tags of data type BOOL:
- "Mode selection", → "Automatic_Start", → "Automatic_Stop" and → "Reset_Counter_Plastic". The start value of the "Automatic_Stop" tag is preset with → "true".

OPERATING_HMI								
	Name	Data type	Start value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static							
2	mode_selector	Bool	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	HMI mode selector manual(0) / automatic(1)
3	automatic_start	Bool	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	HMI pushbutton automatic start
4	automatic_stop	Bool	true	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	HMI pushbutton automatic stop
5	reset_counter_plastic	Bool	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	HMI reset counter workpieces plastic
6	<Add new>							

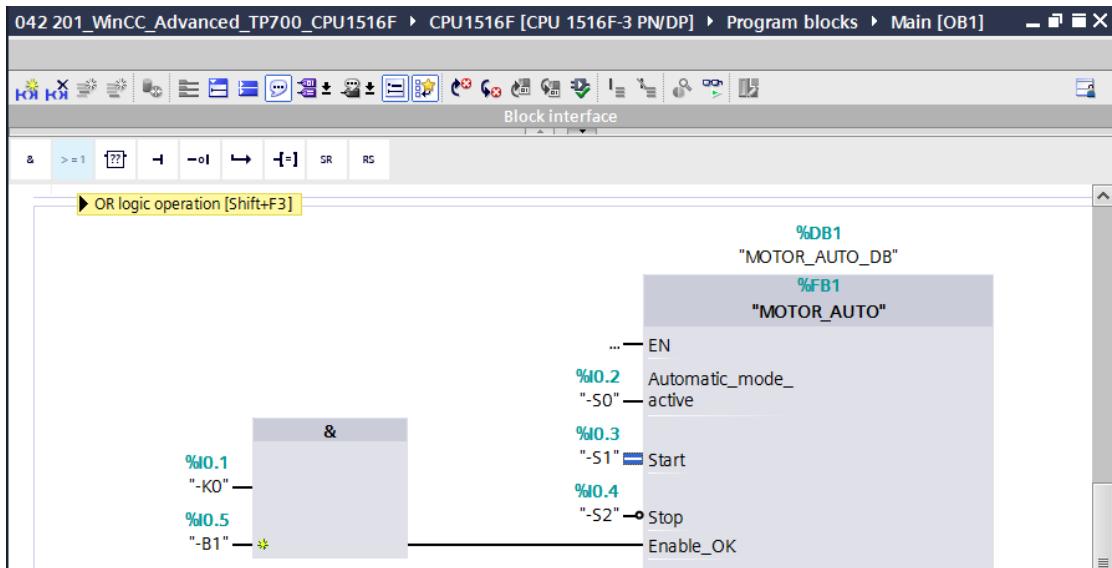
- The "MOTOR_AUTO[FB1]" function block is now expanded to include an input tag → "Reset_Workpiece_Counter_Plastic" of type → "Bool". This tag is moved onto the → "R" input of the "CTUD" counter in network 2 using drag-and-drop.



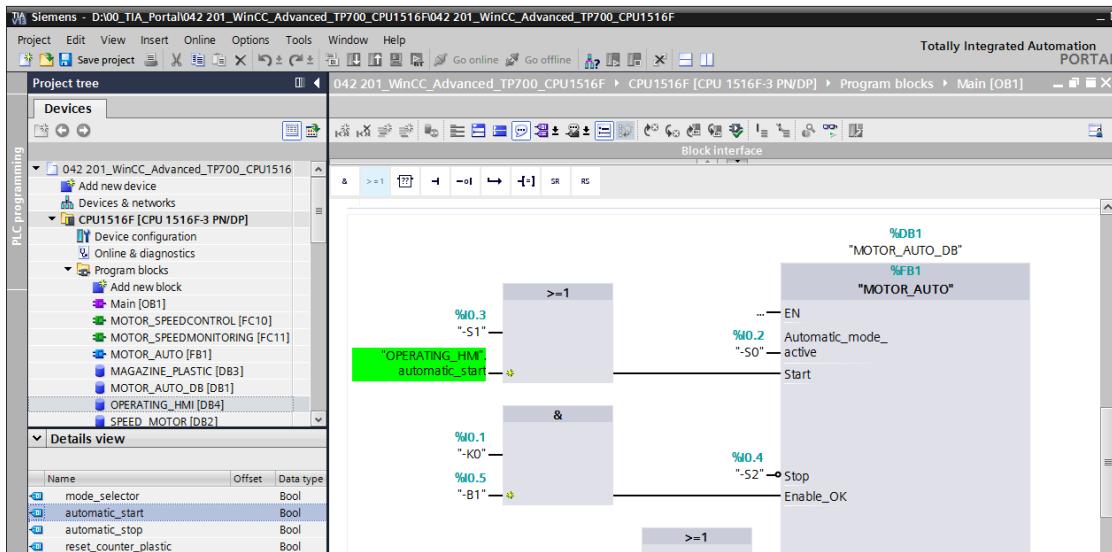
→ The call of the function block "MOTOR_AUTO[FB1]" must be updated in the "Main[OB1]" block. This is done by clicking the icon → "Update inconsistent block calls".



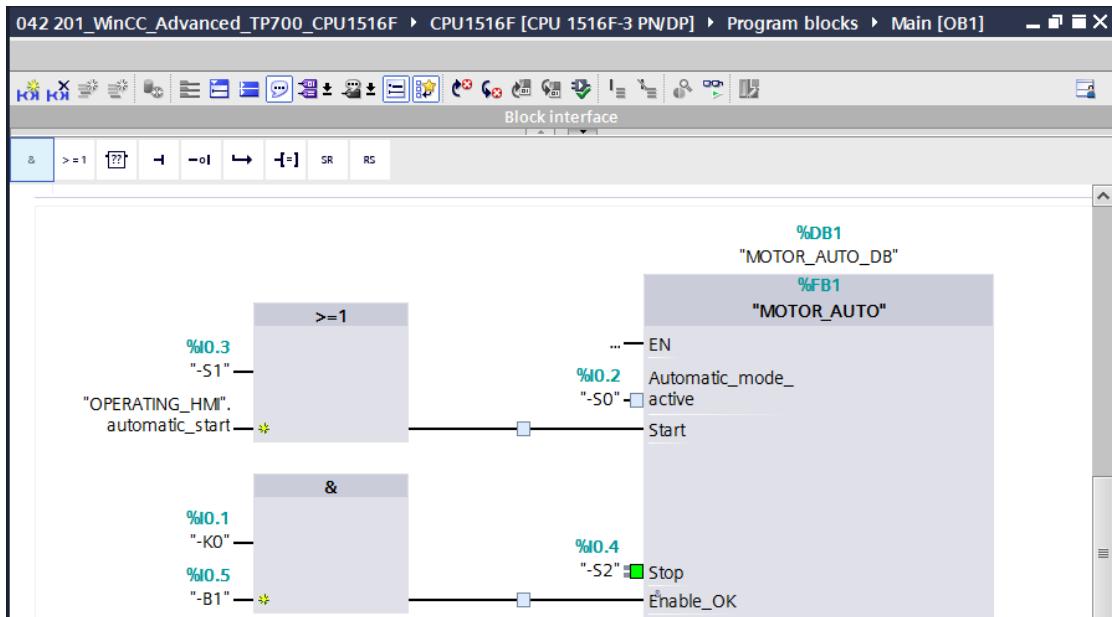
- In network 3 of the "Main[OB1]" block, drag an → "OR" in front of the input tag → "Start_command".



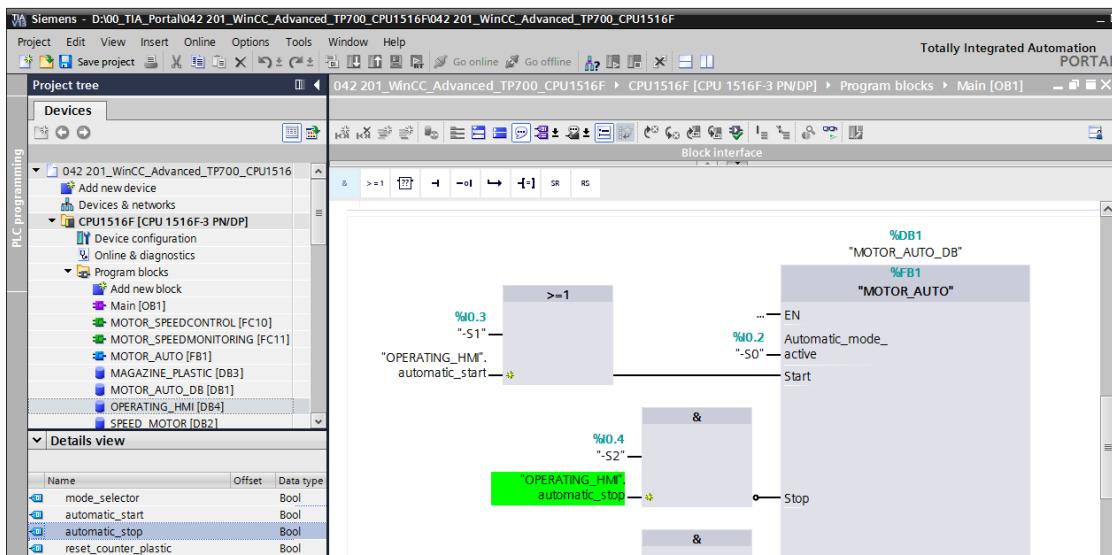
- The second free input of the → "OR" is connected to the → "Automatic_Start" tag from the "OPERATING_HMI" data block.



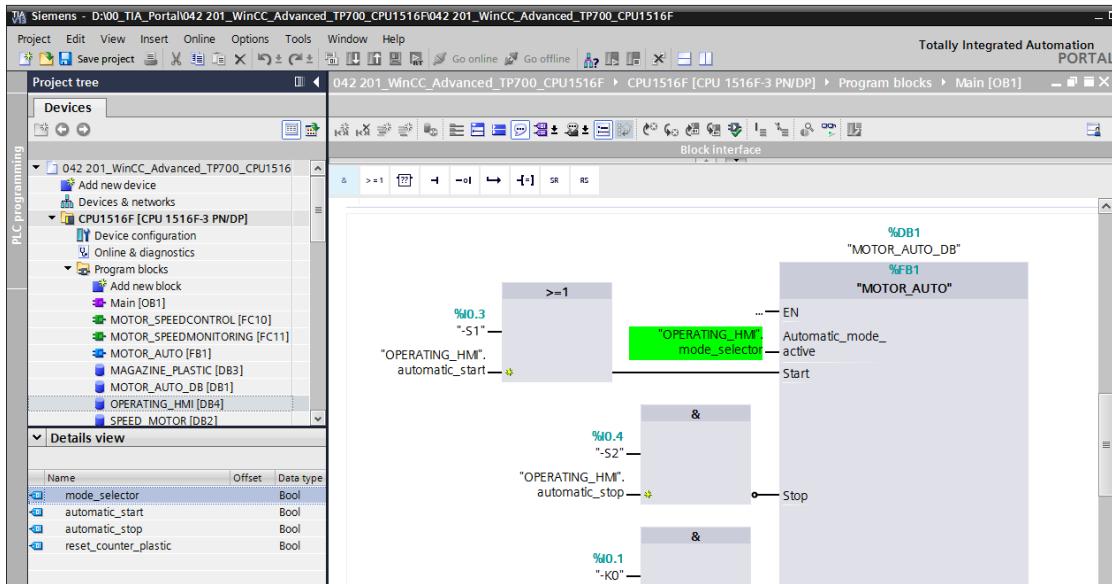
- In network 3 of the "Main[OB1]" block, drag an → "AND" in front of the input tag → "Stop_command".



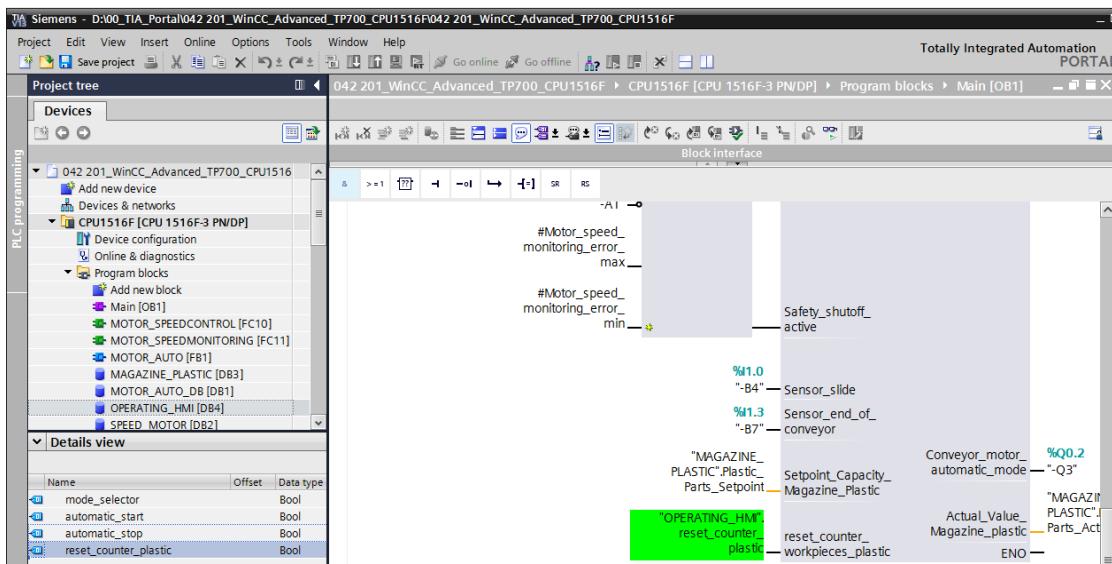
- The second free input of the → "AND" is connected to the → "Automatic_Stop" tag from the "OPERATION_HMI" data block.



- The input tag → "Automatic_mode_active" is connected to the → "Mode selection" tag from the "OPERATION_HMI" data block.



- The input tag → "Reset_part_counter_plastic" is connected to the → "Reset_counter_plastic" tag from the "OPERATION_HMI" data block.

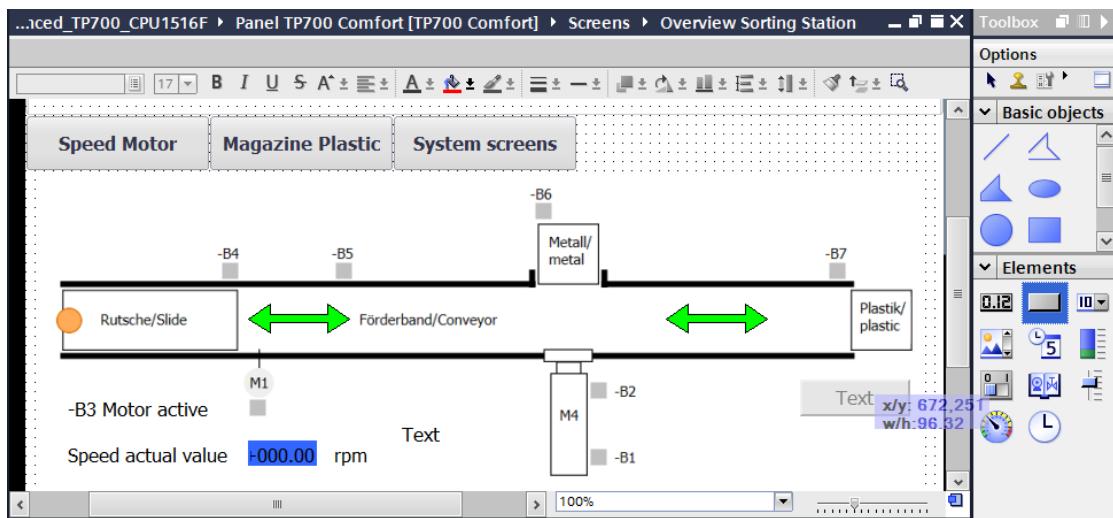


- Compile the CPU again and save the project.

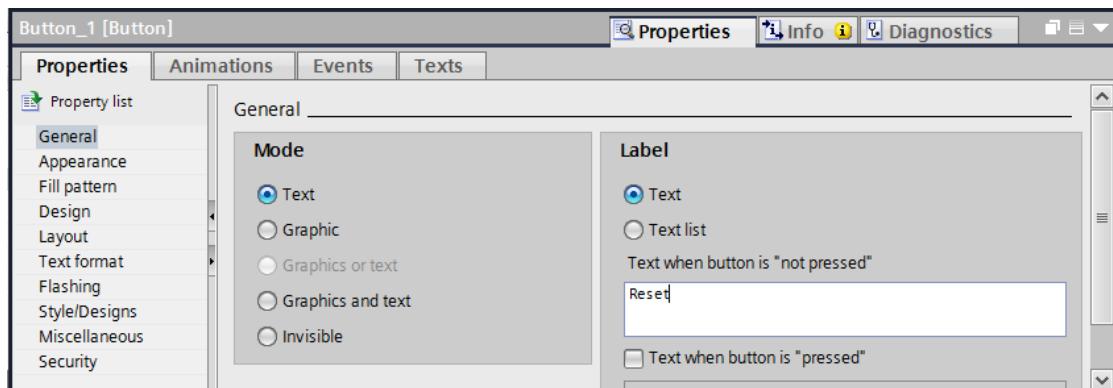
(→CPU 1516F → → Save project)

- You download the modified program including the hardware configuration to the CPU 1516F. (→ 

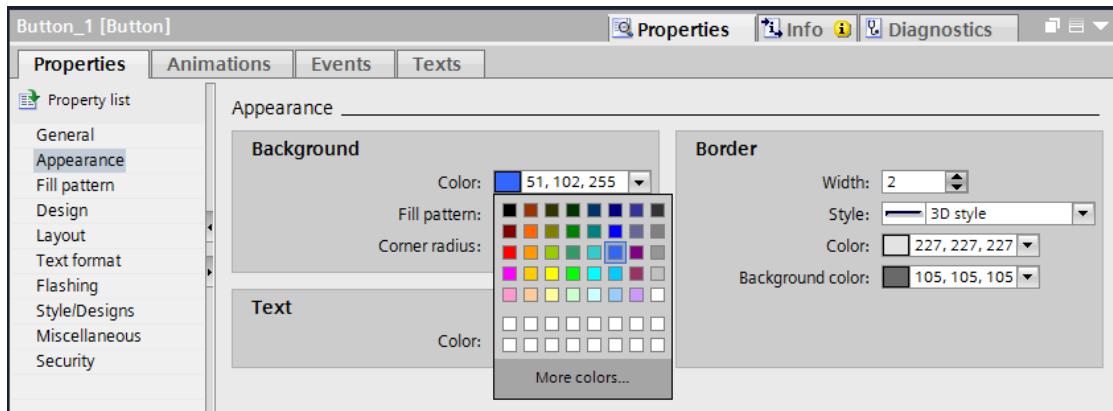
- In order to implement a pushbutton that resets the workpiece counter for the plastic parts, use drag-and-drop to move the → "Button" object from → "Elements" in "Tools" to a location below the plastic parts storage.



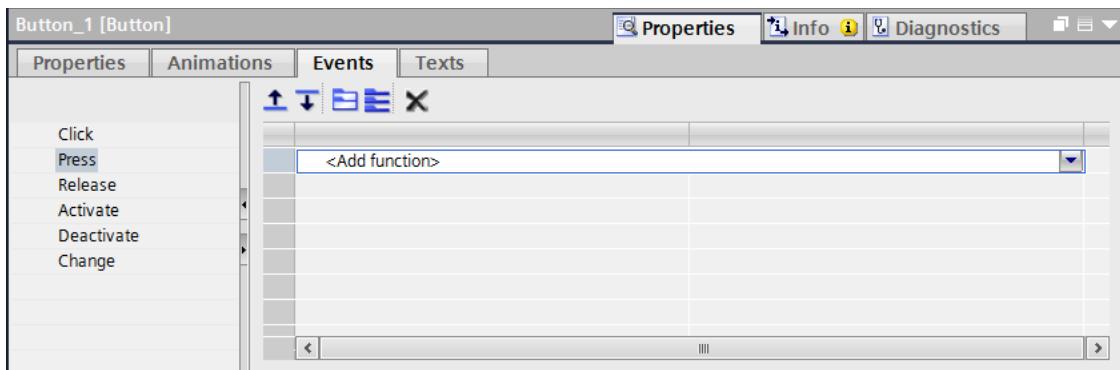
- Under "General" in "Properties", enter → "Reset" as the label.



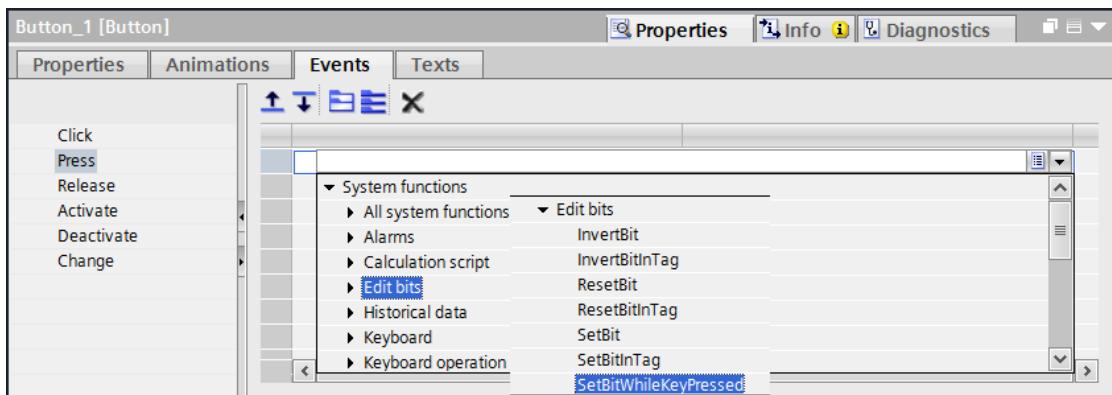
- Under "Appearance" in "Properties", change the "Color" of the "Background" to → "Blue".



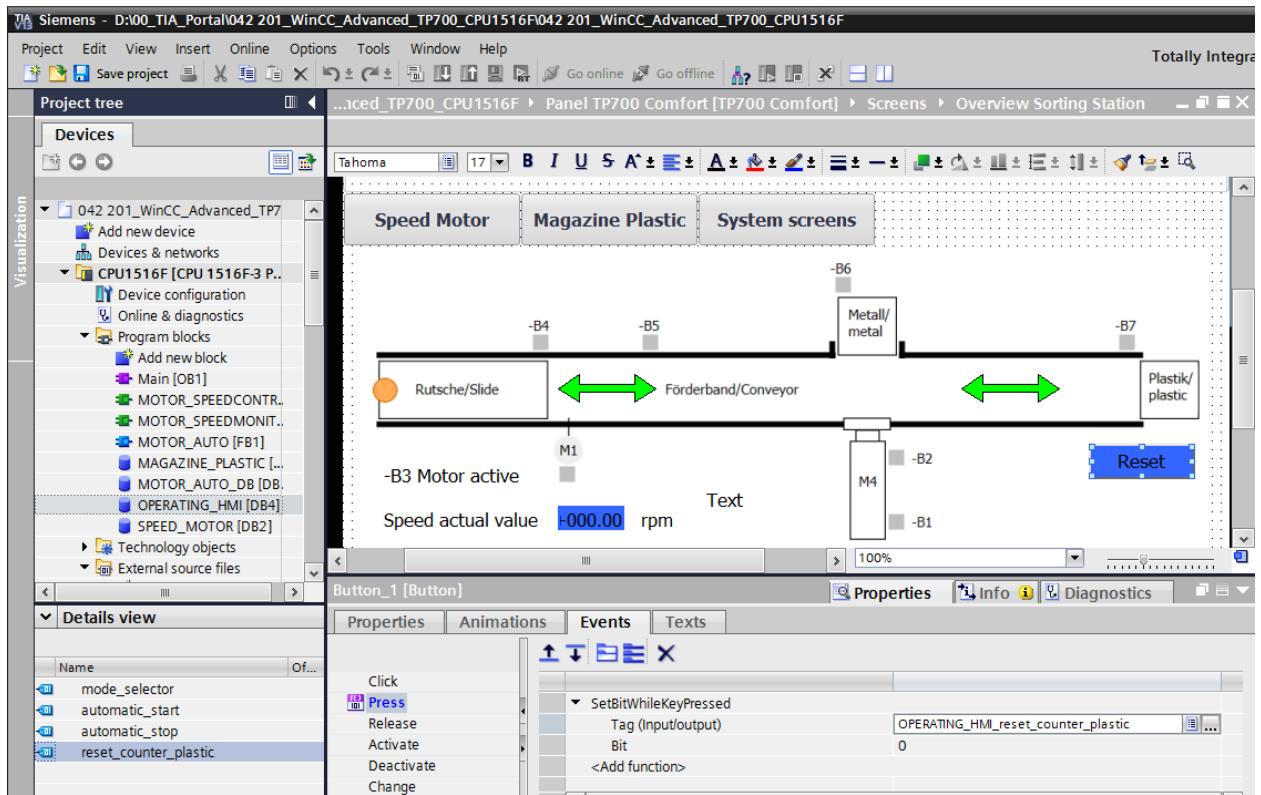
- The functionality must now be configured as a pushbutton. For this, change to the "Events" menu, select → "Press" as the event and → "<Add function>".



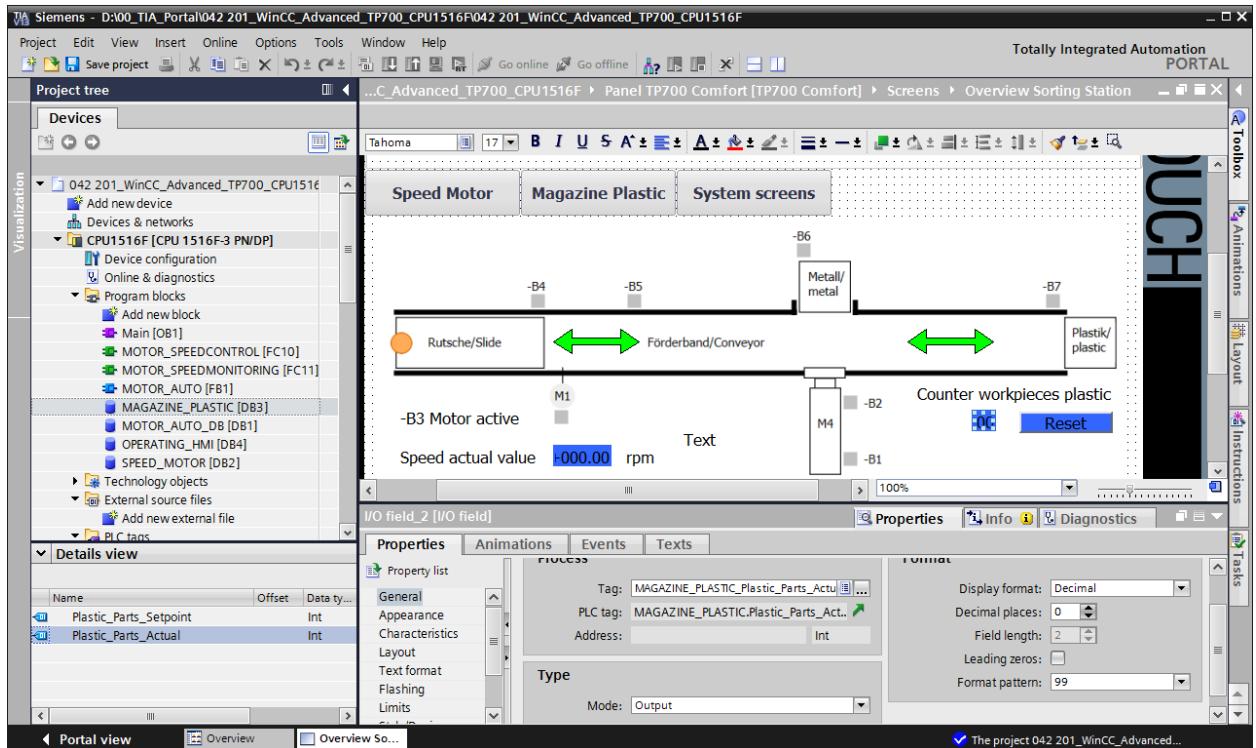
- Under "System functions", select "Bit processing" as the function and select → "SetBitWhileButtonPressed" there.



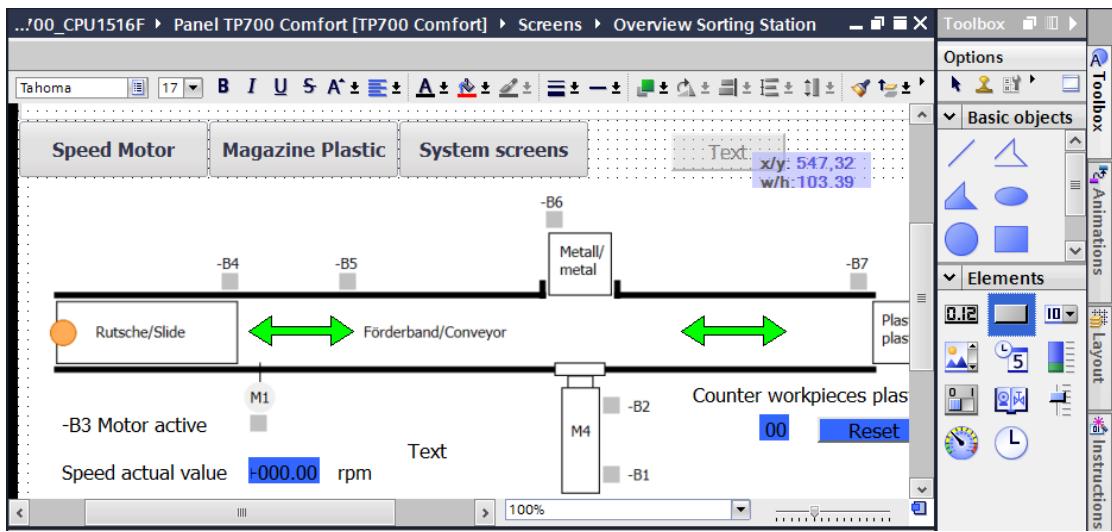
- For the process connection, select the → "Program blocks" of the → "CPU_1516F" and the → "OPERATION_HMI[DB4]" data block there. Then drag the → "Reset_counter_plastic" tag from the → "Detail view" into the field for "Tag (input/output)".



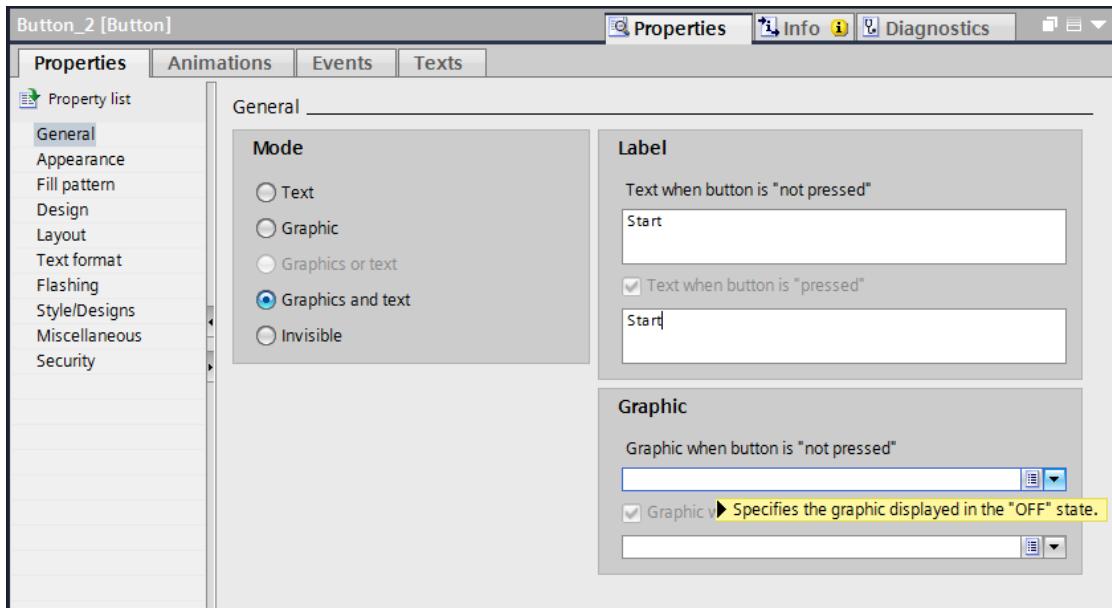
- As shown previously in the document, now insert a text → "Plastic parts counter" above the button and a display of the → "Plastic_parts_actual" tag from the "MAGAZINE_PLASTIC[DB3]" block to the left of the button.



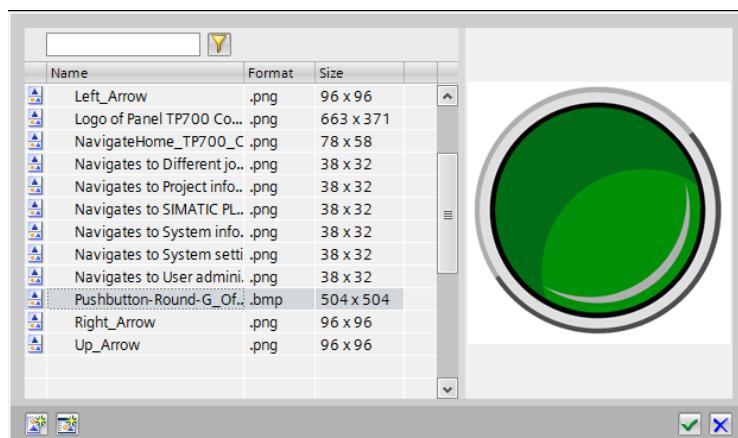
- In order to implement the pushbutton, use drag-and-drop to move the →"Button" object from → "Elements" in "Tools" to a location at the top next to the button for the screen change.



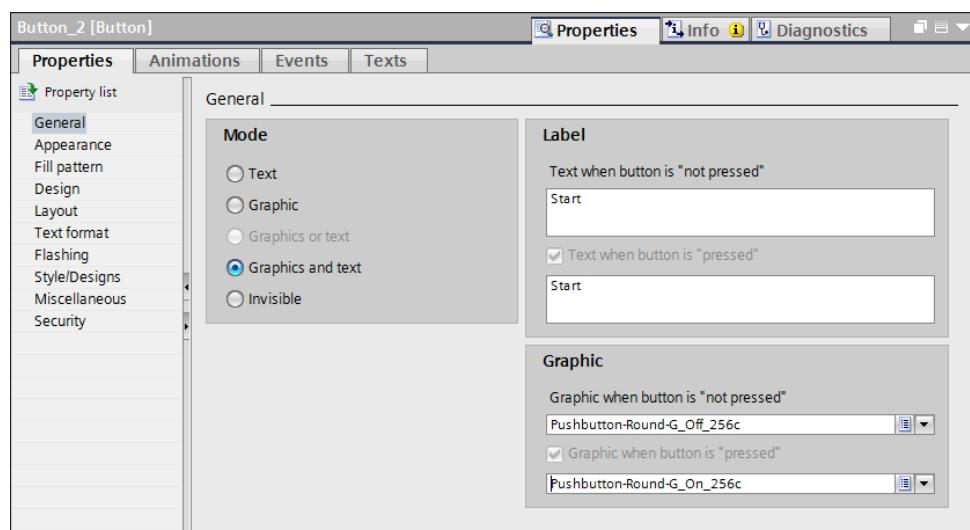
- Under "General" in "Properties", change the "Mode" to → "Graphics and text". To do this, open the selection dialog for → "Graphic when button is not pressed" by clicking the icon.



- Then click the icon for "Create graphic from file" and select the "Pushbutton-Round-G_Off_256c.bmp" file from the "SCE_EN_042-201_Screens" folder by double-clicking it.

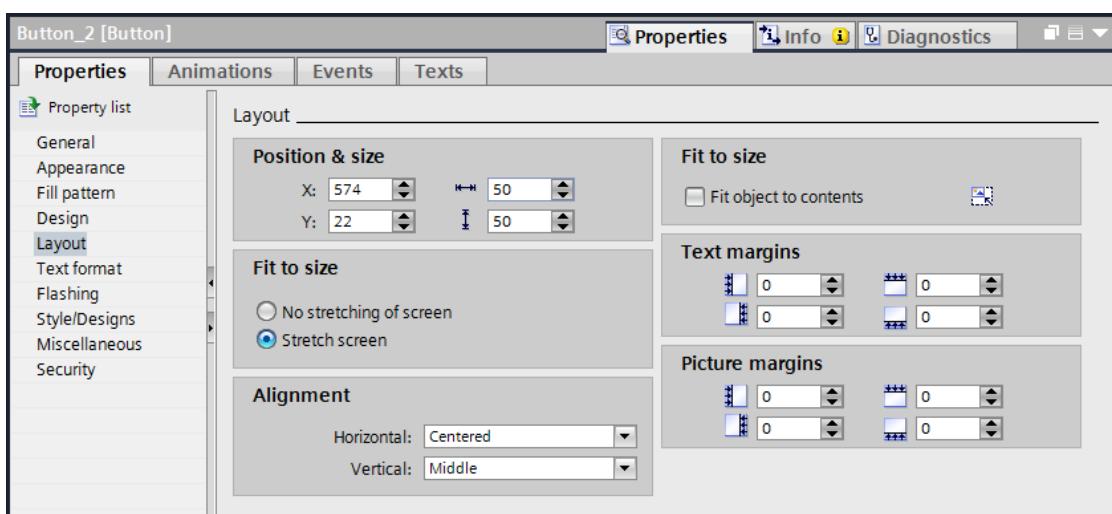


- In exactly the same way, select the "Pushbutton-Round-G_On_256c.bmp" file from the "SCE_EN_042-201_Screens" folder for the "Graphic when button is pressed".

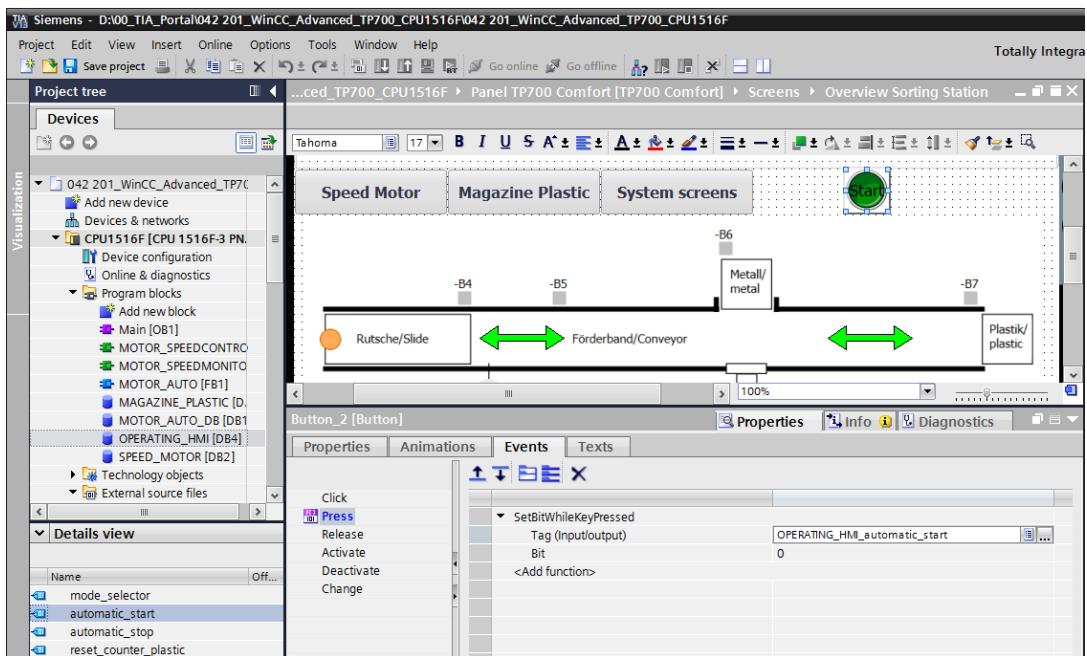


Note: The generated graphics are stored in the "Languages & Resources" path under "Project graphics".

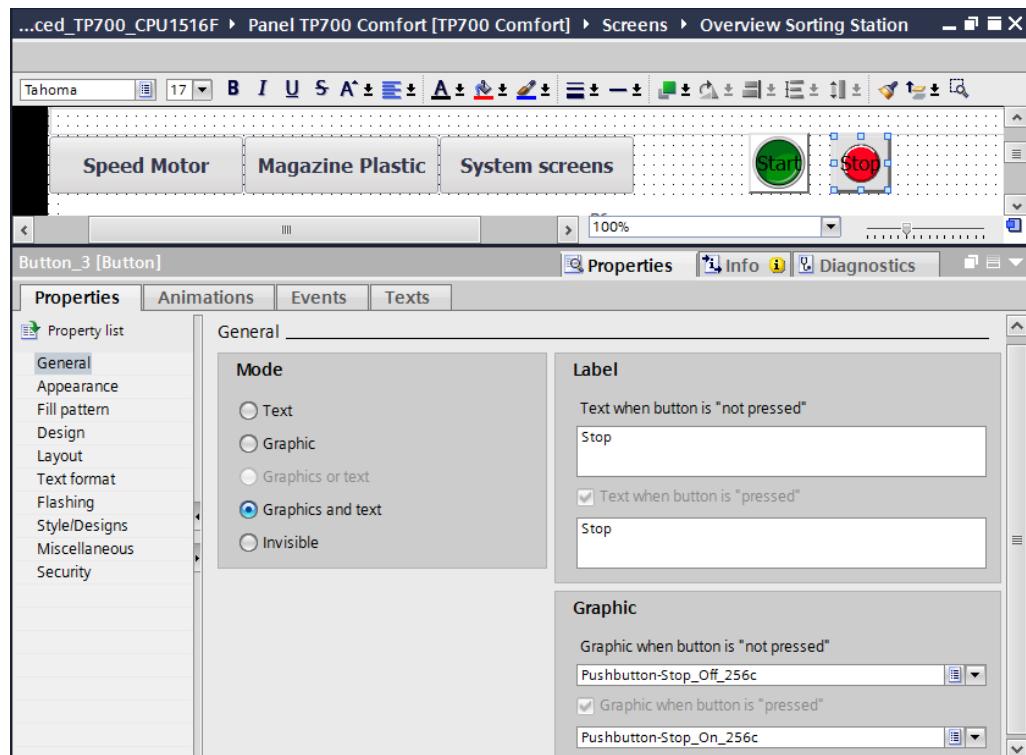
- Under "Layout" in "Properties", adapt the size of the button under → Position & Size.



- The pushbutton functionality is realized here again as event → "Press" with "System function" → "SetBitWhileButtonPressed". The → "Automatic_Start" tag from the data block → "OPERATION_HMI[DB4]" is used for the process connection.

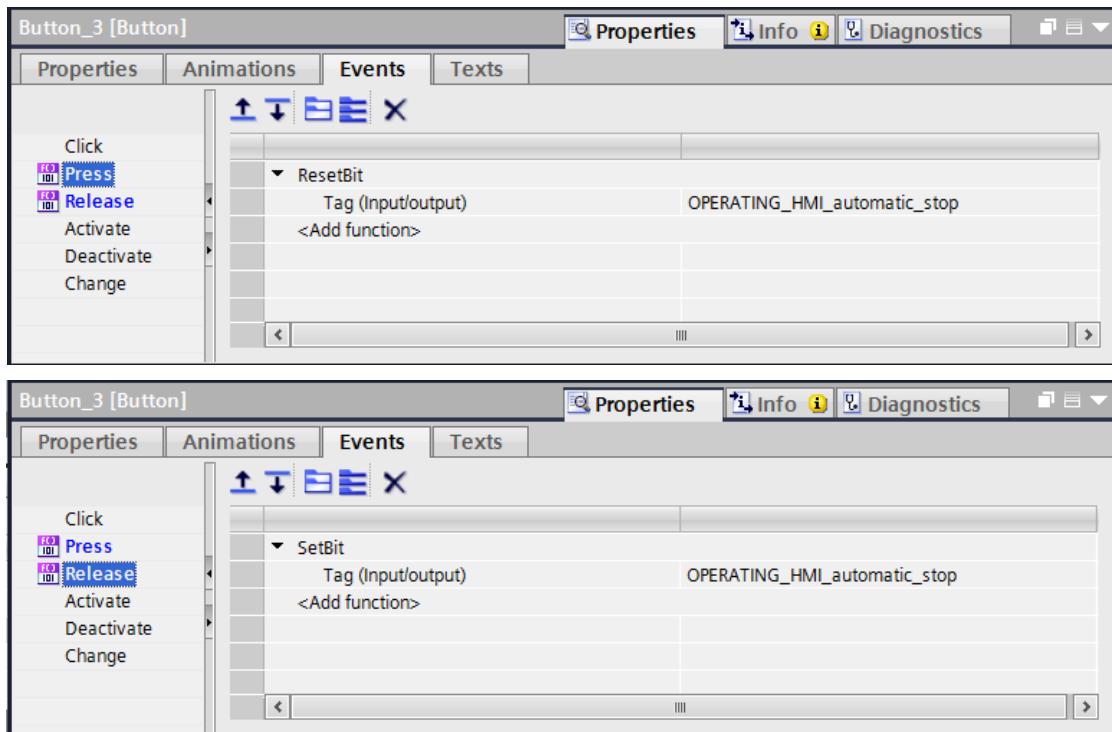


- As shown in the last steps, a "button" for the Stop button is inserted. The files "Pushbutton-Stop_Off_256c.bmp" and "Pushbutton-Stop_On_256c" from the "SCE_EN_042-201_Screens" folder are used as graphics.

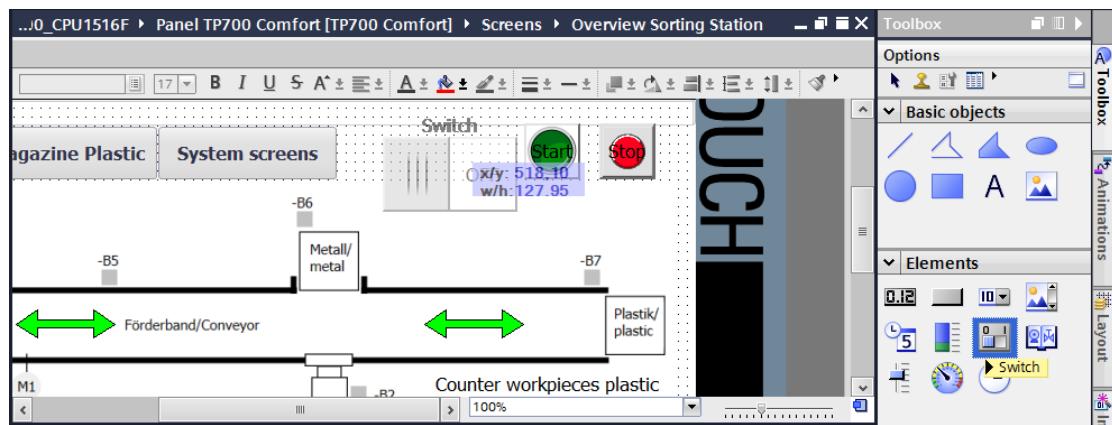


- The functionality as a normally-closed switch is implemented with two events. The first event is → "Press" with "System function" → "ResetBit" and the second event →

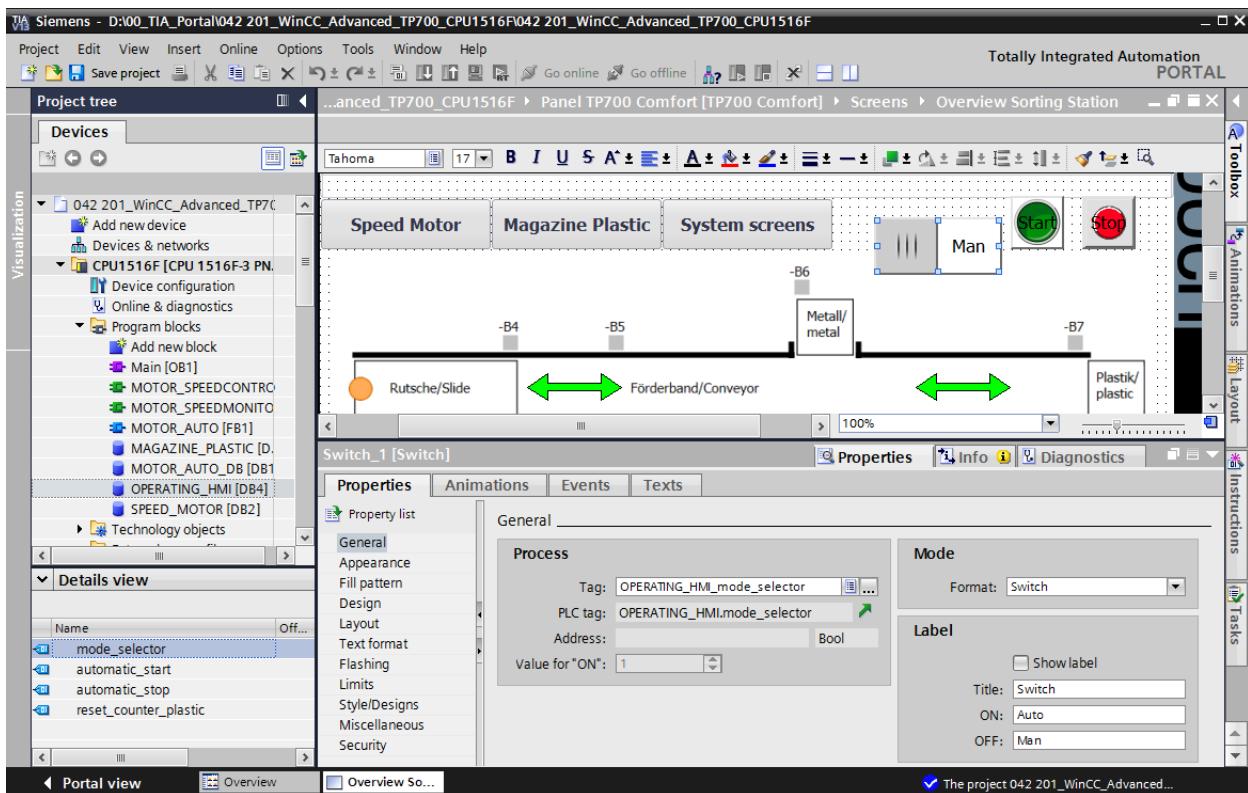
"Release" with "System function" → "SetBit". The → "Automatic_Stop" tag from the data block → "OPERATION_HMI[DB4]" is used for the process connection in both cases.



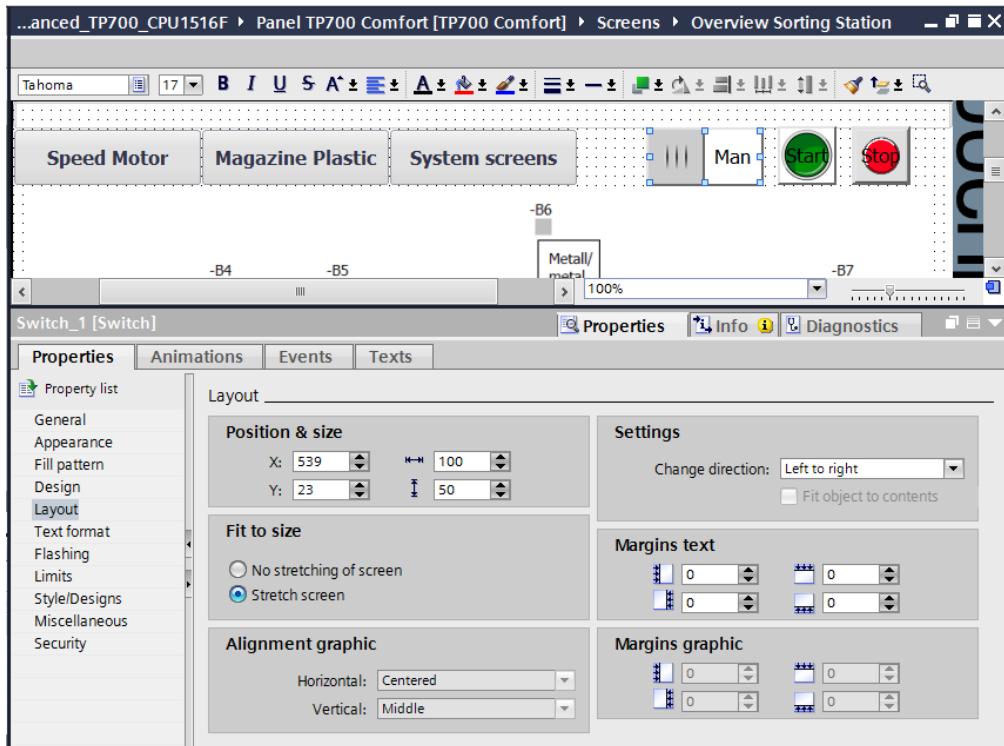
- In order to implement the mode switch, use drag-and-drop to move the → "Switch" object from → "Elements" in "Tools" to a location at the top between the buttons for the screen change and the Start button.



- Under "General" in "Properties", select the "Show label" → option. Next, enter the texts → "Auto" for the "ON" state and → "Manual" for the "OFF" state. The → "Mode selection" tag from the data block → "OPERATION_HMI[DB4]" is used for the process connection.



- Under "Layout" in "Properties", adapt the size of the mode switch under →Position & Size.



- Now compile the CPU and save the project.

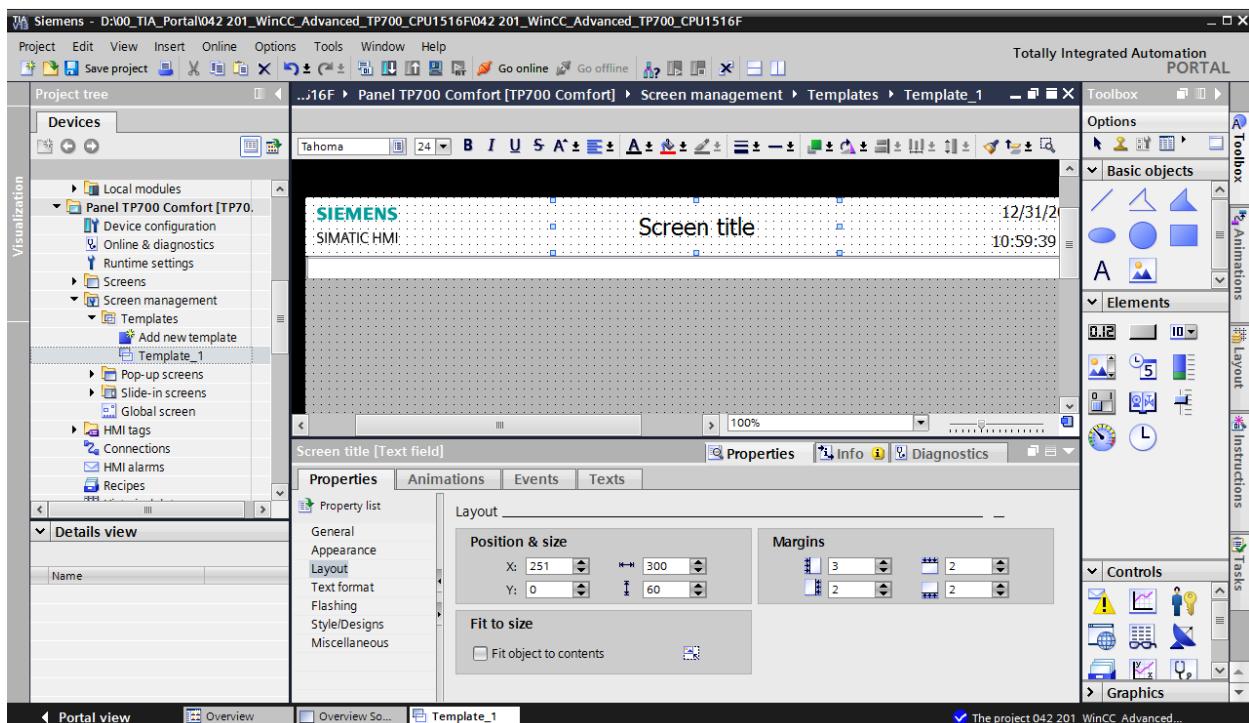
(→ Panel TP700 Comfort → → Save project)

- Afterwards, download the changed visualization to the panel.

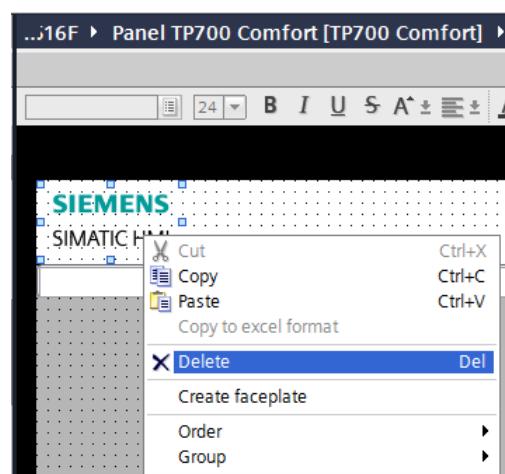
(→

7.14 Adapting the headers in the template

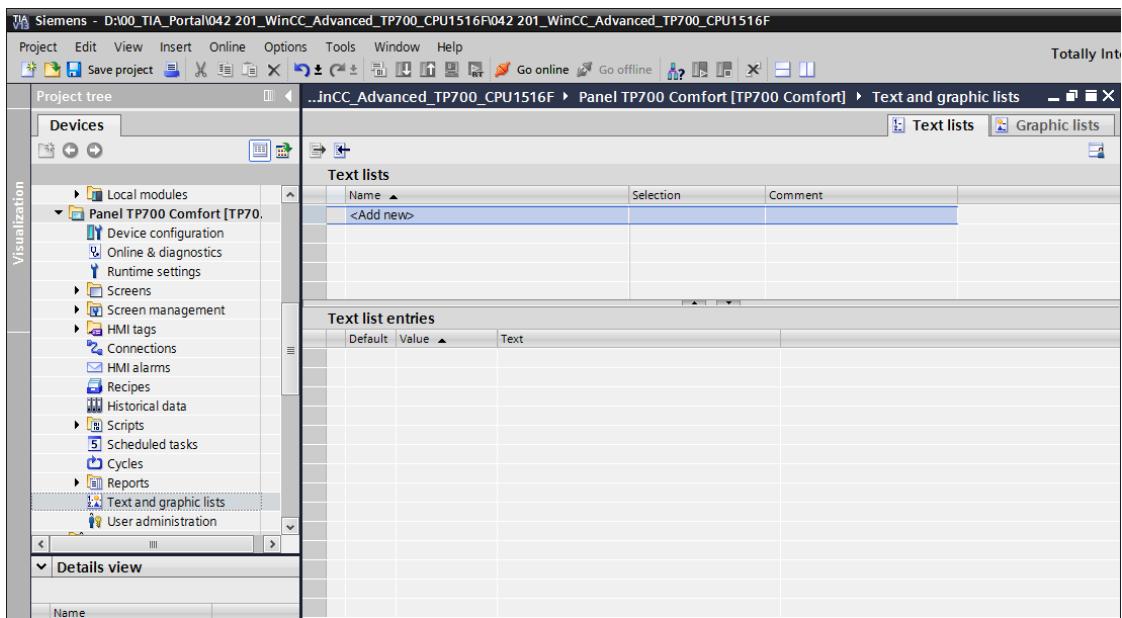
- The plant states are to be shown overall in the header. "Template_1" for our header and footer was already created by the wizard when the panel was created. The footer contains the system buttons and a logo, date and time as well as the "Screen title" text field was previously created in the header.
- You would now like to adapt the screen title to the dimensions specified here under "Layout" in "Properties" in → "Position & Size".



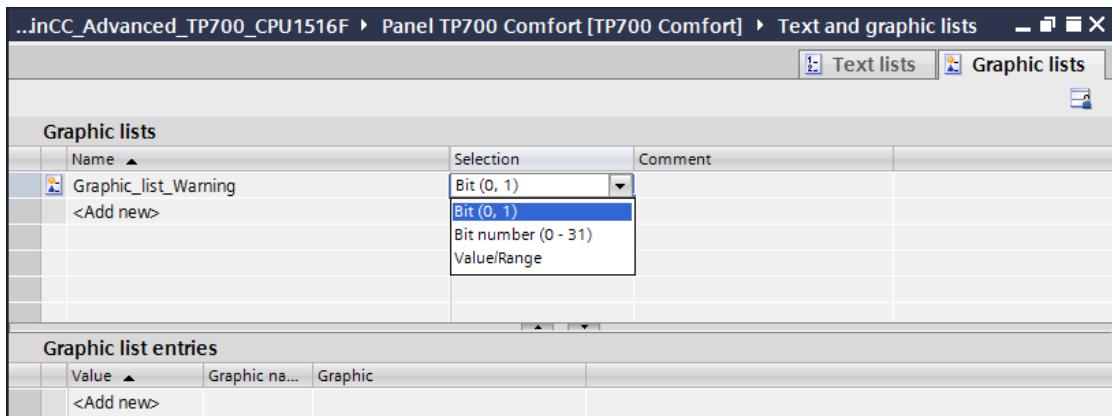
- Delete the logo on the left in the header by selecting the → graphic display for the LOGO with the right mouse button and clicking → "Delete".



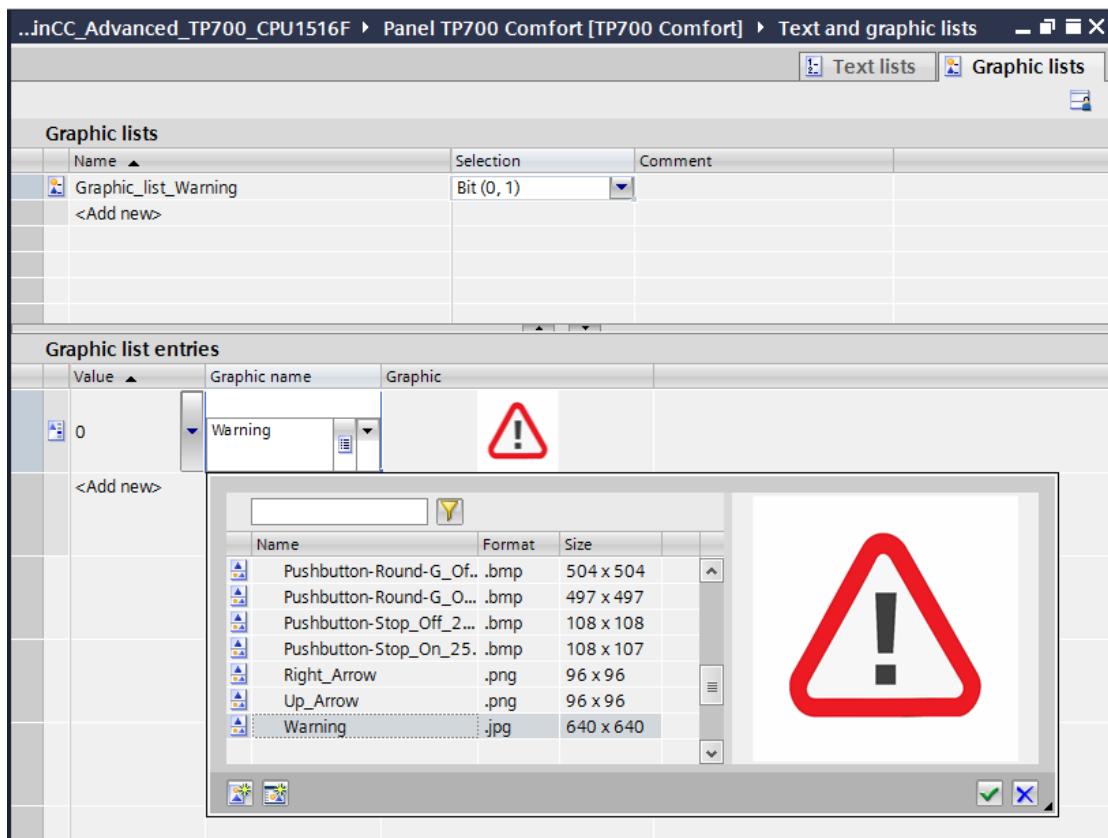
- In the "Panel TP700 Comfort", open the → "Text and Graphic lists" folder.



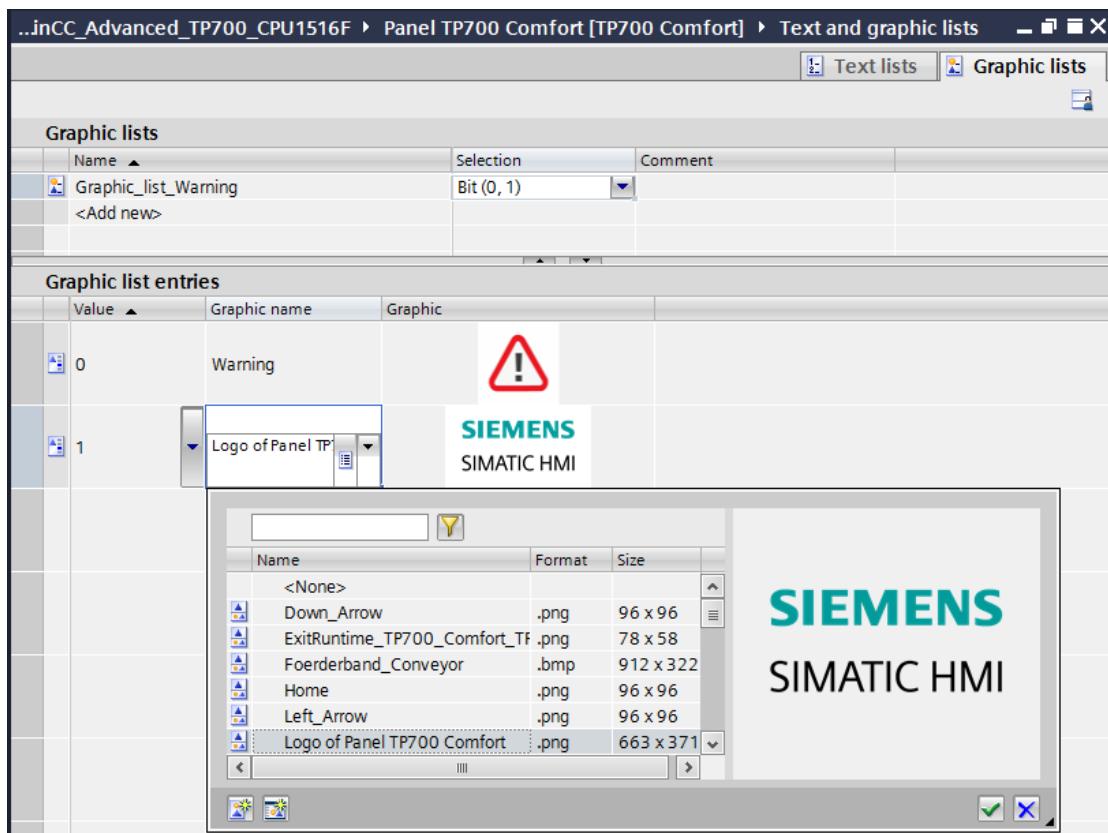
- Under "Graphic lists", create a → "Graphic_List_Warning" with → selection "Bit(0,1)".



- Click the icon for "Value 0" to open the selection dialog for the graphics stored under "Project graphics" in the "Language & Resources" path. Click the icon for "Create graphic from file" and select the "Warning.bmp" file from the "SCE_EN_042-201_Screens" folder by double-clicking it. This file is stored in the "Languages & Resources" path under "Project graphics".



- The graphic that you want to assign to "Value 1" is already stored in the "Languages & Resources" path under "Project graphics". After clicking the → icon, you can select the → "Logo of Panel TP700 Comfort" file directly.



- Now change to the "Text lists" and create the following three text lists: → "Text list_Emergency stop" → "Text list_Main switch" and → "Text list_Automatic". Use the → selection "Bit(0,1)" in each case.

Name	Selection	Comment
Text_list_main_switch	Bit (0,1)	Display status main switch
Text_list_emergency_stop	Bit (0,1)	Display status emergency stop
Text_list_automatic	Bit (0,1)	Display automatic start/stop

- Make the following assignments in "Text list_Emergency stop": "Value 0" → "EMERGENCY STOP triggered" and → "Value 1" → "EMERGENCY STOP OK".

Name	Selection	Comment
Text_list_main_switch	Bit (0,1)	Display status main switch
Text_list_emergency_stop	Bit (0,1)	Display status emergency stop
Text_list_automatic	Bit (0,1)	Display automatic start/stop

Value	Text
0	emergency stop released
1	emergency stop OK

- Make the desired assignments in "Text list_Main switch": "Value 0" → "Main switch OFF" and → "Value 1" → "Main switch ON".

Name	Selection	Comment
Text_list_main_switch	Bit (0,1)	Display status main switch
Text_list_emergency_stop	Bit (0,1)	Display status emergency stop
Text_list_automatic	Bit (0,1)	Display automatic start/stop

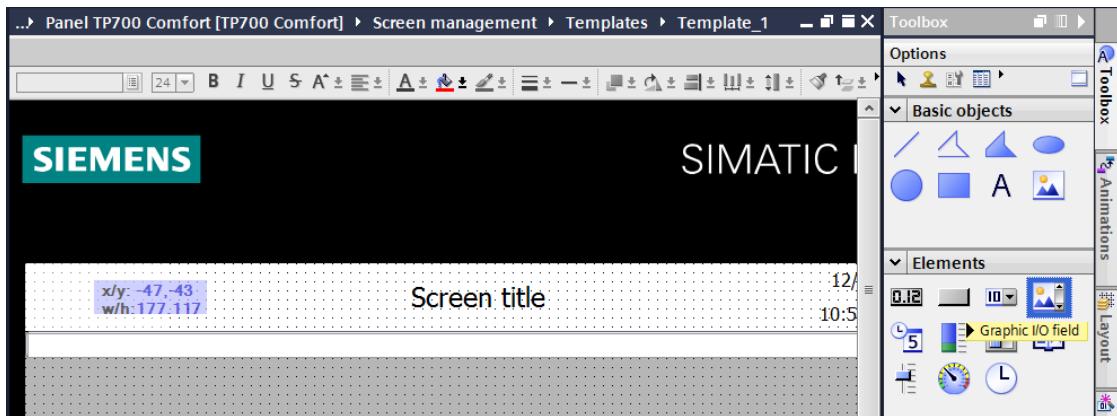
Value	Text
0	main switch OFF
1	main switch ON

- Make the following assignments in "Text list_Automatic": "Value 0" → "Automatic stopped" and → "Value 1" → "Automatic started".

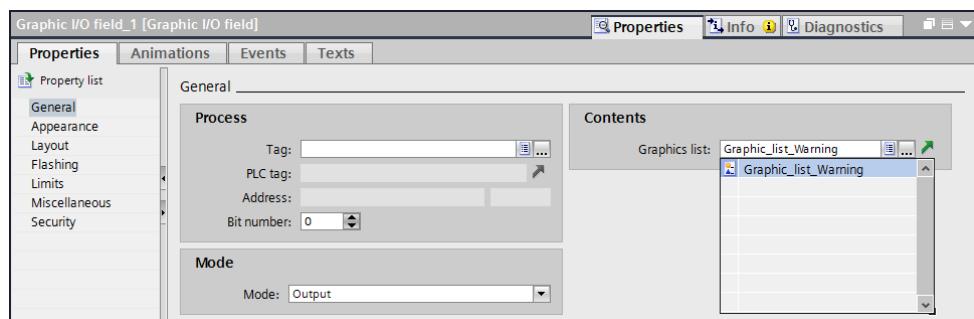
Name	Selection	Comment
Text_list_main_switch	Bit (0,1)	Display status main switch
Text_list_emergency_stop	Bit (0,1)	Display status emergency stop
Text_list_automatic	Bit (0,1)	Display automatic start/stop

Value	Text
0	automatic stopped
1	automatic started

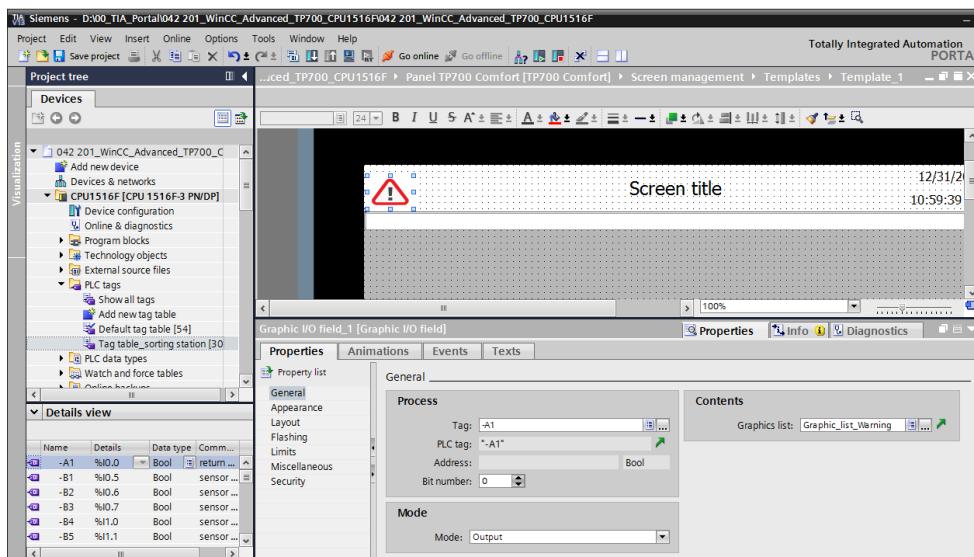
- Back in "Template_1" for our header, use drag-and-drop to move the → "Graphic IO field" object  from → "Elements" in Tools to the top left corner.



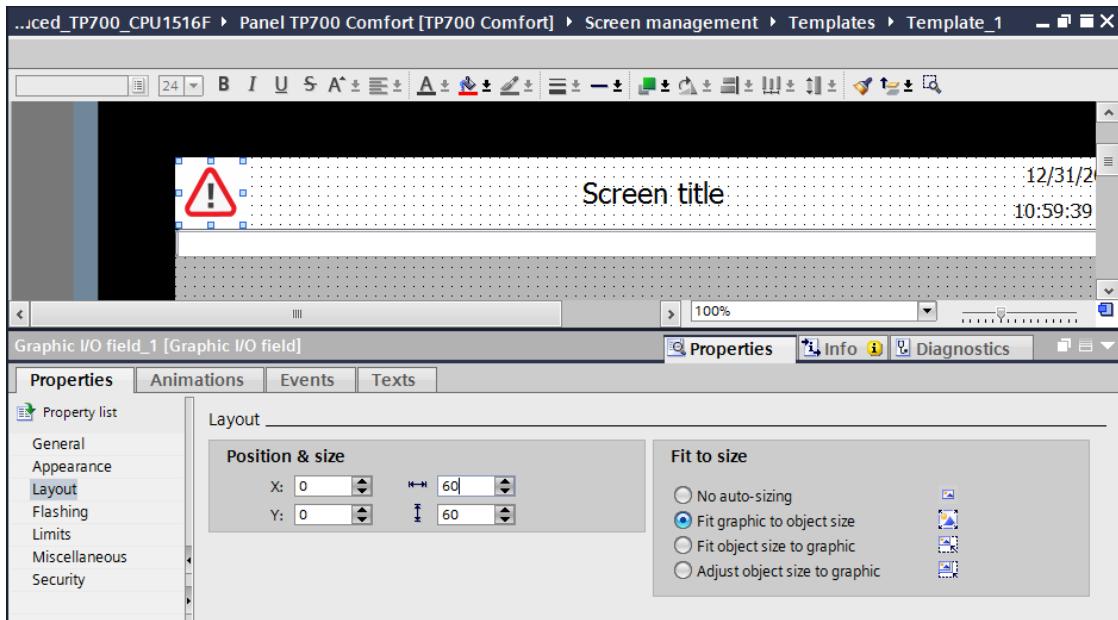
- Under "General" in "Properties", change the "Mode" to → "Output".
Then open the selection dialog for → "Graphic list" by clicking the  icon. Select the "Graphic list_Warning" you just created.



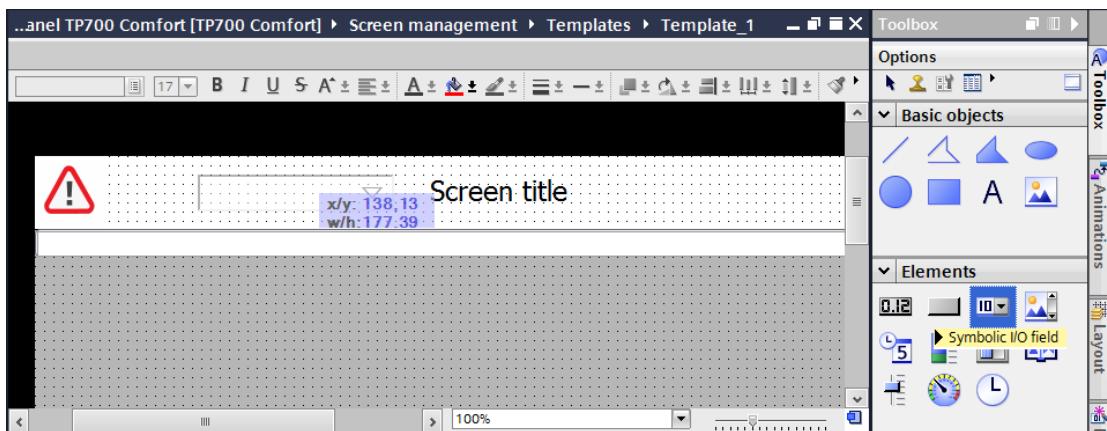
- To establish the connection to the global tag in the CPU, select → "PLC tags" under → "CPU_1516F" and below that → "Tag_table_sorting_station". Now drag the → "-A1" tag from the "Detail view" into the "Tag" field In addition, select → "Bit number 0" as the type of evaluation.



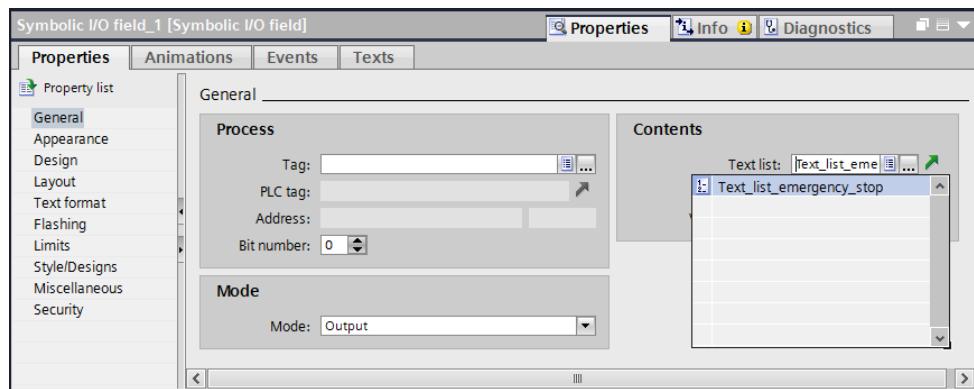
- Under "Layout" in "Properties", adapt the size of the "Graphic IO field" under → "Position & Size".



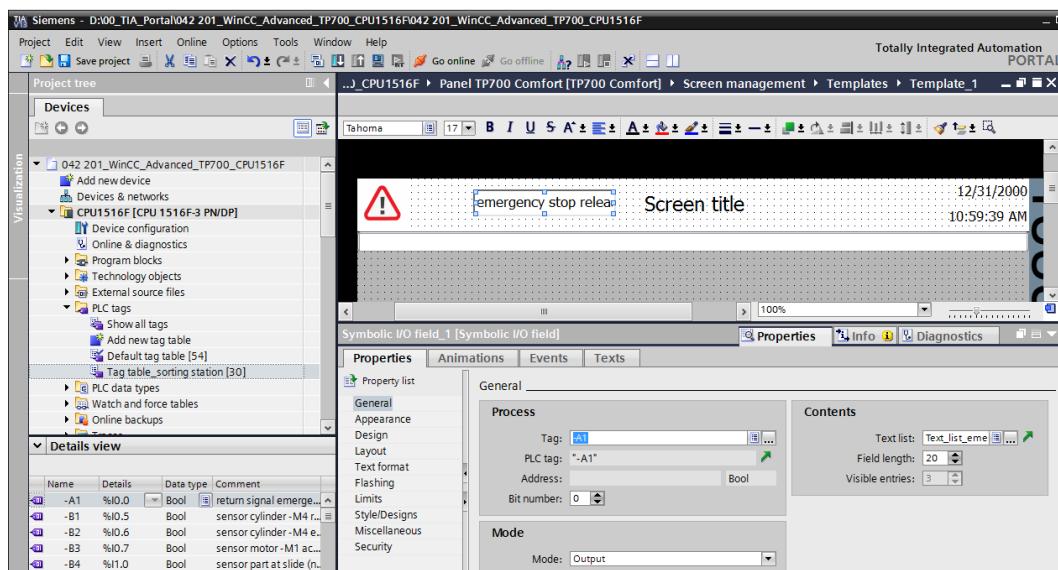
- To display the status of the EMERGENCY STOP in the header, use drag-and-drop to move the → "Symbolic IO field" object from → "Elements" in Tools to the right of the "Graphic I/O field".



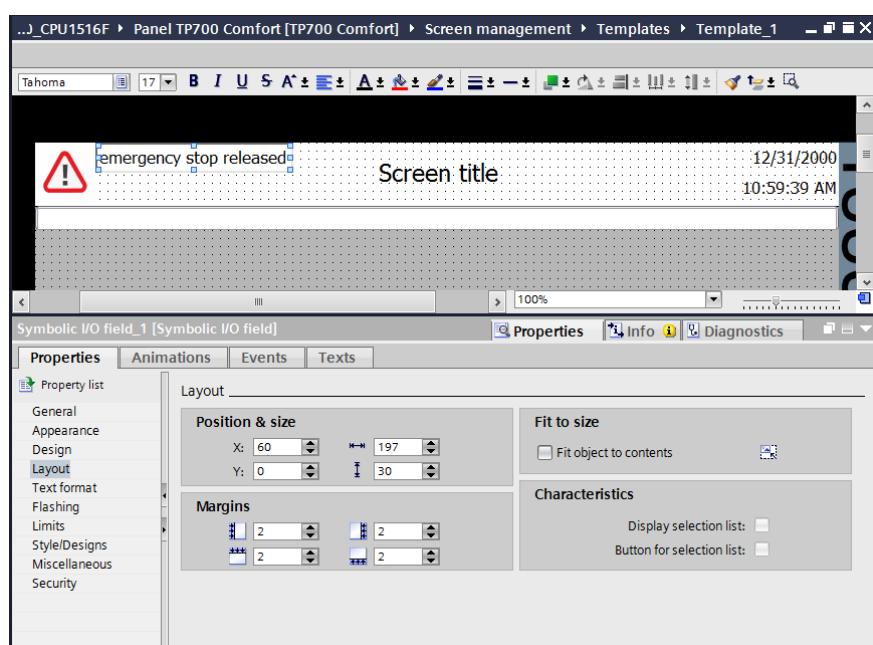
- Under "General" in "Properties", change the "Mode" to → "Output". Next, open the selection dialog for → "Text list" by clicking the icon. Select the "Text list_Emergency stop" you just created.



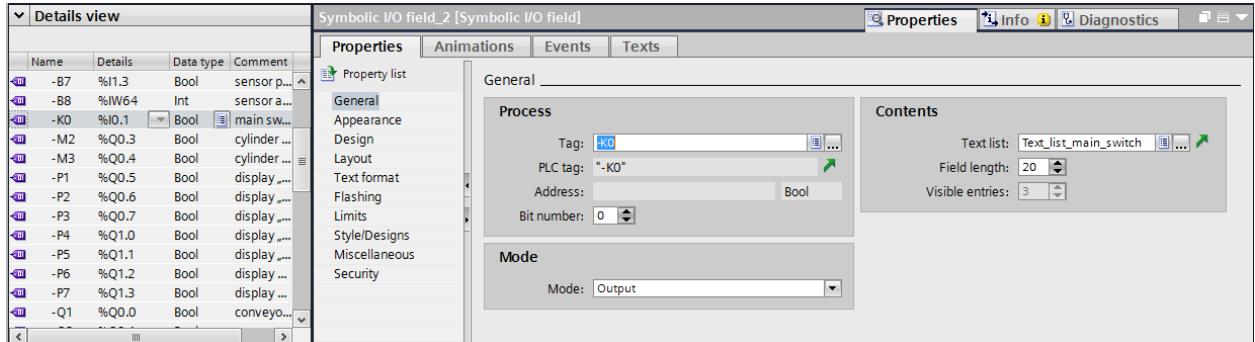
- To establish the connection to the global tag in the CPU, select → "PLC tags" under → "CPU_1516F" and below that → "Tag_table_sorting_station". Now drag the → "-A1" tag from the "Detail view" into the "Tag" field and also select → "Bit number 0"



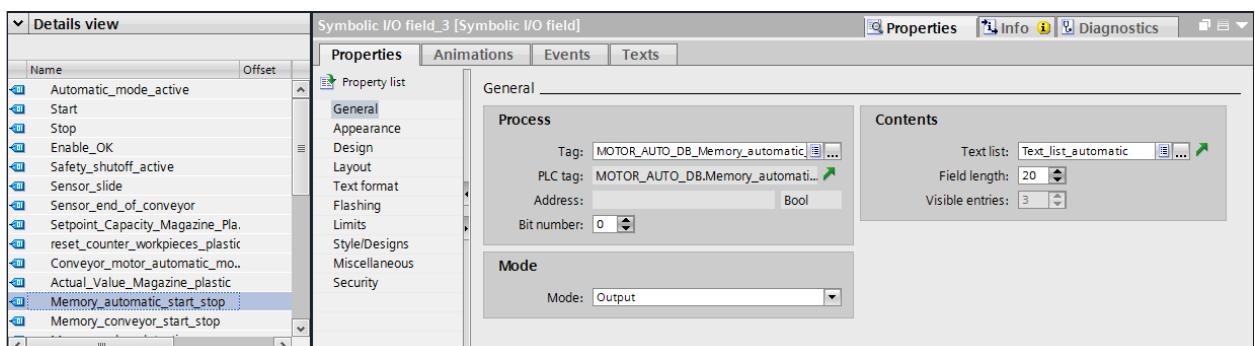
- Under "Layout" in "Properties", adapt the size of the "Graphic IO field" under → "Position & Size".



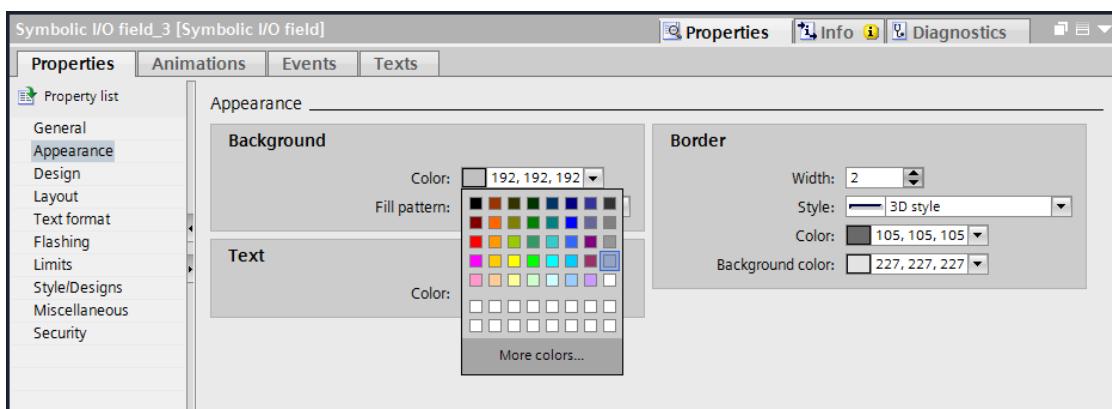
- Repeat the preceding steps again for the text lists → "Text list_Main switch" and → "Text list_Automatic" to insert these left of the date and time directly one beneath the other.
- The "Text list_Main switch" is connected using the → "-K0" tag from Tag_table_sorting_station".



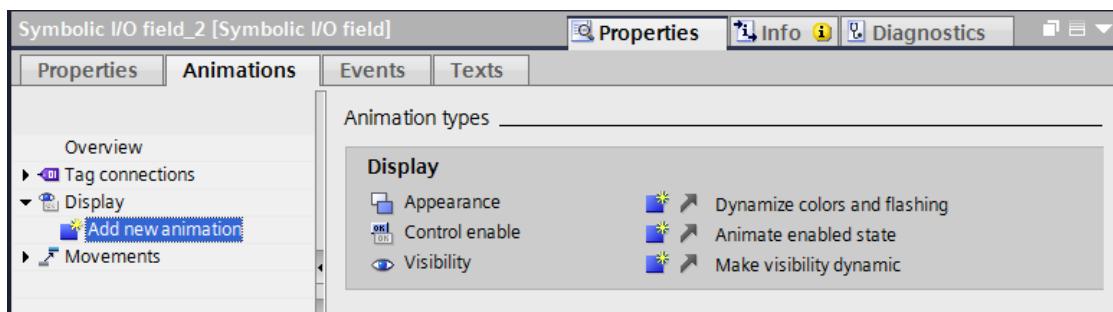
- The "Text list_Automatic" is connected using the → "-Memory_Automatic_Start_Stop" tag from "MOTOR_AUTO_DB1[DB1]".



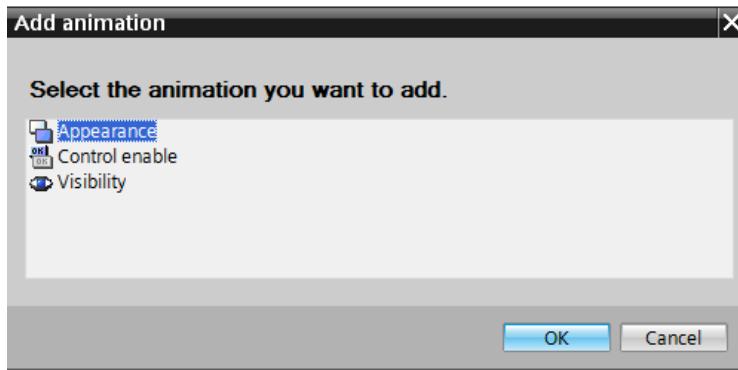
- Under "Appearance" in "Properties", change the "Color" of the "Background" for → "Text list_Main switch" and → "Text list_Automatic" to → "Gray".



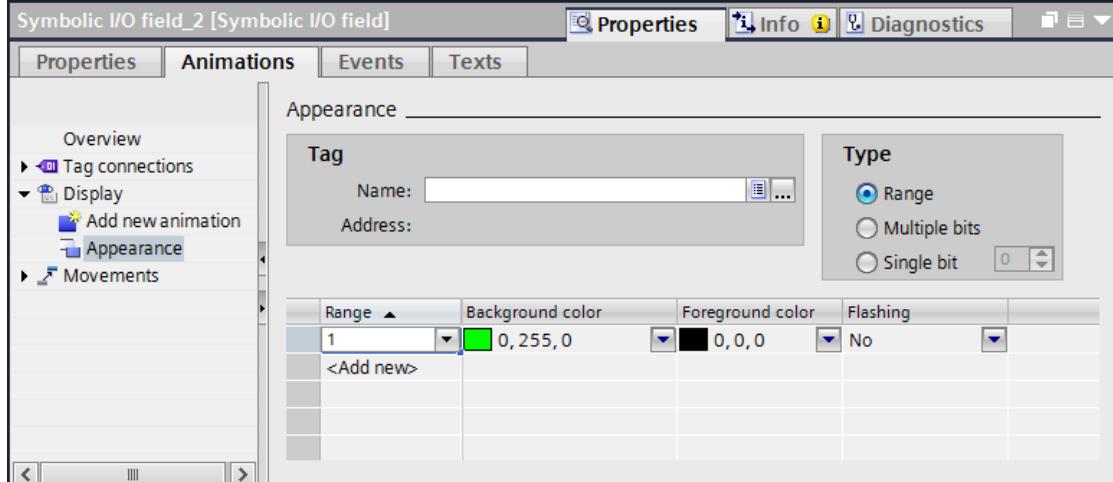
- For → "Text list_Main switch" and → "Text list_Automatic", change to the "Animation" tab and select "Display" and click → "Add new animation"



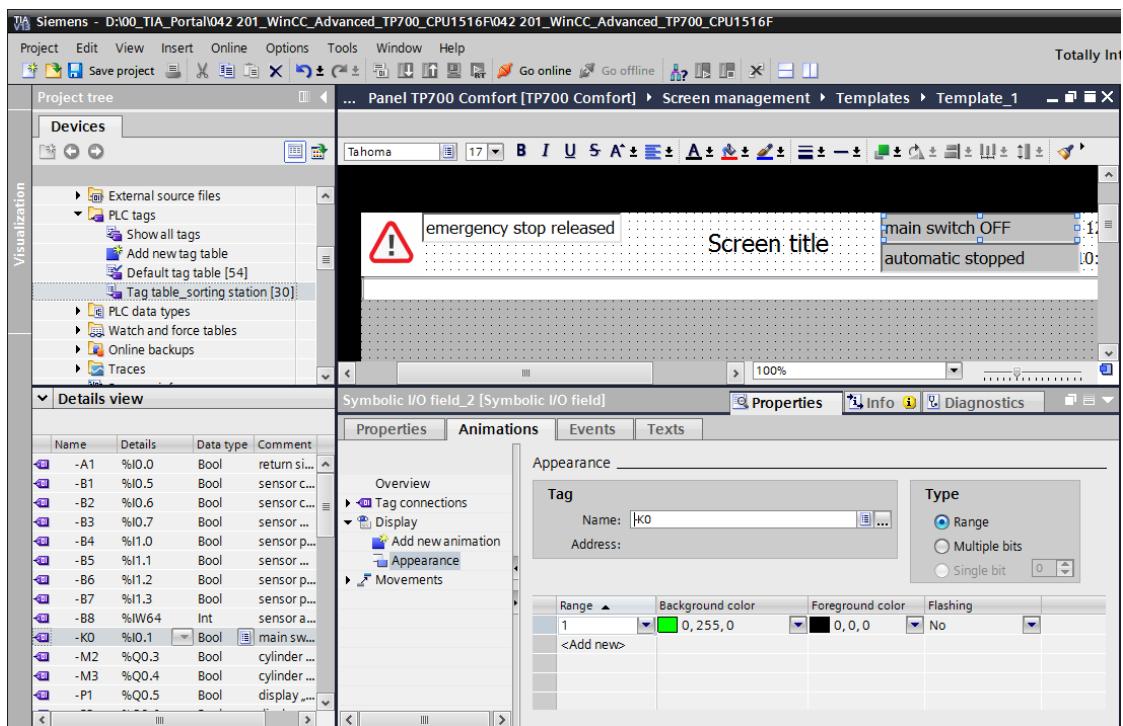
→ In the displayed dialog, select → "Appearance" and click → "OK".



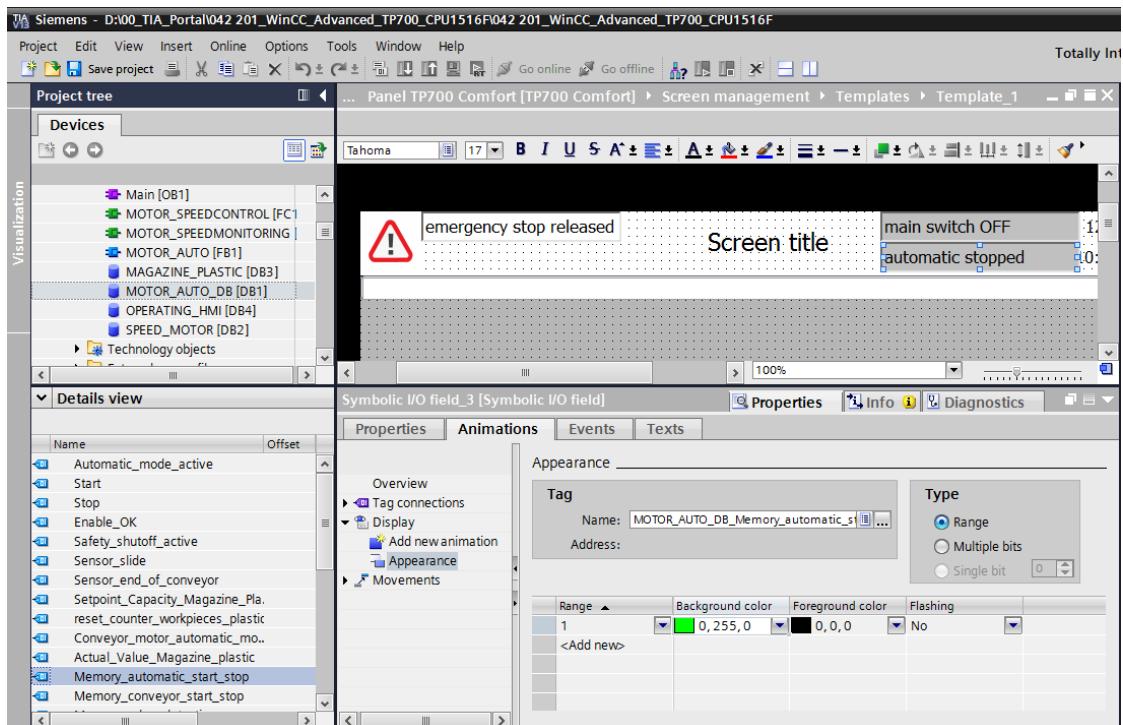
→ In "Appearance" of both "Symbolic IO fields", add an area with the value → "1" (signal state "High") and change the "Background color" to → "Green".



→ The "Text list_Main switch" is connected using the → "-K0" tag from Tag_table_sorting_station".



- The "Text list_Automatic" is connected using the
- "-Memory_Automatic_Start_Stop" tag from "MOTOR_AUTO_DB1[DB1]".



- In the default tag table, the "Acquisition cycle" of all tags is also to be accelerated from 1 second to 100 milliseconds here.

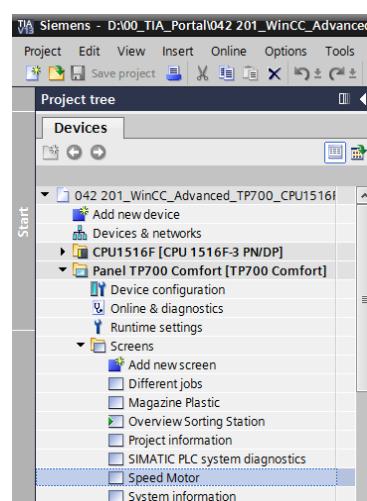
Name	Data type	Connection	PLC name	PLC tag	Address	Access mode	Acquisition cycle	Logged	Source ...
-A1	Bool	HMI_Connectio...	CPU1516F	"-A1"		<symbolic access>	1 s	<input type="checkbox"/>	return s...
-B1	Bool	HMI_Connectio...	CPU1516F	"-B1"		<symbolic access>	100 ms	<input type="checkbox"/>	sensor ...
-B2	Bool	HMI_Connectio...	CPU1516F	"-B2"		<symbolic access>	100 ms	<input type="checkbox"/>	sensor ...
-B3	Bool	HMI_Connectio...	CPU1516F	"-B3"		<symbolic access>	100 ms	<input type="checkbox"/>	sensor ...
-B4	Bool	HMI_Connectio...	CPU1516F	"-B4"		<symbolic access>	100 ms	<input type="checkbox"/>	sensor ...
-B5	Bool	HMI_Connectio...	CPU1516F	"-B5"		<symbolic access>	100 ms	<input type="checkbox"/>	sensor ...
-B6	Bool	HMI_Connectio...	CPU1516F	"-B6"		<symbolic access>	100 ms	<input type="checkbox"/>	sensor ...
-B7	Bool	HMI_Connectio...	CPU1516F	"-B7"		<symbolic access>	100 ms	<input type="checkbox"/>	sensor ...
-K0	Bool	HMI_Connectio...	CPU1516F	"-K0"		<symbolic access>	100 ms	<input type="checkbox"/>	main s...
MAGAZINE_PLASTIC_Plastic_Parts_Actual	Int	HMI_Connectio...	CPU1516F	MAGAZINE_PLASTIC.Plast		<symbolic access>	100 ms	<input type="checkbox"/>	Actual ...
MOTOR_AUTO_DB_Memory_automatic_start_stop	Bool	HMI_Connectio...	CPU1516F	MOTOR_AUTO_DB.Mem...		<symbolic access>	100 ms	<input type="checkbox"/>	Memor...
OPERATING_HMI_automatic_start	Bool	HMI_Connectio...	CPU1516F	OPERATING_HMI.automa...		<symbolic access>	100 ms	<input type="checkbox"/>	HMI pu...
OPERATING_HMI_automatic_stop	Bool	HMI_Connectio...	CPU1516F	OPERATING_HMI.automa...		<symbolic access>	100 ms	<input type="checkbox"/>	HMI pu...
OPERATING_HMI_mode_selector	Bool	HMI_Connectio...	CPU1516F	OPERATING_HMI.mode_s...		<symbolic access>	100 ms	<input type="checkbox"/>	HMI mo...
OPERATING_HMI_reset_counter_plastic	Bool	HMI_Connectio...	CPU1516F	OPERATING_HMI.reset_c...		<symbolic access>	100 ms	<input type="checkbox"/>	HMI res...
-Q3	Bool	HMI_Connectio...	CPU1516F	"-Q3"		<symbolic access>	100 ms	<input type="checkbox"/>	convey...
SPEED_MOTOR_Speed_Actual_Value	Real	HMI_Conne...	CPU1516F	SPEED_MOTOR.Speed...		<symbolic acce...	100 ms	<input type="checkbox"/>	Speed a...
<Add new>									

- Before the project is downloaded to the CPU and the panel, compile the CPU and panel again and save the project. (→ CPU_1516F → → Panel TP700 Comfort → →

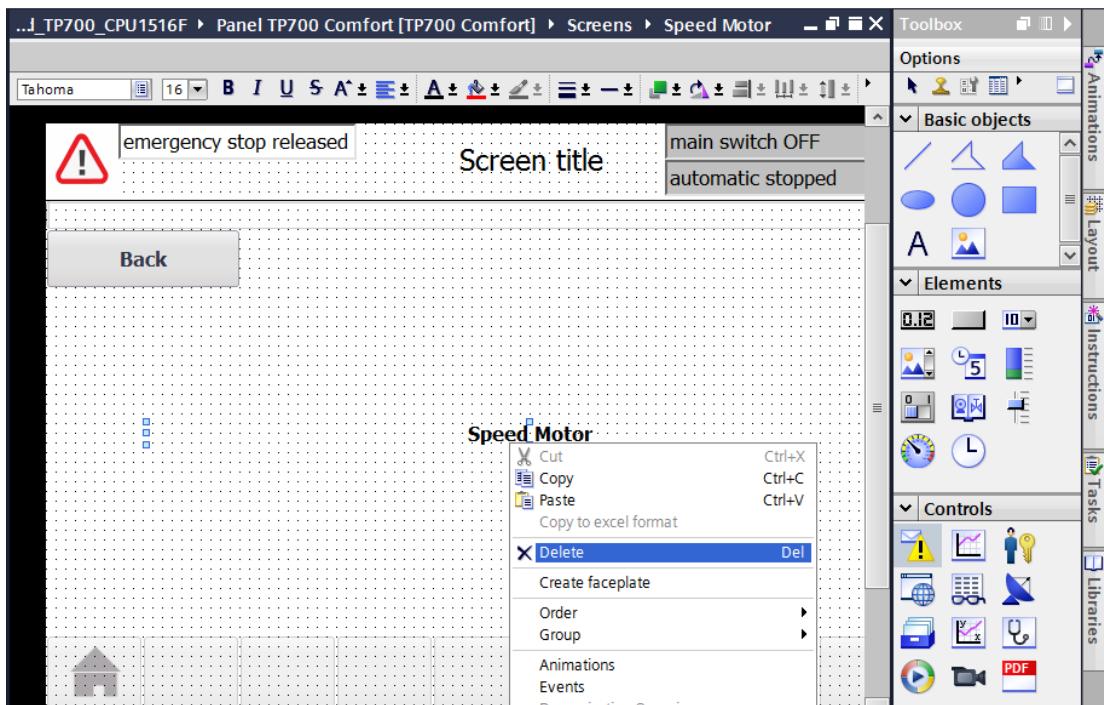
- After successful compilation, the complete controller with the created program including the hardware configuration can be downloaded, as described in the previous modules.
- To download the visualization to the panel, proceed in a similar way. Select the → "Panel TP700 Comfort [TP700 Comfort]" folder and click the icon
 → "Download to device".

7.15 Bar graph

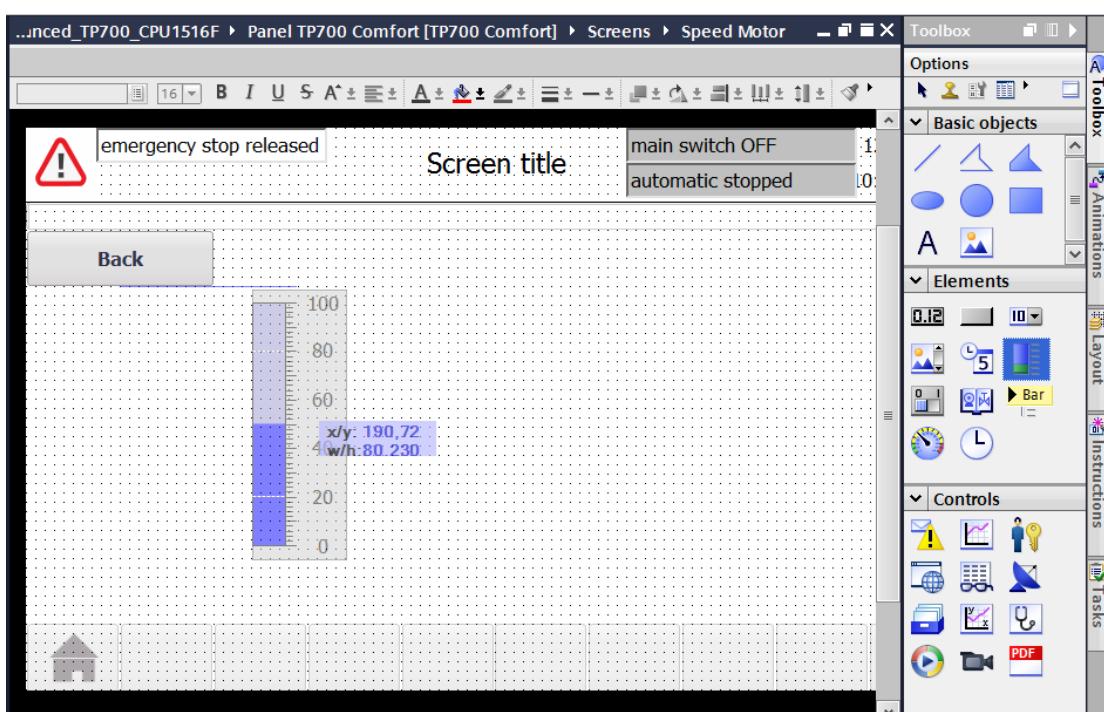
- You would like to specify the setpoint for the speed control of the motor and display the actual value. To do this, open the → "SPEED MOTOR" screen by double-clicking it.



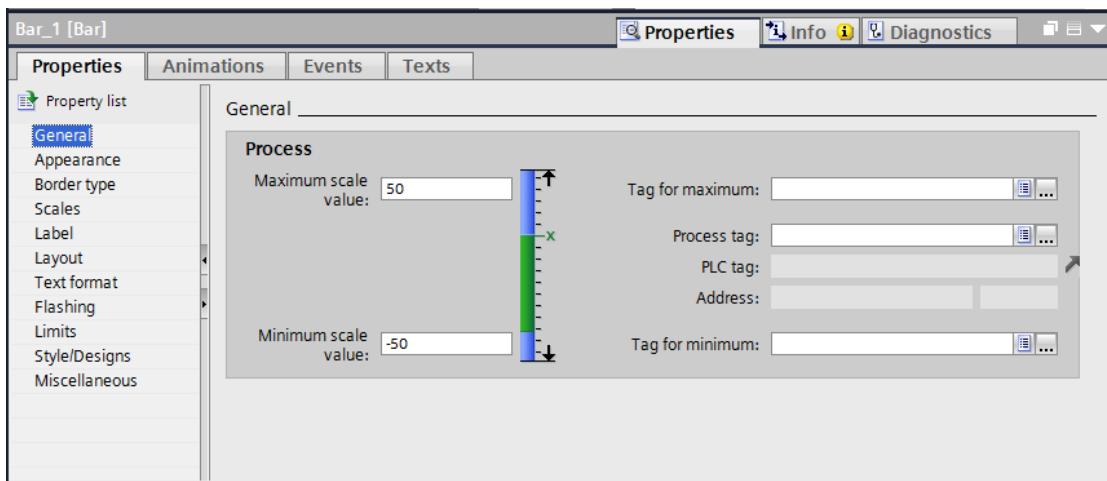
- The text field in the center of the screen is to be removed by right-clicking it and selecting → "Delete" in the displayed dialog.



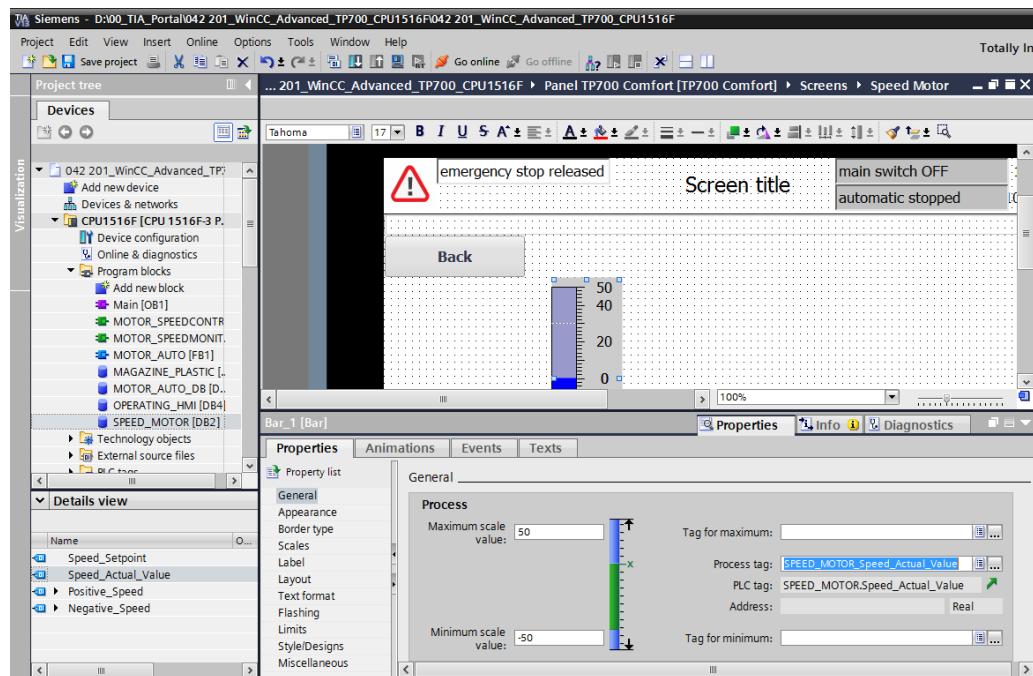
- In order to graphically display the actual speed value, use drag-and-drop to move the → "Bar" object from → "Elements" in "Tools" to the center of the screen.



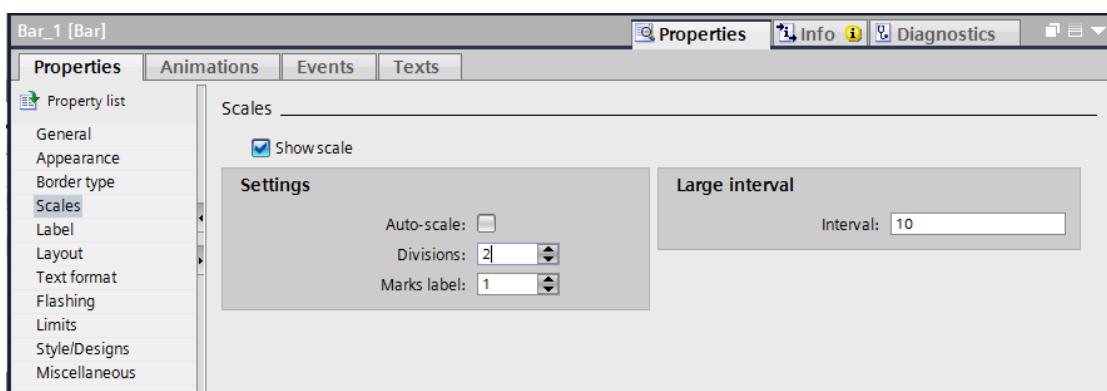
- Under "General" in "Properties", change the "Maximum scale value" to → 50 and the "Minimum scale value" to → -50.



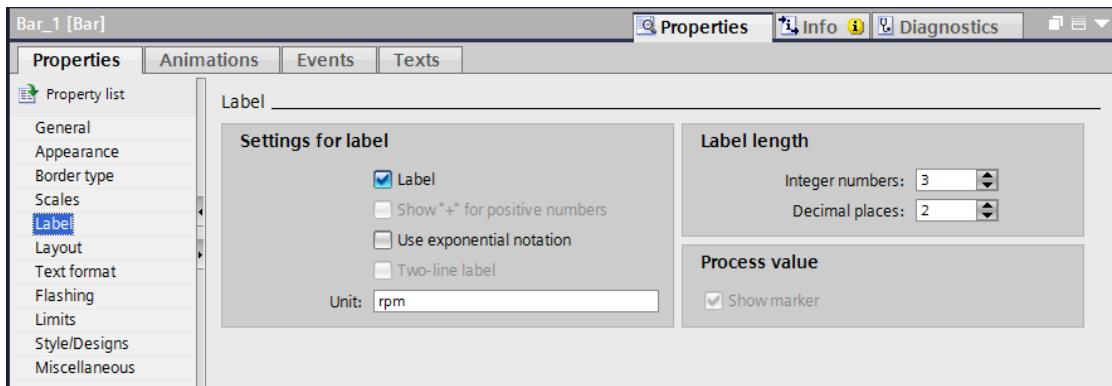
- For the process connection, select the → "Program blocks" of the → "CPU_1516F" and the → "SPEED_MOTOR[DB2]" data block there. Then drag the → "Actual speed value" tag from the → "Detail view" into the field for "Process tag".



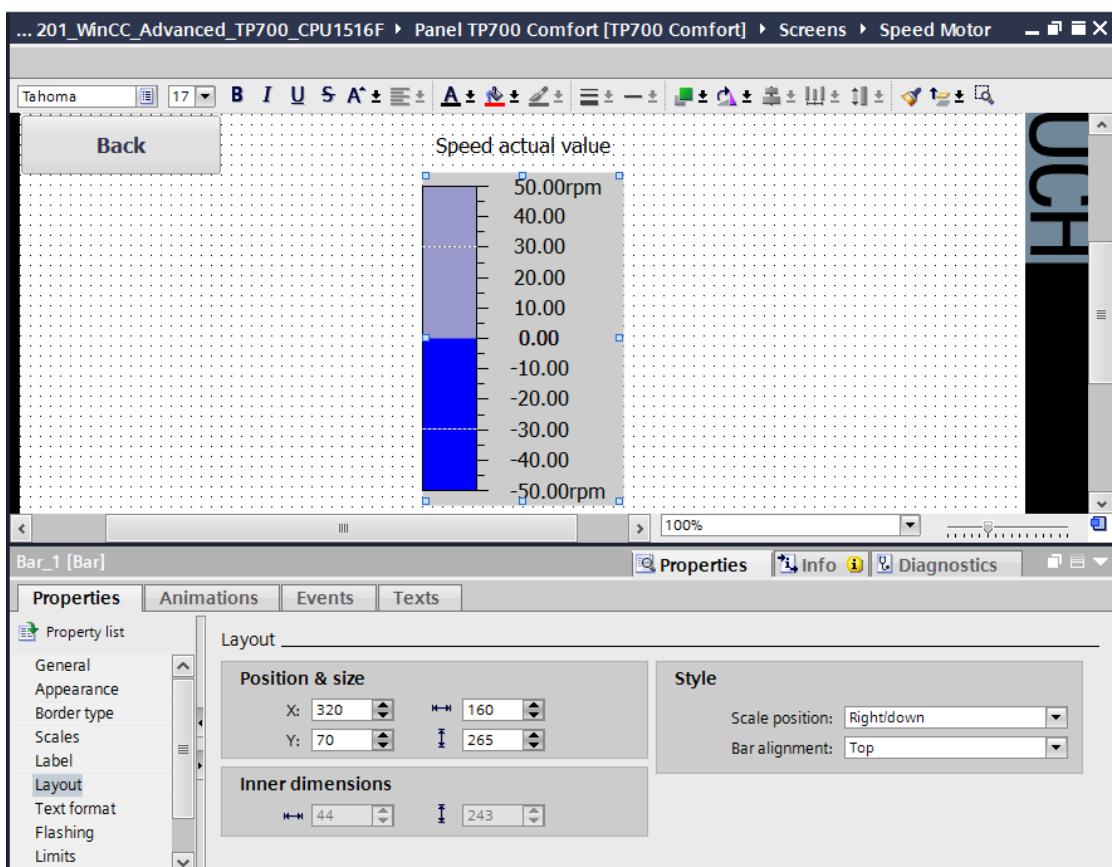
- Under "Scales" in "Properties", select → "Show scale". Under "Divisions" select → 2. Under "Marks label" select → 1. Under "Interval", select → 10.



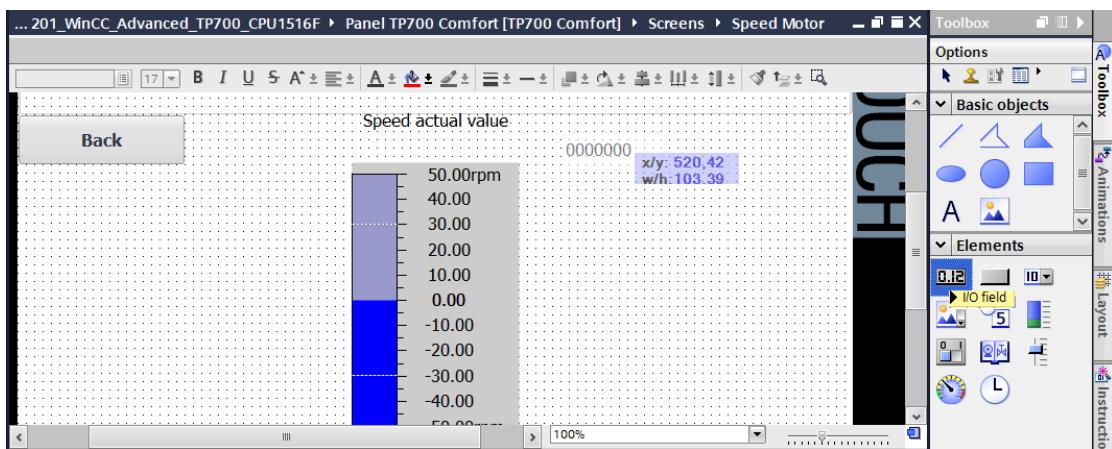
- Under "Label" in "Properties", select → "Label". Under "Unit" select → rpm. Under "Decimal places", select → 2.



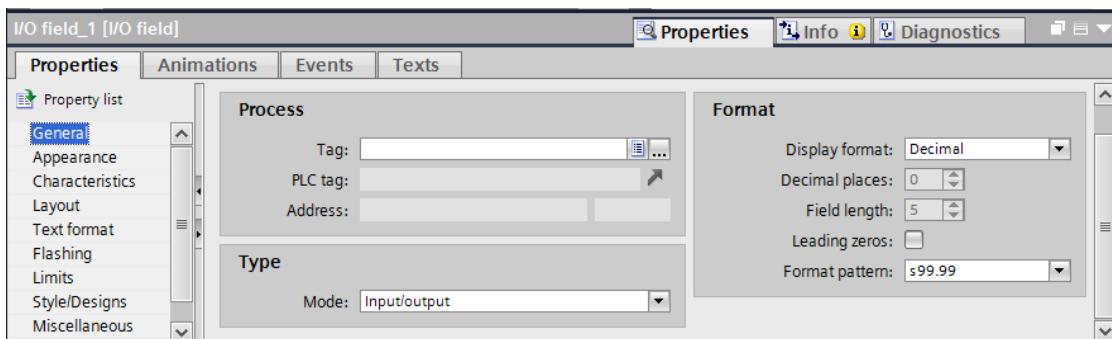
- Under "Layout" in "Properties", adapt the position and size of the bar under → "Position & Size". Above the bar diagram, insert a → "Text field" **A** for the description with text → "Actual speed value".



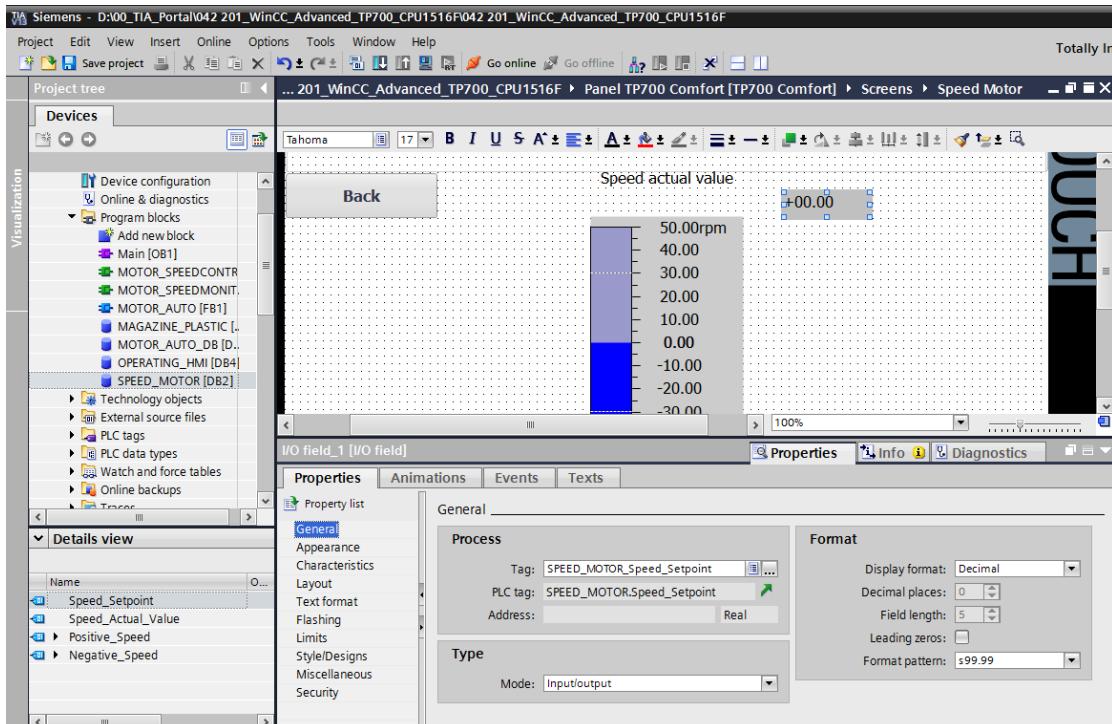
- In order to specify the speed setpoint, move the → "IO field" object **B** from → "Elements" in "Tools" to a location above and to the right of the bar display.



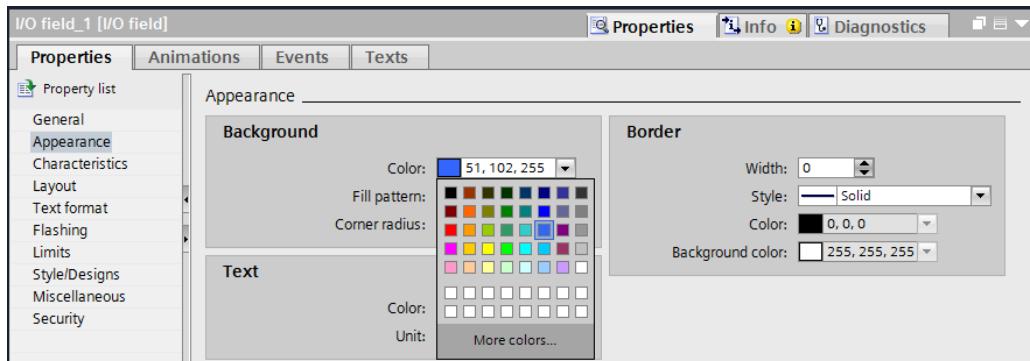
- Under "General" in "Properties" keep the "Type" as →"Input/Output" and change the "Format pattern" to → s99.99.



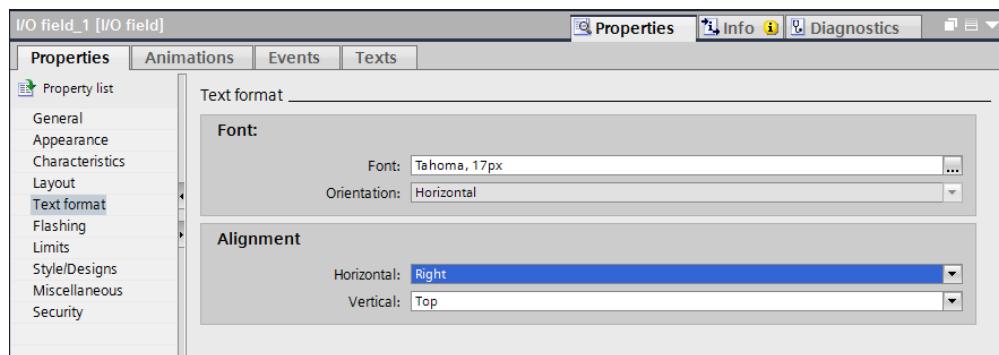
- For the process connection, select the → "Program blocks" of the → "CPU_1516F" and the → "SPEED_MOTOR[DB2]" data block there.
Now drag the → "Speed setpoint" tag from the → "Detail view" into the field for "Tag".



- Under "Appearance" in "Properties", change the "Color" of the "Background" to → "Blue".

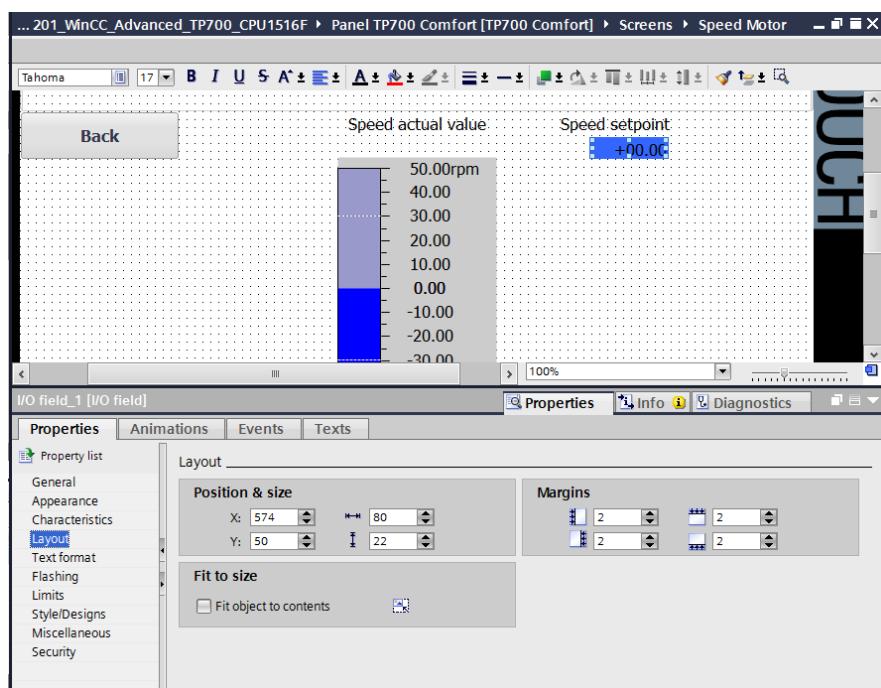


- Under "Text format" in "Properties", change the "Horizontal" setting of "Alignment" to → "Right".



- Under "Layout" in "Properties", adapt the position and size of the IO field under → "Position & Size".

- Above the bar diagram, insert a → "Text field" A for the description with text → "Speed setpoint".



- In the default tag table, the "Acquisition cycle" of the newly created "SPEED_MOTOR_speed setpoint" tag is again changed from 1 second to 100 milliseconds.
- Before the visualization is downloaded to the panel, compile the panel again and save the project.

(→ Panel TP700 Comfort → → Save project)

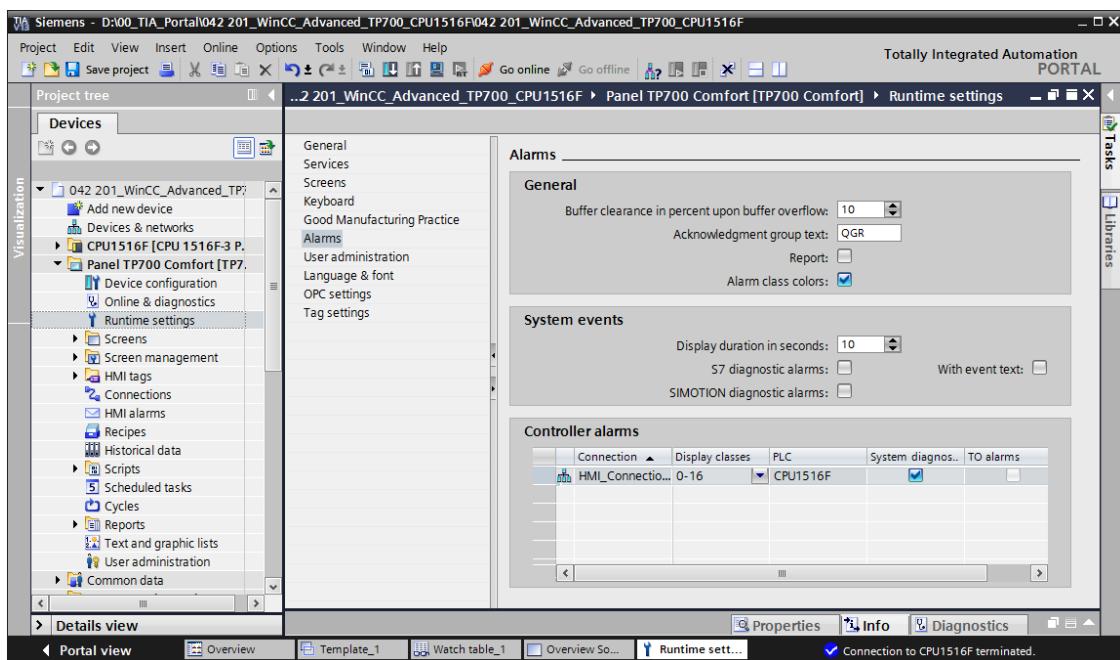
- Select the → "Panel TP700 Comfort [TP700 Comfort]" folder and click the icon → "Download to device".

7.16 Alarms

When you created the TP700 Comfort Panel using the wizard, you created a pair of alarm windows at the same time. You should now look these over in more detail.

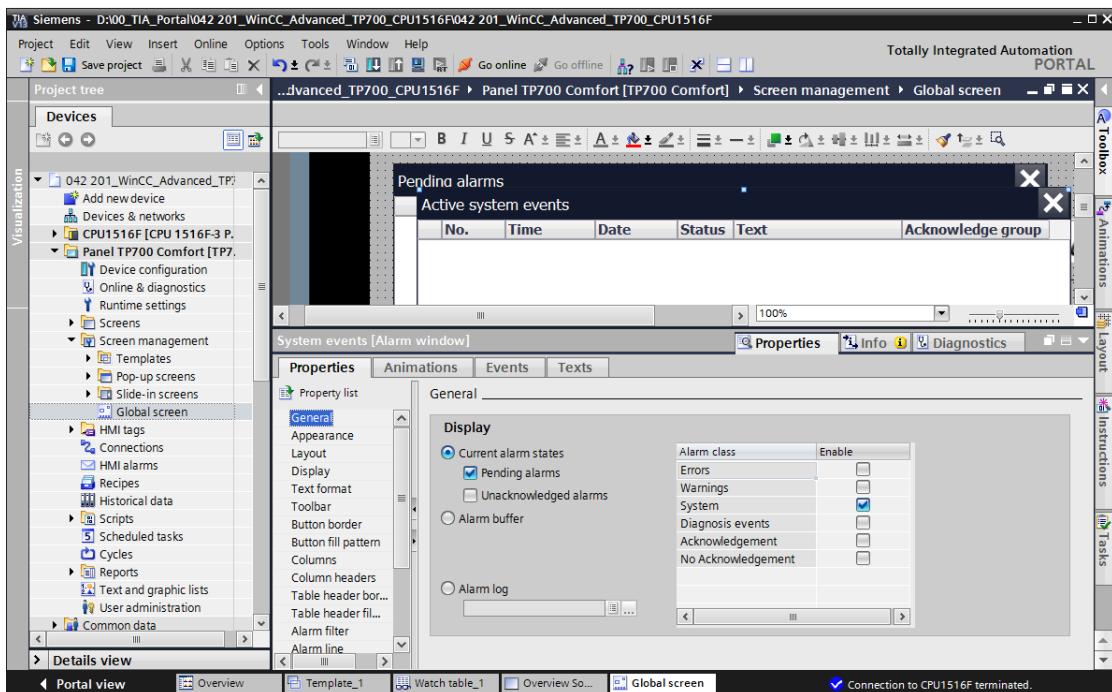
7.16.1 General alarm settings

- You first want to make a pair of settings for the display of alarms in Runtime. To do this, double-click the → "Runtime settings" folder in → "Panel TP700 Comfort" to select it. Under "General" in "Alarms", select → "Colors of alarm classes". Under "System events", change → display duration in seconds to "10". Under "Controller alarms", check whether the "System diagnostics" check box is selected .

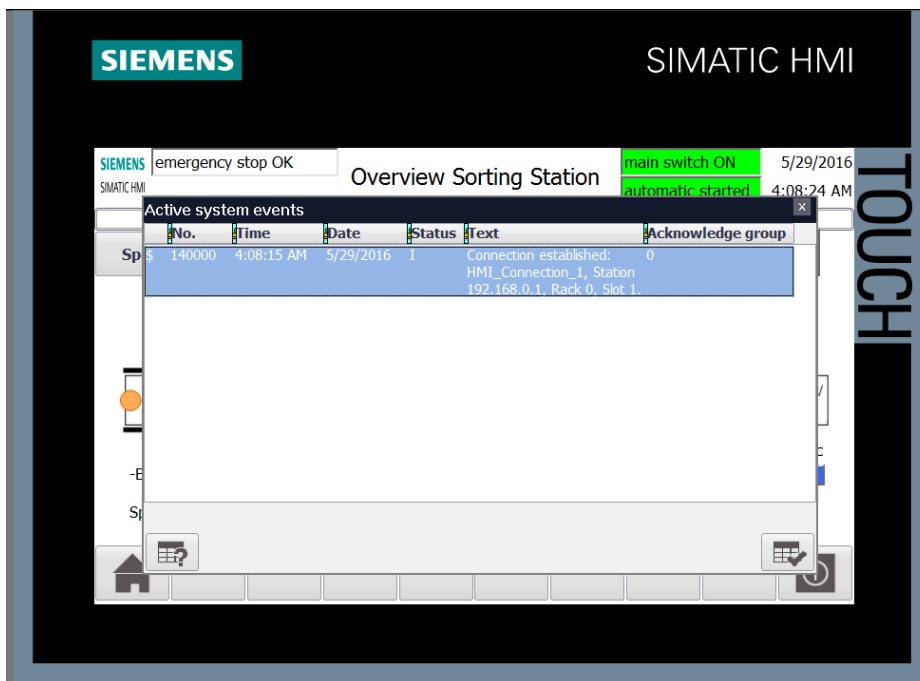


7.16.2 Alarm window

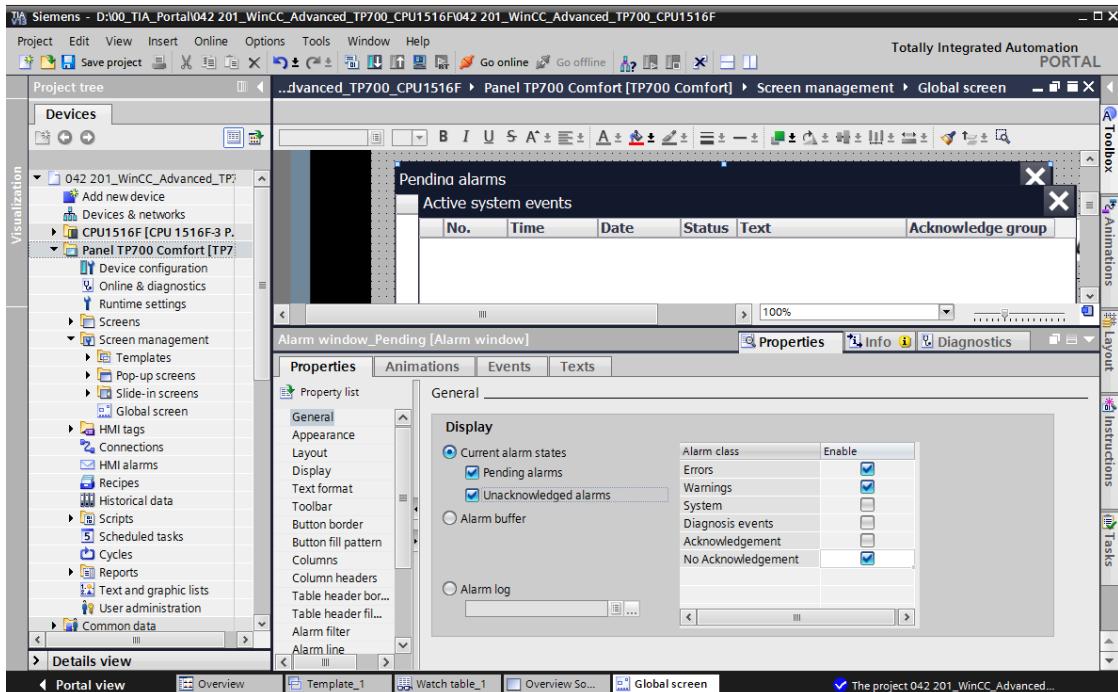
- In order for the alarm window to be shown in the foreground in every screen, there is a → "Global screen" in the → "Screen management" folder in → "Panel TP700 Comfort".
- Open this with a double-click. An alarm window → "System events" has already been created in this screen. Under "General" in "Properties", "Pending alarms" of "System" alarm class are already activated.



- System events will thus automatically be displayed for ten seconds in Runtime.

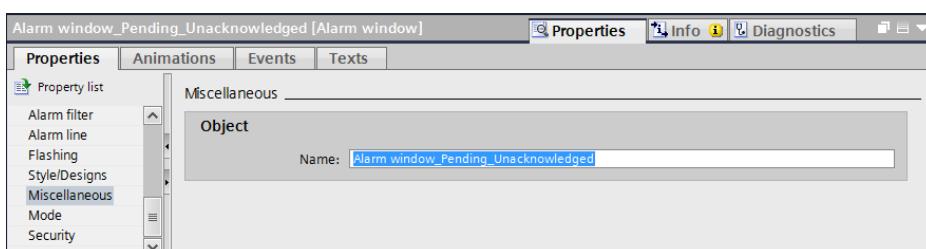


- As a second alarm window in the "Global screen" there is → "Pending alarms". Under "General" in "Properties", activate "Pending alarms" and "Unacknowledged alarms". Activate "Errors", "Warnings" and "No Acknowledgment" as alarm classes.

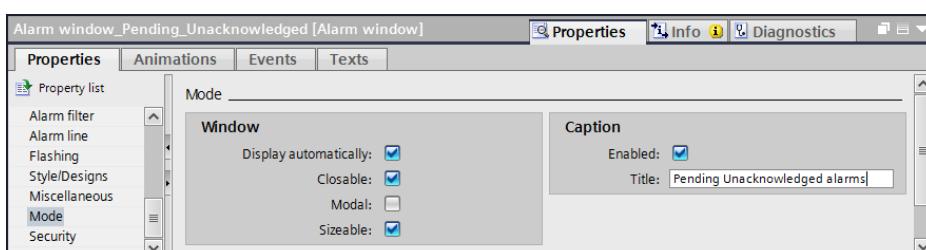


Note: You will create alarm classes of types "Errors" and "Warnings" in the panel itself in the following steps. The alarm class of the "No acknowledgment" type is produced automatically by the settings for the system diagnostics in the CPU 1516F.

- Under "Miscellaneous" in "Properties", change the "Name" to → "Alarm window_pending_not_acknowledged".

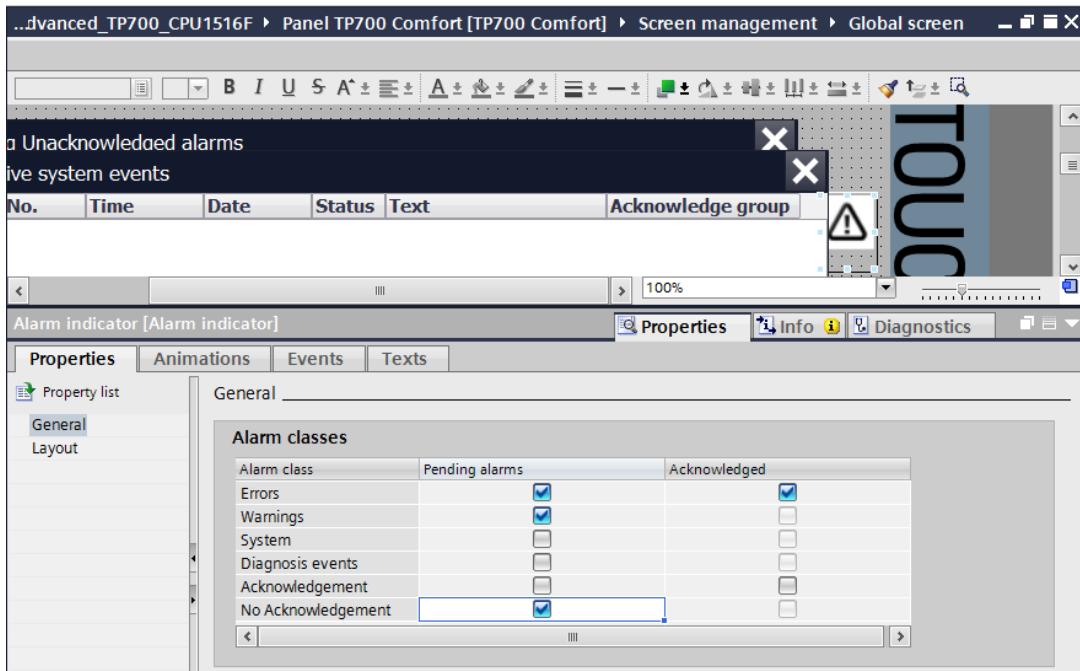


- Under "Mode" in "Properties", change the "Title" to → "Pending/not acknowledged alarms".

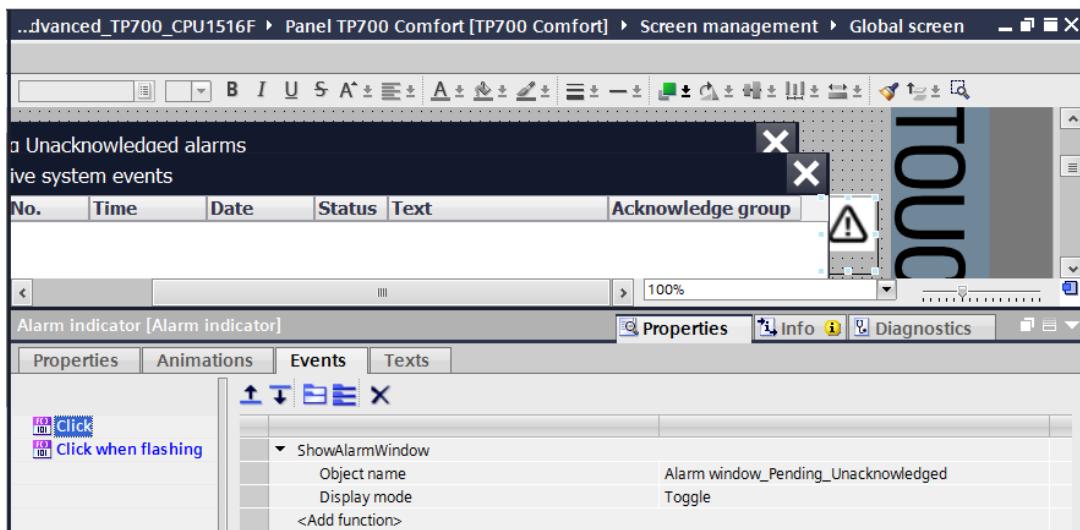


7.16.3 Alarm indicator

- In addition to the alarm windows there is an → "Alarm indicator" in the "Global screen". This is used so that an alarm window that was closed by the user is displayed again. Under "General" in "Properties", activate the alarm classes "Errors: Pending alarms", "Errors: Acknowledged", "Warnings: Pending alarms" and "No Acknowledgment: Pending alarms".

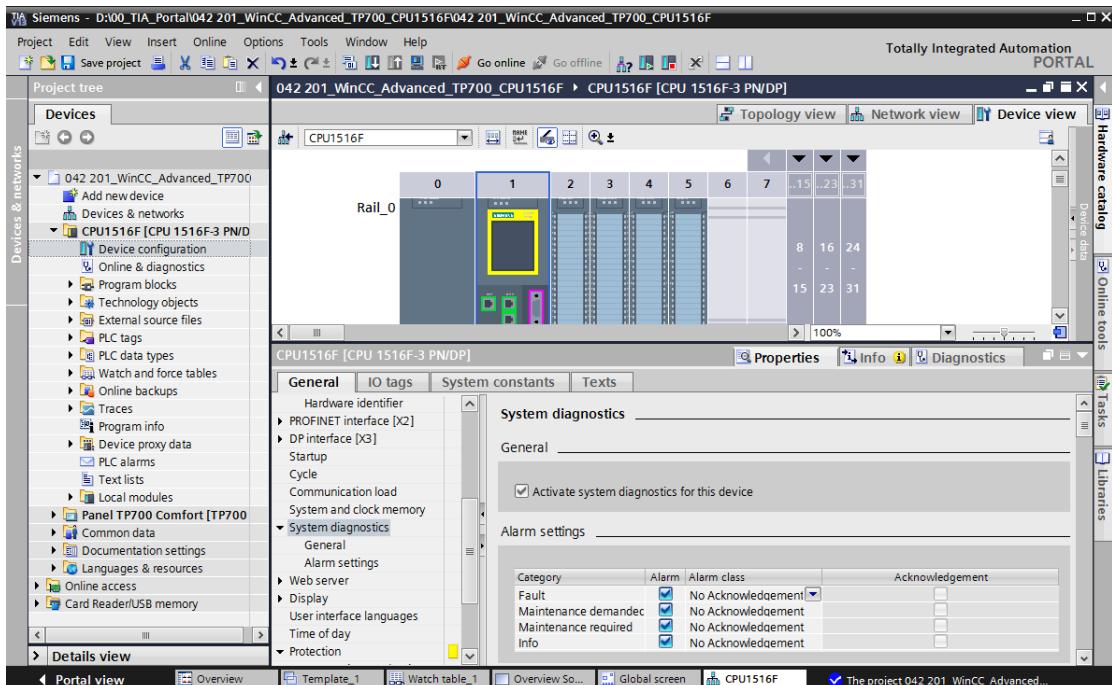


- In → "Events", the display of the alarm window "Alarm window_pending_not_acknowledged" with function "ShowMessageWindow" has already been stored for "Click" and "Click when flashing".

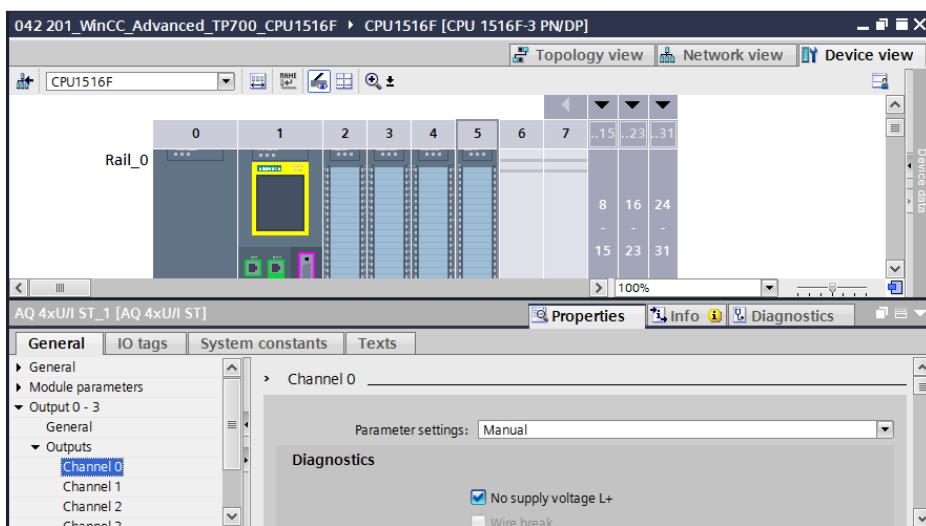


7.16.4 Alarms for system diagnostics of CPU 1516F

- Open the "Device configuration" of the "CPU_1516F" and select the → "System diagnostics" menu under → "General" in → "Properties".
 System diagnostics is always activated for devices of the SIMATIC S7-1500 series.
 Under → "Alarm setting", you see which alarm classes are being used. Here, the alarm class is "No Acknowledgment". These alarm classes are referenced in turn by the HMI systems in the alarm windows and for the other settings for the alarms.



- Select the analog output module → "AQ4xU/I" and activate → "Diagnostics" →
 "Missing voltage supply L+" for all channels under → "Outputs" in → Properties.



- Before the project is downloaded to the CPU and the panel, compile the CPU and panel again and save the project.

(→ CPU_1516F → → Panel TP700 Comfort → → Save project)

- After successful compilation, the complete controller with the created program including the hardware configuration can be downloaded, as described in the previous modules.

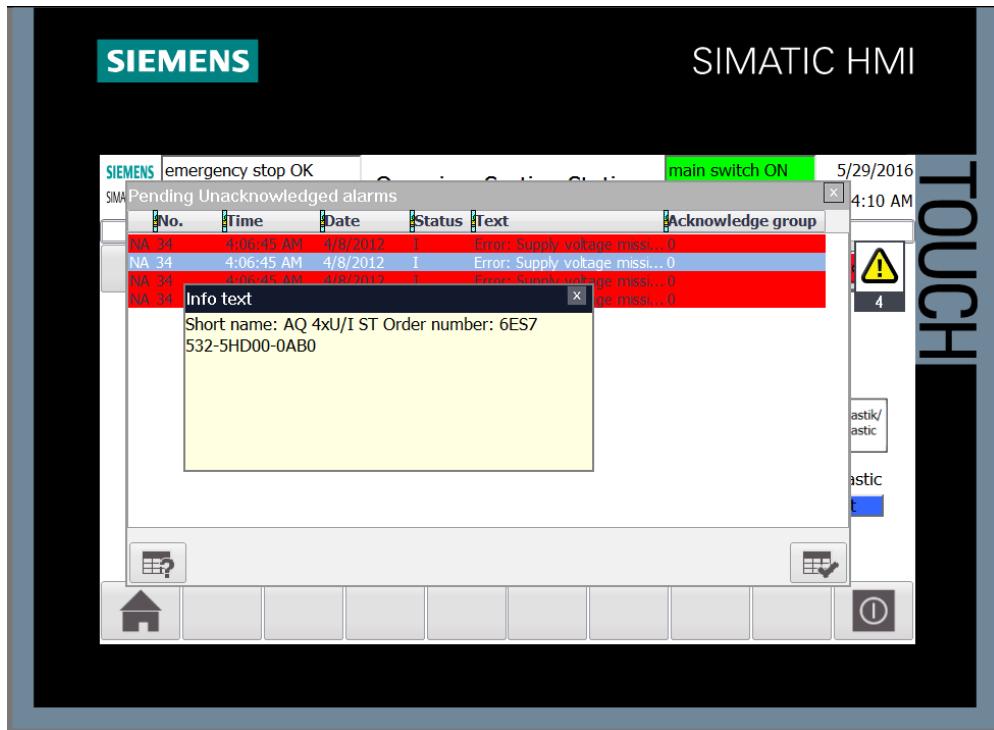
(→ CPU_1516F →

- To download the visualization to the panel, proceed in a similar way. Select the → "Panel TP700 Comfort [TP700 Comfort]" folder and click the icon → "Download to device".

- Alarms for system diagnostics of the CPU 1516F are now automatically displayed in the alarm window "Pending/Unacknowledged alarms". Details and help texts can be displayed and alarms can be acknowledged if necessary in this alarm window. If the alarm window has been closed, it can be displayed again by clicking the displayed

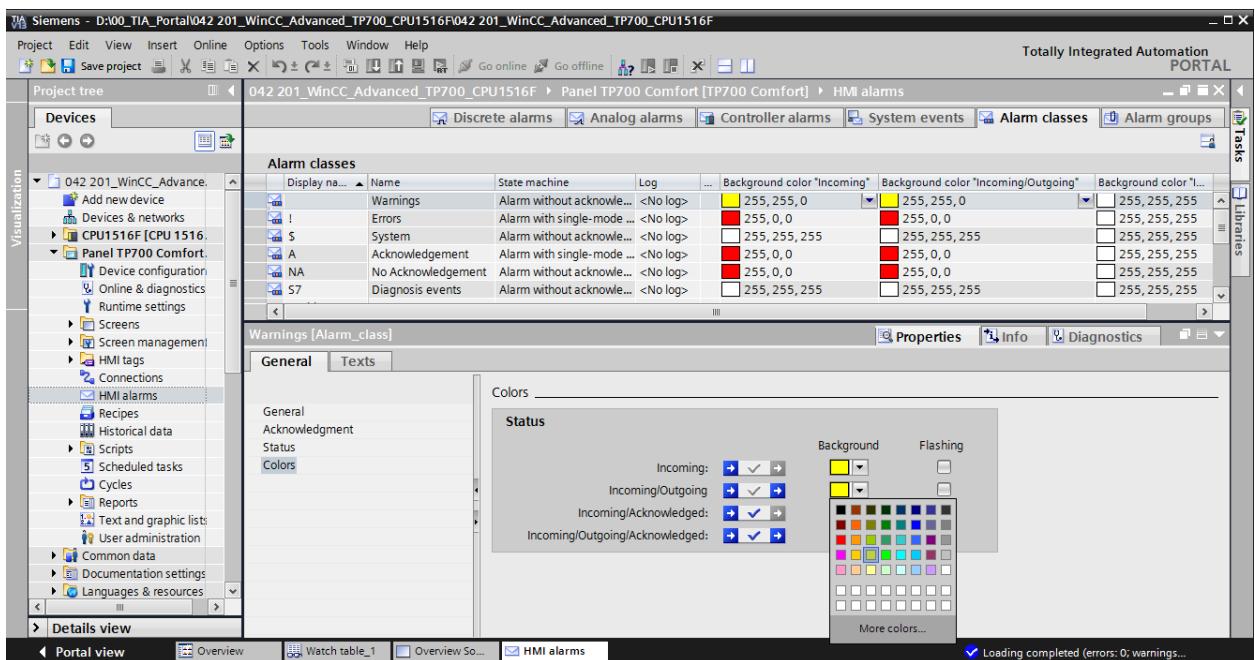


alarm indicator. This example shows a failure of the supply voltage for the "AQ4xU/I" module.



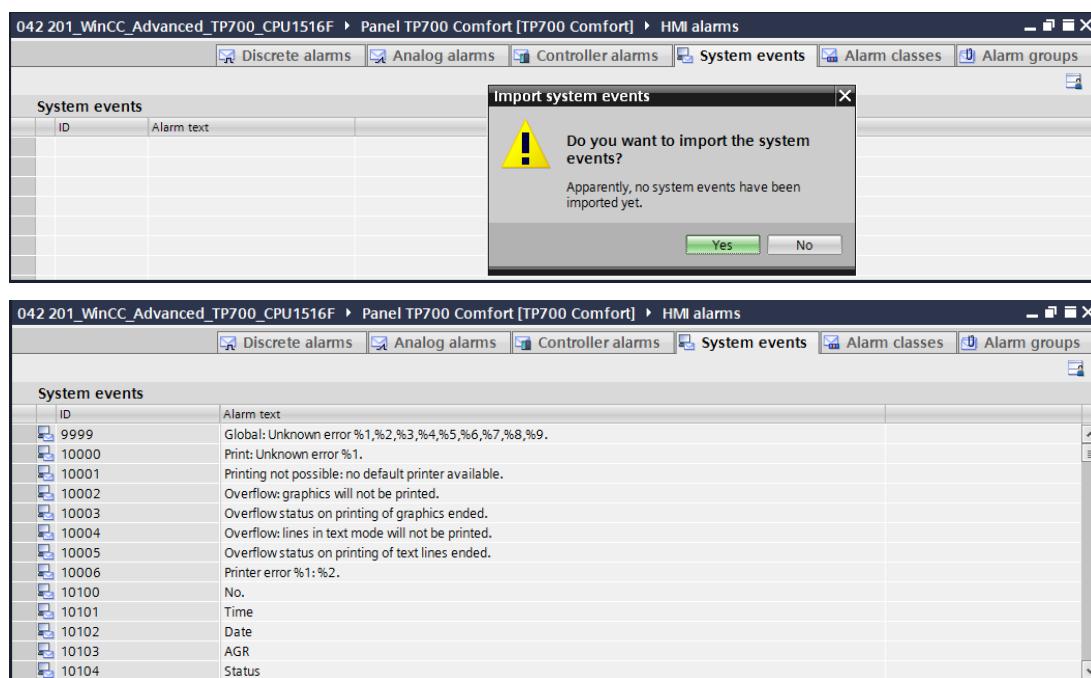
7.16.5 Alarm class settings

- The item → "HMI alarms" is available in the → "Panel TP700 Comfort" for configuring the alarm system and creating individual alarms. Open this with a double-click. The alarm classes we are using have already been created in the menu command "Alarm classes" but these can also be changed. For alarm class → "Warnings", change the background color for the "Incoming" and "Outgoing" states to → "Yellow".



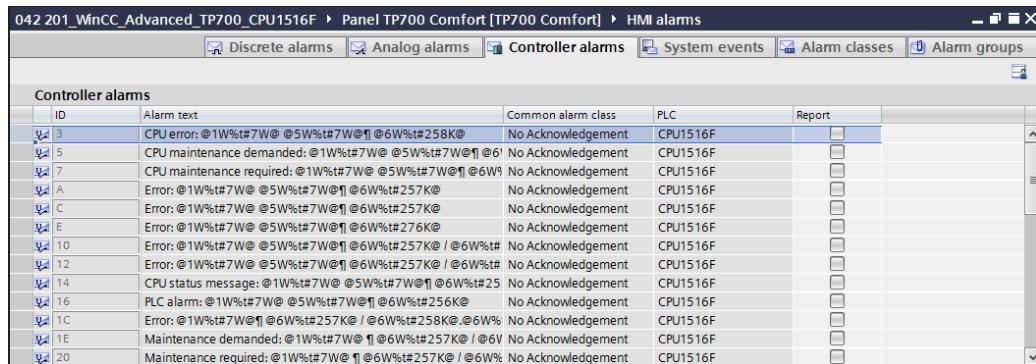
7.16.6 System events

- With the menu command "System events", you can have these automatically imported by clicking → "Yes".



7.16.7 Controller alarms

- The controller alarms have already been created with the menu command "Controller alarms".

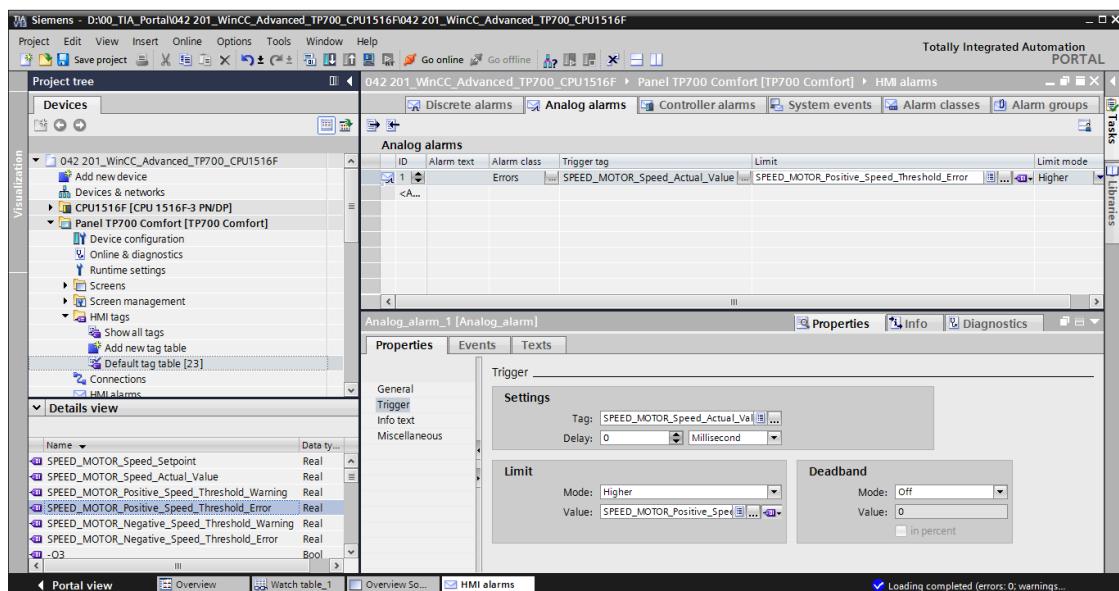


The screenshot shows the 'Controller alarms' tab in the 'HMI alarms' window of the TIA Portal. The table lists 20 entries, each with an ID, alarm text, common alarm class, PLC, and report status. Most entries are related to CPU errors and maintenance.

ID	Alarm text	Common alarm class	PLC	Report
3	CPU error: @1W%#7W@ @5W%#7W@1 @6W%#258K@	No Acknowledgement	CPU1516F	<input type="checkbox"/>
5	CPU maintenance demanded: @1W%#7W@ @5W%#7W@1 @6V	No Acknowledgement	CPU1516F	<input type="checkbox"/>
7	CPU maintenance required: @1W%#7W@ @5W%#7W@1 @6W%	No Acknowledgement	CPU1516F	<input type="checkbox"/>
A	Error: @1W%#7W@ @5W%#7W@1 @6W%#257K@	No Acknowledgement	CPU1516F	<input type="checkbox"/>
C	Error: @1W%#7W@ @5W%#7W@1 @6W%#257K@	No Acknowledgement	CPU1516F	<input type="checkbox"/>
E	Error: @1W%#7W@ @5W%#7W@1 @6W%#276K@	No Acknowledgement	CPU1516F	<input type="checkbox"/>
10	Error: @1W%#7W@ @5W%#7W@1 @6W%#257K@ / @6W%#	No Acknowledgement	CPU1516F	<input type="checkbox"/>
12	Error: @1W%#7W@ @5W%#7W@1 @6W%#257K@ / @6W%#	No Acknowledgement	CPU1516F	<input type="checkbox"/>
14	CPU status message: @1W%#7W@ @5W%#7W@1 @6W%#25	No Acknowledgement	CPU1516F	<input type="checkbox"/>
16	PLC alarm: @1W%#7W@ @5W%#7W@1 @6W%#256K@	No Acknowledgement	CPU1516F	<input type="checkbox"/>
1C	Error: @1W%#7W@1 @6W%#257K@ / @6W%#258K@ @6W%	No Acknowledgement	CPU1516F	<input type="checkbox"/>
1E	Maintenance demanded: @1W%#7W@1 @6W%#257K@ / @6V	No Acknowledgement	CPU1516F	<input type="checkbox"/>
20	Maintenance required: @1W%#7W@1 @6W%#257K@ / @6W%	No Acknowledgement	CPU1516F	<input type="checkbox"/>

7.16.8 Analog alarms

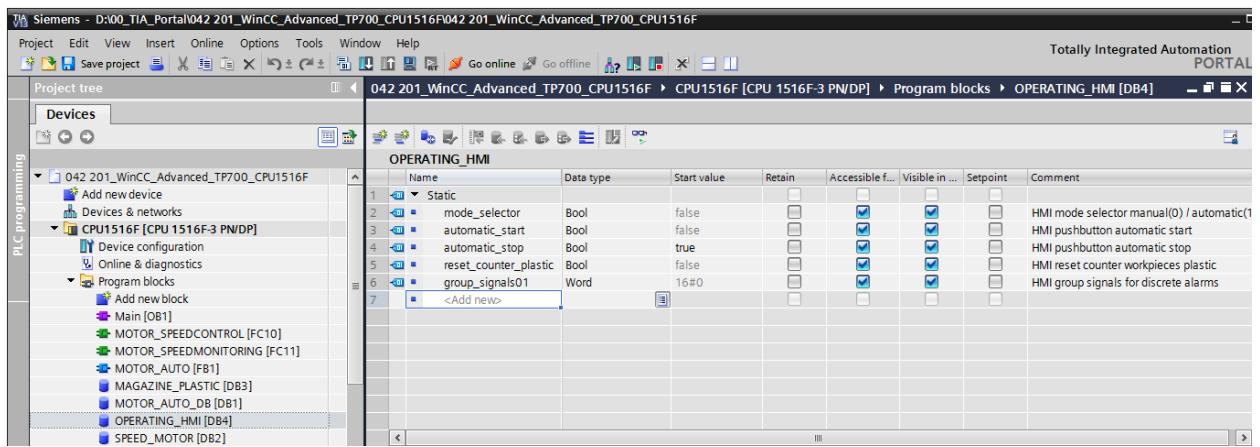
- You can monitor tags to determine whether they are within limits in "Analog alarms". Create a new alarm by clicking "Add". For monitoring purposes, select the → "Default tag table" in → "Panel TP700 Comfort" and drag the → "SPEED_MOTOR_actual_speed_value" tag that is to be monitored from the → "Detail view" into the field for "Trigger tag". Drag the variable limit → "SPEED_MOTOR_positive_speed_error_limit" from the → "Detail view" into the field for "Limit".



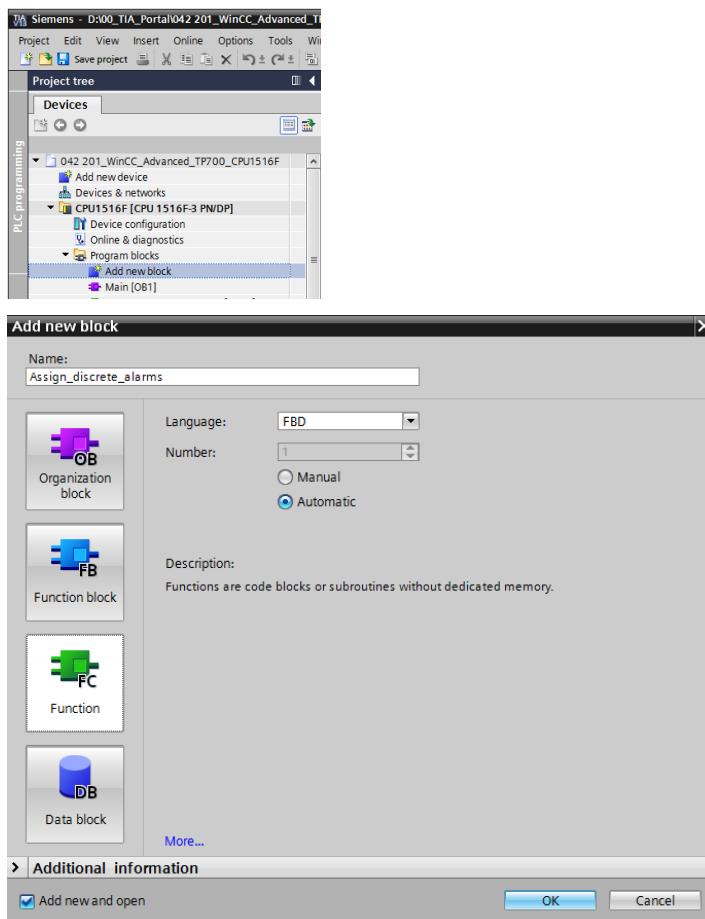
- Enter the text → "Error limit exceeded positive motor speed", select the "Alarm class" → "Errors" and "Mode" → "High". Create the three other alarms of alarm classes "Warnings" and "Errors" shown below in the same way.

7.16.9 Discrete alarms

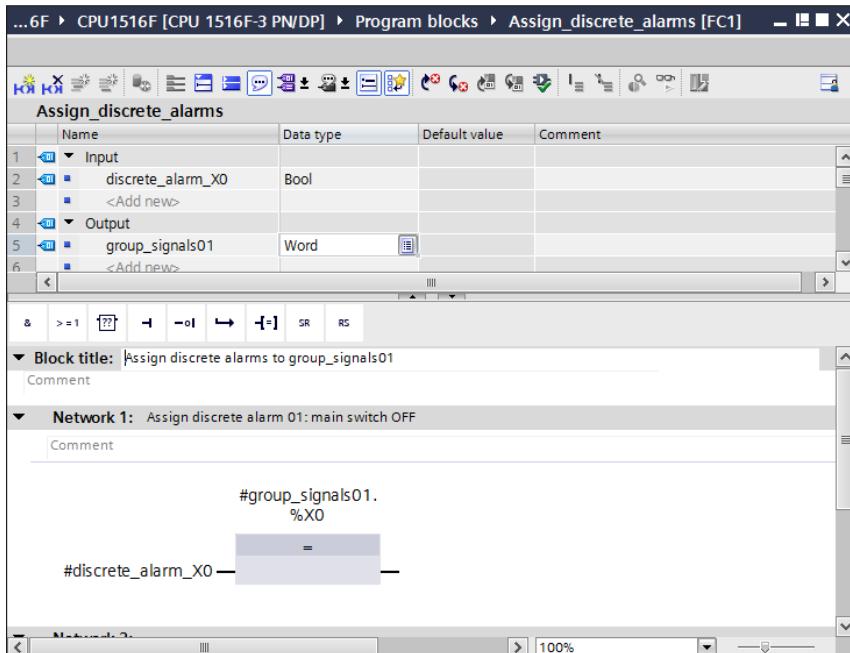
- Before you can create discrete alarms in the panel, you need a global tag with at least 16 bits in the CPU 1516F. You will use this tag to trigger the discrete alarms from the PLC. In "CPU 1516F", open the data block → "OPERATION_HMI[DB4]" in the → "Program blocks" folder and create a global tag → "Group alarms 01" of data type → "Word" there.



- In the → "Program blocks" folder, click → "Add new block" to create → Funktion → "Discrete alarm_assignment".

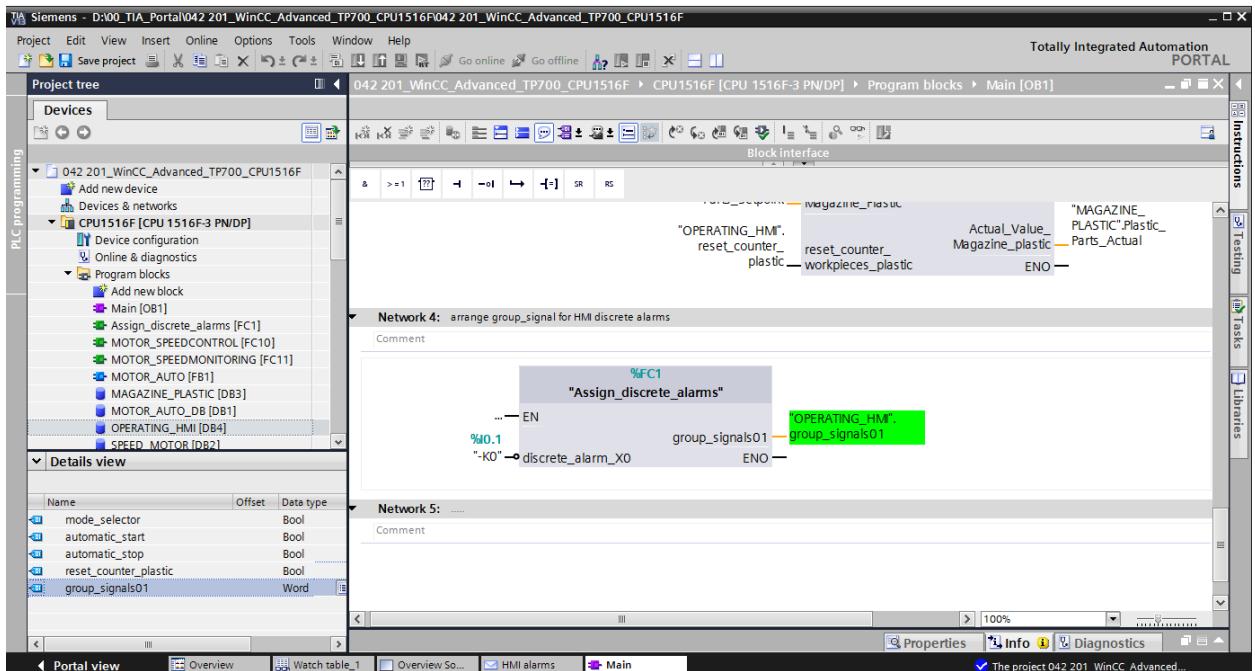


- In the "Discrete alarm_assignment" function, create a local input tag → "DiscreteAlarmX0" of data type → "Bool" and a local output tag → "GroupAlarms01" of data type → "Word". In the first network, program a single $\neg=1$ assignment of the → "DiscreteAlarmX0" tag to Bit X0 in the → "GroupAlarms01" tag.

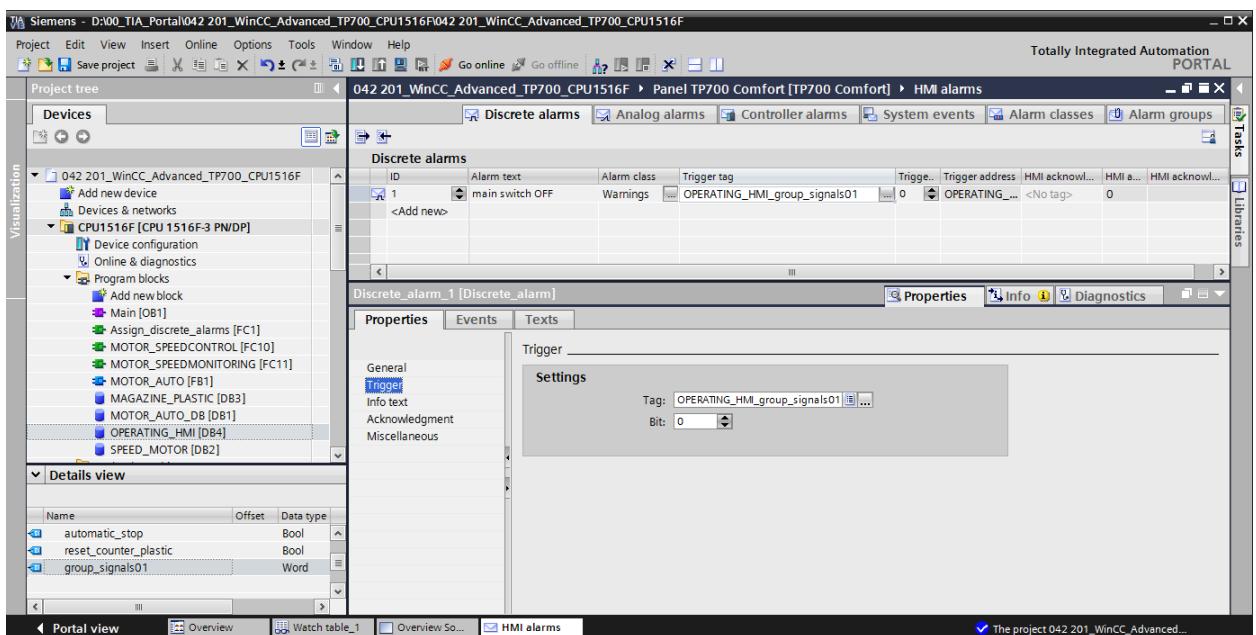


Note: The "Tag1.%X0" syntax is referred to as **Slice** access in the TIA Portal. This enables access to the individual bits of a tag of data type Byte, Word or DWord: If you need additional information about this, search for the term "Slice" in the online help for STEP 7 Professional V1X.

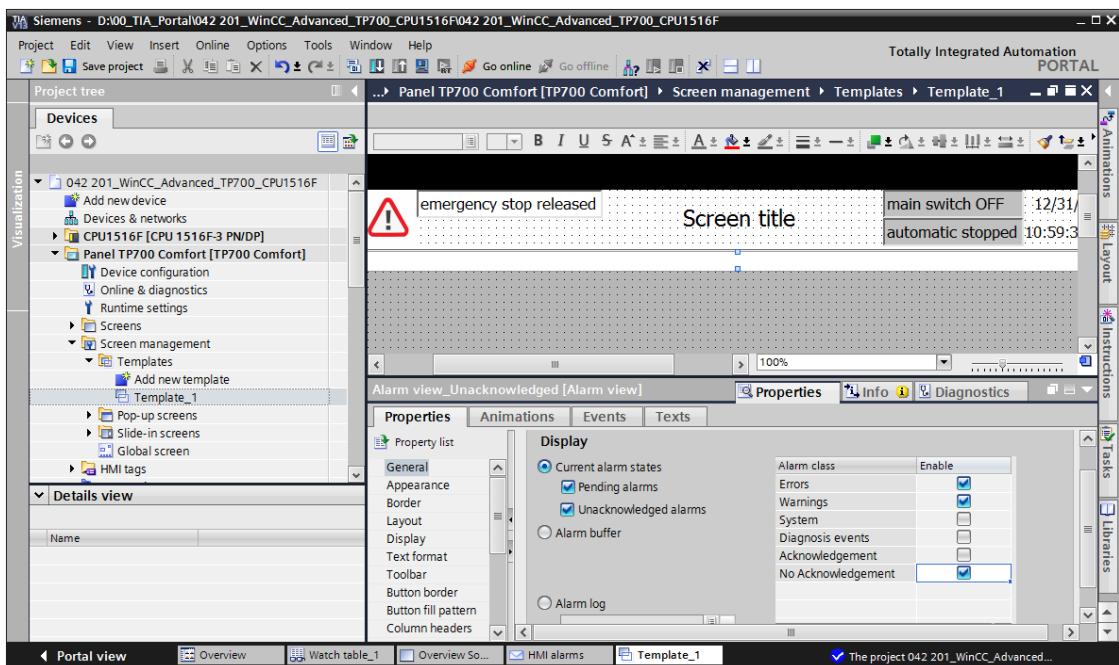
- Open the → "Main[OB1]" block from the "Program blocks" folder and call the → "Discrete_alarm_assignment[FC1]" function in → "Network 4". Connect the input of the "Discrete_alarm_assignment[FC1]" function to the **inverted** global tag → "-K0" / %I0.1 / Station "ON"(no) from the "Tag_table_sorting_station". Connect the output of the "Discrete_alarm_assignment[FC1]" function to the global tag → "GroupAlarms01" from the data block "OPERATION_HMI[DB4]".



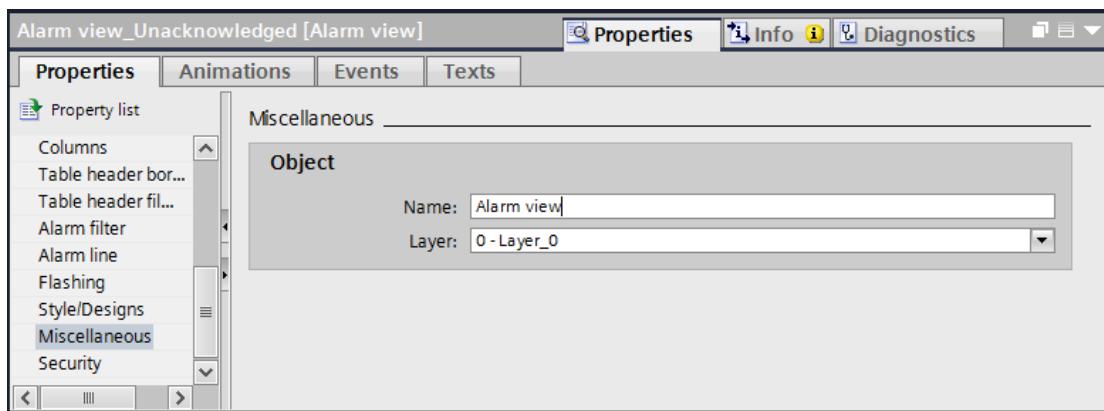
- Return to → "HMI alarms" → "Discrete alarms" in the "Panel TP700 Comfort". Create a new alarm by clicking → "Add". Select the → "GroupAlarms01" tag you just created from the "OPERATION_HMI[DB4]" data block as the "Trigger tag". Enter the text → "Main switch OFF" in the "Alarm text" column, select the "Alarm class" → "Warnings" and "Trigger bit" → "0". In the "Trigger address" column, "OPERATION_HMI. GroupAlarms01.x0" is now displayed.



- The alarm display is also adapted in "Template_1" in the "Screen management" folder. Under "General" in "Properties", activate → "Pending alarms". Activate → "Errors", → "Warnings" and → "No Acknowledgment" as alarm classes.



→ Under "Miscellaneous" in "Properties", change the "Name" to → "Alarm line".



→ Before the project is downloaded to the CPU and the panel, compile the CPU and panel again and save the project.

(→ CPU_1516F → → Panel TP700 Comfort → →

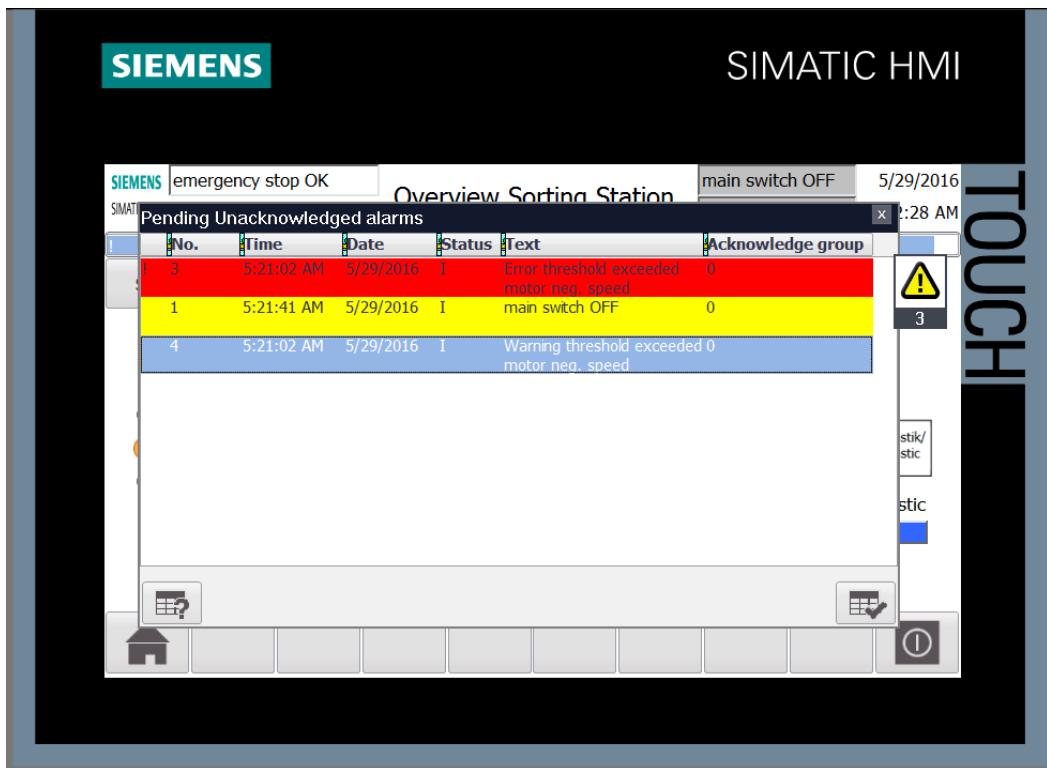
→ After successful compilation, the complete controller with the created program including the hardware configuration can be downloaded, as described in the previous modules.

(→ CPU_1516F →

→ To download the visualization to the panel, proceed in a similar way. Select the → "Panel TP700 Comfort [TP700 Comfort]" folder and click the icon → "Download to device".

→ Analog alarms and discrete alarms for system are now automatically displayed in the alarm window "Pending/Unacknowledged alarms" and in the "Alarm line". Details and help texts can be displayed and alarms can be acknowledged if necessary in the alarm

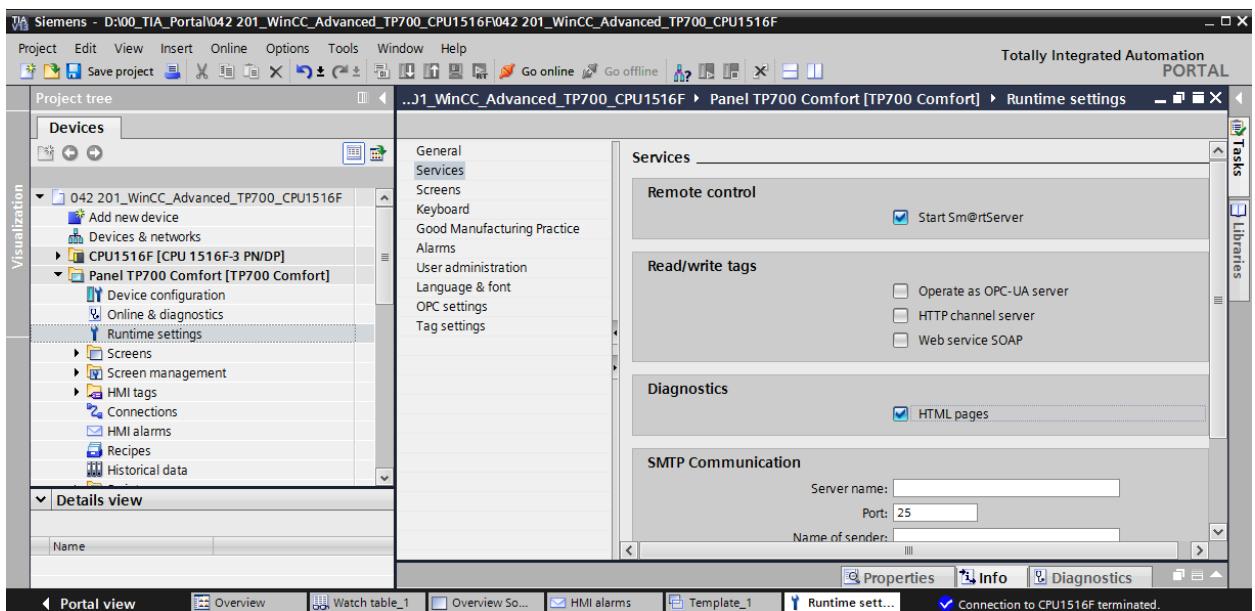
window. If the alarm window has been closed, it can be displayed again by clicking the displayed alarm indicator. Various alarm classes appear in different colors.



7.17 Remote control of the TP700 Comfort Panel

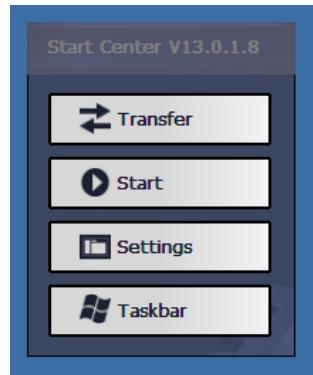
7.17.1 Activating web services for Runtime

- To enable remote control, double-click the → "Runtime settings" in the configuration for the → Panel TP700 Comfort to open them. The → "Start Sm@rtServer" option is activated for → "Layout" under "Remote control" and the → "HTML pages" option is activated under "Diagnostics".



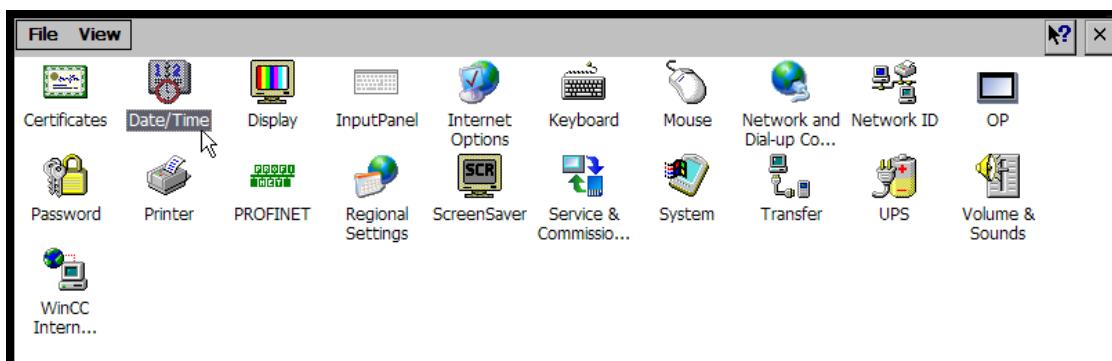
7.17.2 WinCC Internet settings in the TP700 Comfort Panel

- Settings must also be made directly on the panel. Select → "Settings" in the "Start Center" directly after switching on the voltage supply and the start of the panel.

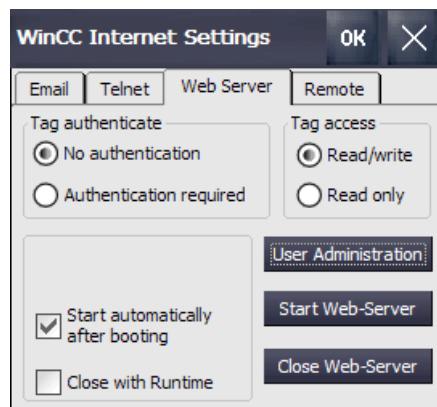


Note: The selection of "Settings" must occur fast enough, and before the automatic "Start" of Runtime.

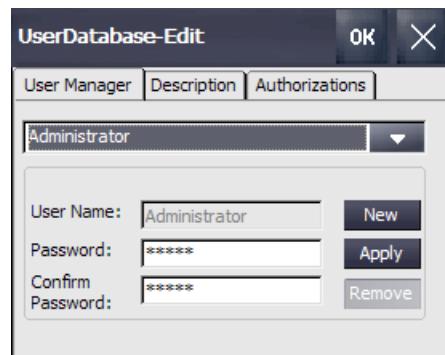
- Select  from the icons in order to make the web server settings.



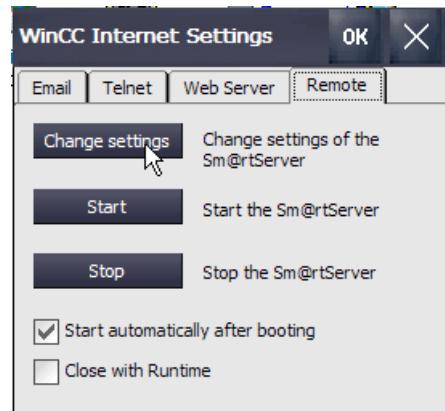
- With menu command "Web Server", first select the "Start automatically after booting" option and click on → "User Administration".



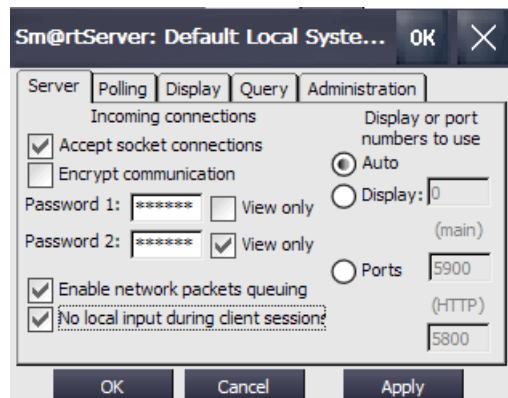
- Assign the administrator password in the "Password" field → "100", confirm it in the "Confirm Password" field → "100" and apply it with → "Apply".



- With menu command "Remote", first select the "Start automatically after booting" option and click → "Change settings".

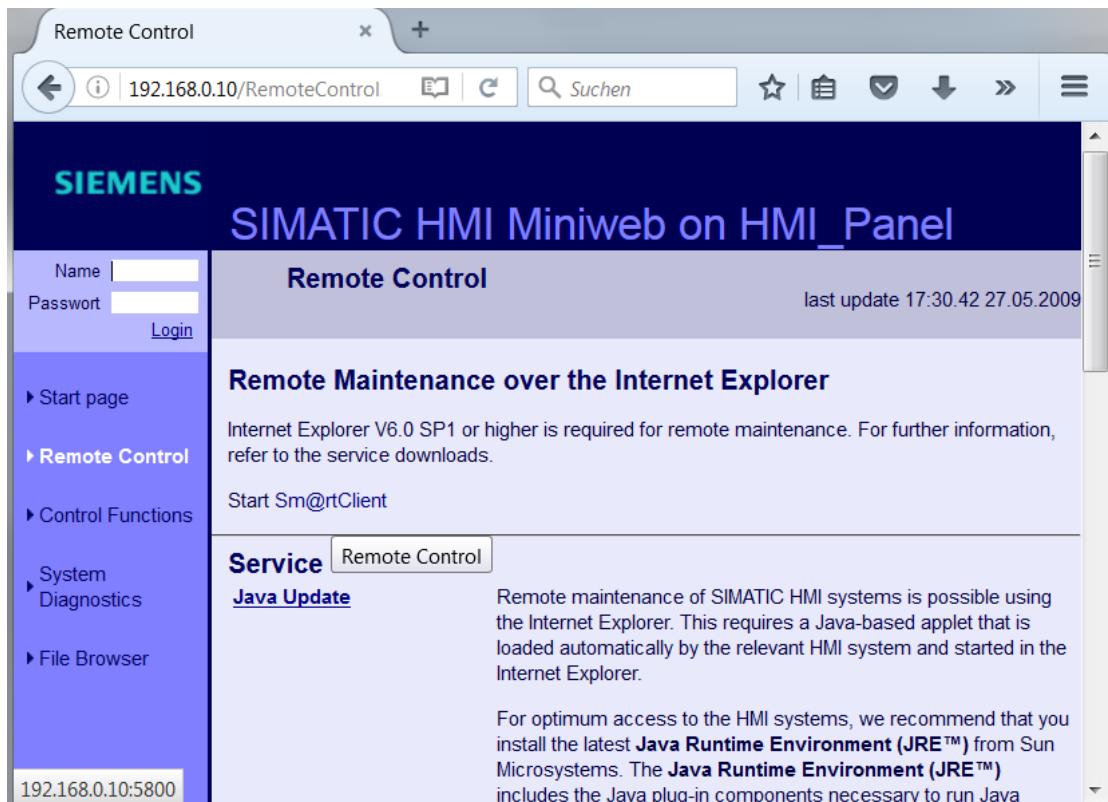


- Also assign → "100" for Password1 and → "100" for Password2 and put this into effect with → "Apply".



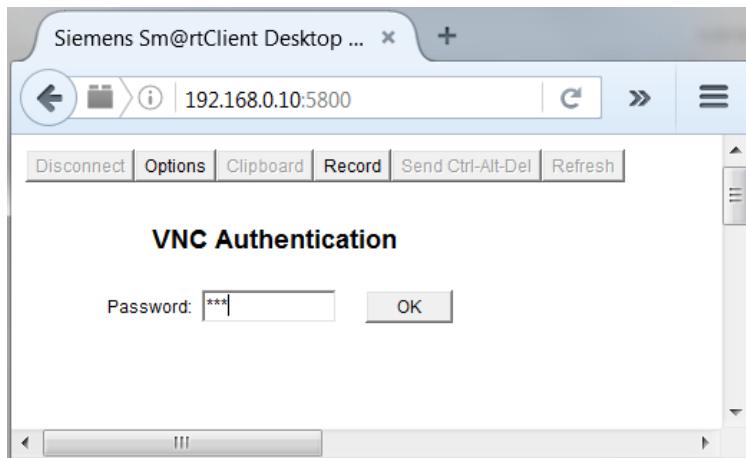
7.17.3 Starting remote access to the TP700 Comfort Panel

- To start the remote access to your panel, enter the IP address of the in your browser
- "192.168.0.10".

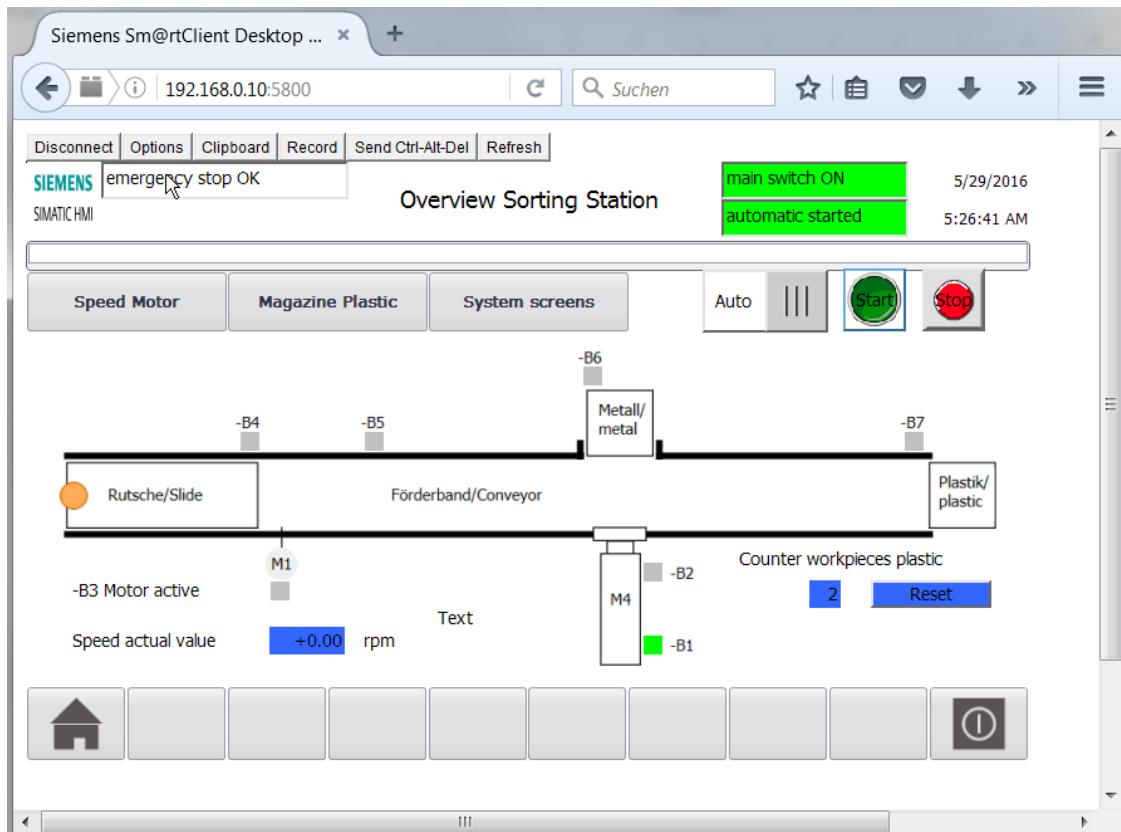


Note: A current installation of Java Runtime Environment is required for secure access to the web services of the TP700 Comfort Panel.

- Enter the password that was previously set in the panel → "100".



- You then have the option of monitoring, and operating the panel remotely and even changing the settings in Windows CE of the device.



8 Checklist

No.	Description	Completed
1	Program changes successfully performed in the CPU 1516F	
2	CPU 1516F successfully compiled without error message	
3	CPU 1516F successfully downloaded without error message	
4	Process visualization for the Touch Panel TP700 Comfort successfully created	
5	Touch Panel TP700 Comfort successfully compiled without error message	
6	Touch Panel TP700 Comfort successfully downloaded without error message	
7	Switch on station (-K0 = 1) Cylinder retracted/Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (in the panel) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (in the panel) Sensor part at slide activated (-B4 = 1) then conveyor motor -M1 variable speed (-Q3 = 1) switches on and remains active The speed corresponds to the speed setpoint in the range +/- 50 rpm	
8	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
9	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
10	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
11	Manual mode (in the panel) → -Q3 = 0	
12	Switch off station (-K0 = 0) → -Q3 = 0	
13	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
14	Speed > SPEED_MOTOR_monitoring_error_max → -Q3 = 0	
15	Speed < SPEED_MOTOR_monitoring_error_min → -Q3 = 0	
16	Values and alarms are displayed in the panel.	
17	Project successfully archived	



Training Curriculum

TIA Portal Module 012

PID Controller
for SIMATIC S7-1500

Table of contents

1	Goal.....	441
2	Prerequisite.....	441
3	Required hardware and software.....	441
4	Theory	442
4.1	Tasks of closed loop controls.....	442
4.2	Components of a control loop	442
4.3	Step function for analysis of controlled systems	444
4.4	Controlled systems with self-regulation	445
4.4.1	Proportional system without time delay.....	445
4.4.2	Proportional system with time delay	446
4.4.3	Proportional system with two time delays	447
4.4.4	Proportional system with n time delays.....	448
4.5	Systems without self-regulation.....	449
4.6	Basic types of continuous controllers	450
4.6.1	The proportional controller (P controller).....	450
4.6.2	The integral controller (I controller)	451
4.6.3	The PI controller	452
4.6.4	The derivative controller (D controller).....	453
4.6.5	The PID controller	453
4.7	Controller tuning using the oscillation test	454
4.8	Controller tuning with T_u - T_g approximation	455
4.8.1	Tuning the PI controller according to the Ziegler-Nichols method.....	455
4.8.2	Tuning the PI controller according to the Chien, Hrones and Reswick method	456
4.9	Digital controllers	456
5	Task.....	458
6	Planning	458
6.1	PID_Compact closed-loop control block.....	458
6.2	Technology diagram	459
6.3	Reference list	459
7	Structured step-by-step instructions.....	460
7.1	Retrieve an existing project.....	460
7.2	Call PID_Compact controller in a cyclic interrupt OB	461
7.3	Save and compile the program	467
7.4	Download the program.....	468
7.5	Monitor PID_Compact	469
7.6	PID_Compact pretuning.....	470
7.7	PID_Compact fine tuning	472
8	Checklist	475

PID Controller for SIMATIC S7-1500

1 Goal

In this chapter, you will become acquainted with the use of software PID controllers for the SIMATIC S7-1500 with the TIA Portal programming tool.

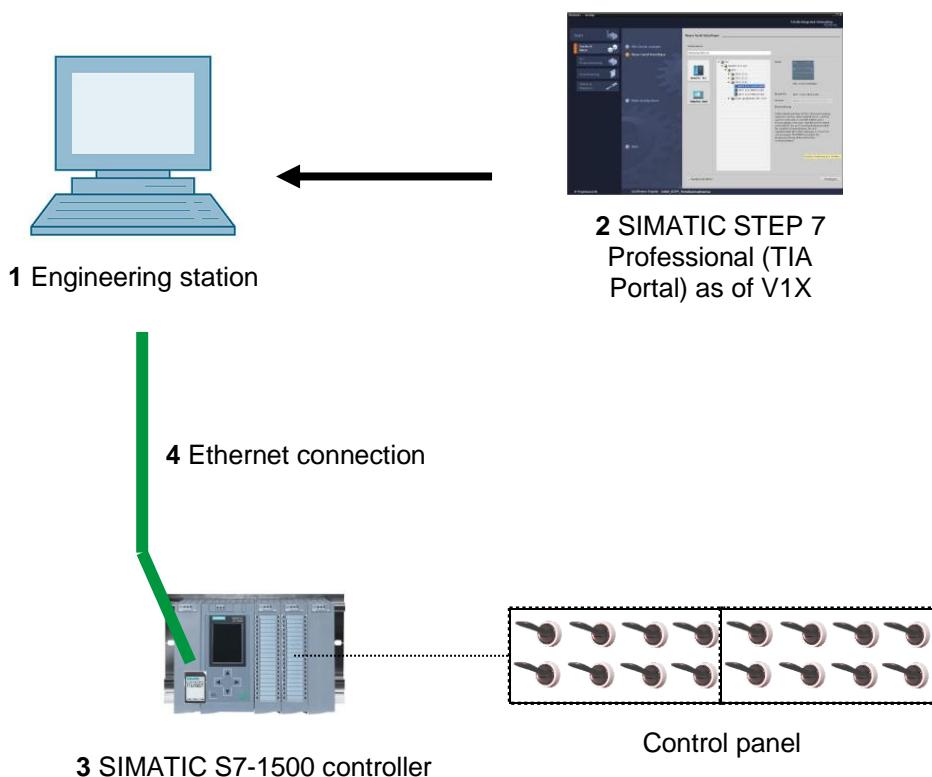
The module explains the call-up, connection, configuration and optimization of a PID controller for the SIMATIC S7-1500. It also shows the steps for calling the PID controller in the TIA Portal and integrating it into a user program.

2 Prerequisite

This chapter builds on the chapter Analog Values with the SIMATIC S7 CPU1516F-3 PN/DP. You can use the following project for this chapter, for example: Analog

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system
(for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP –
Firmware as of V1.6 with memory card and 16DI/16DO and 2AI/1AO
Note: The digital inputs and analog inputs and outputs should be fed out to a control panel.
- 4 Ethernet connection between engineering station and controller



4 Theory

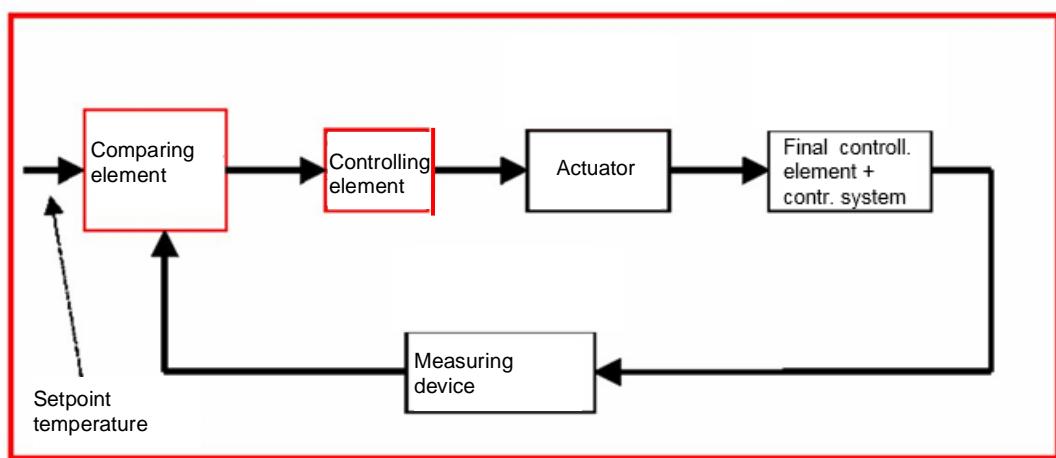
4.1 Tasks of closed loop controls

Closed loop control is a process in which the value of a variable is generated and maintained continuously through an intervention based on measurements of this variable.

This produces an action path that takes place in a closed loop – the control loop – because the process runs based on measurements of a variable that is, in turn, influenced by itself.

The variable to be controlled is continuously measured and compared with another preset variable of the same type. Depending on the result of this comparison, an adjustment of the variable to be controlled to the value of the preset variable is made.

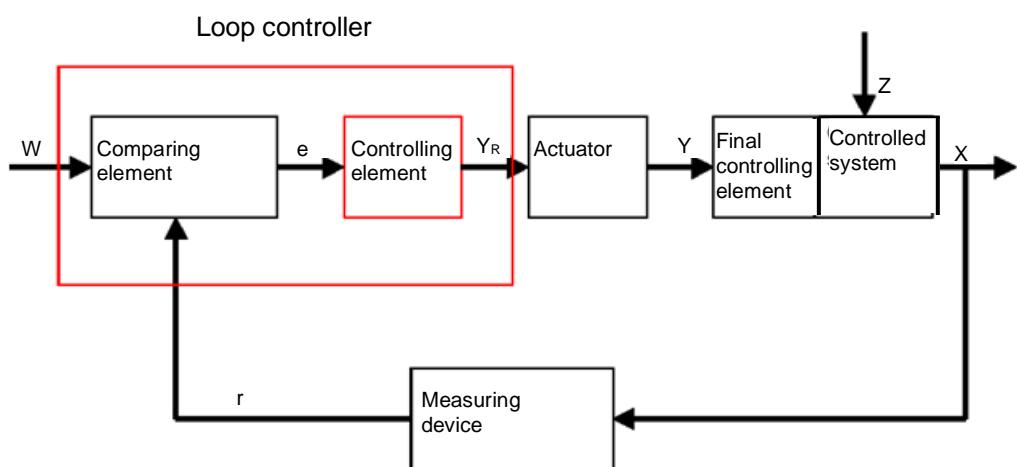
Diagrammatic representation of a closed loop control



4.2 Components of a control loop

The fundamental concepts of closed loop controls are explained in detail in the following.

An overview based on a diagram is presented here to start.



1. The controlled variable x

This is the actual "target" of the closed-loop control, namely the variable that is to be influenced or kept constant. In our example, this would be the room temperature. The instantaneous value of the controlled variable at a particular time is called the "actual value" at this time.

2. The feedback variable r

In a control loop, the controlled variable is continuously checked to enable a response to unwanted changes. The measured quantity proportional to the controlled variable is called the feedback variable. In the "Heating" example, it would correspond to the measured voltage of the inside thermometer.

3. The disturbance variable z

The disturbance variable is the variable that influences the controlled variable in an unwanted way and moves it away from the current setpoint. In the case of fixed setpoint control, this control is only necessary in the first place due to the existence of the disturbance variable. In the examined heating system, this would be, for example, the outside temperature or any other variable that causes the room temperature to move away from its ideal value.

4. The setpoint w

The setpoint at a given time is the value that the controlled variable should ideally have at this time. Note that the setpoint may vary continuously in a slave control. In our example, the setpoint would be the currently desired room temperature.

5. The comparing element

This is the point at which the current measured value of the controlled variable and the instantaneous value of the reference variable are compared. In most cases, both variables are measured voltages. The difference between the two variables is the "system error" e. This is passed to the controlling element and evaluated there (see below).

6. The controlling element

The controlling element is the actual heart of a closed loop control. It evaluates the system error, thus the information regarding whether, how and how much the controlled variable deviates from the current setpoint, as an input variable and derives from this the "**Controller output variable**" Y_R , which is ultimately used to influence the controlled variable. In the heating system example, the controller output variable would be the voltage for the mixer motor.

The manner in which the controlling element determines the controller output variable from the system error is the main criterion of the closed-loop control.

7. The actuator

The actuator is, so to speak, the "executive organ" of the closed loop control. It receives information from the controlling element in the form of the controller output variable indicating how the controlled variable is to be influenced and translates this into a change of the "manipulated variable". In our example, this would be the mixer motor controller.

8. The final controlling element

This is the element of the control loop that influences the controlled variable (more or less directly) as a function of the **manipulated variable** Y . In the example, this would be the combination of the mixer, heating lines and radiators. The adjustment of the mixer (the manipulated variable) is made by the mixer motor (actuator) and influences the room temperature by means of the water temperature.

9. The controlled system

The controlled system is the system containing the variable to be controlled, thus the living space in the heating example.

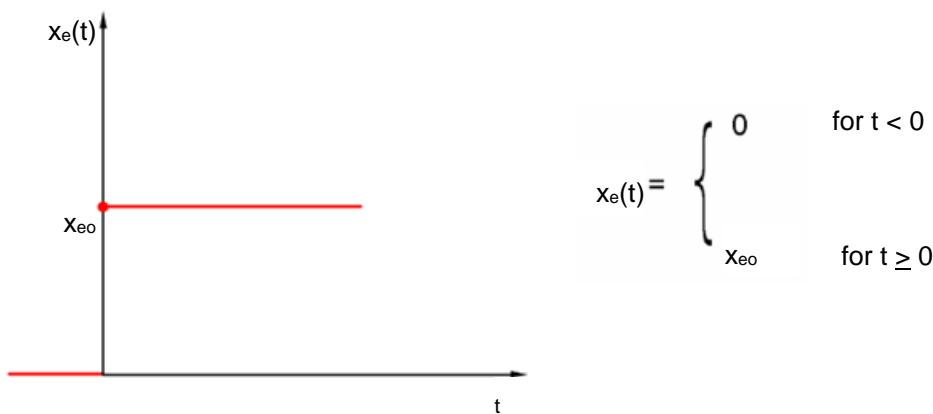
10. The dead time

The dead time refers to the time that elapses from a change in the controller output variable until there is a measurable response in the controlled system. In the example, this would be the time between a change in the voltage for the mixer motor and a measurable change in the room temperature resulting from this.

4.3 Step function for analysis of controlled systems

To analyze the response of controlled systems, controllers and control loops, a uniform function for the input signal is used – the step function.

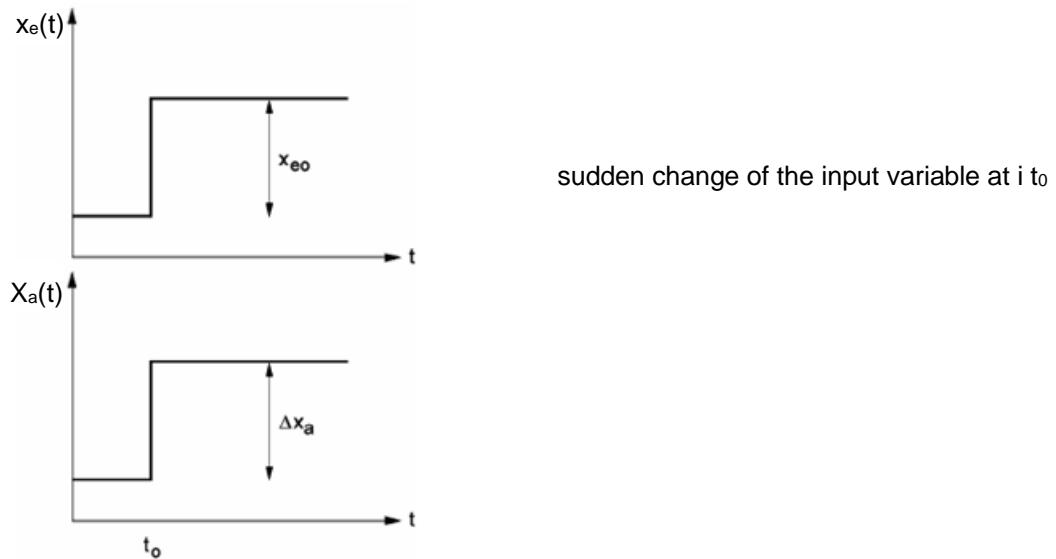
Depending on whether a control loop element or the entire control loop is being analyzed, the controlled variable $x(t)$, the manipulated variable $y(t)$, the reference variable $w(t)$ or the disturbance variable $z(t)$ can be assigned the step function. The input signal is often designated $x_e(t)$ and the output signal $x_a(t)$.



4.4 Controlled systems with self-regulation

4.4.1 Proportional system without time delay

This controlled system is called a P system for short.



Controlled variable/manipulated variable:

$$x = K_{ss} \cdot y$$

K_{ss} : Proportional coefficient for a manipulated variable change:

$$K_{ss} = \frac{\Delta x}{\Delta y} = \tan \alpha$$

Controlled variable/disturbance variable:

$$x = K_{sz} \cdot z$$

K_{sz} : Proportional value for a disturbance variable change

Range:

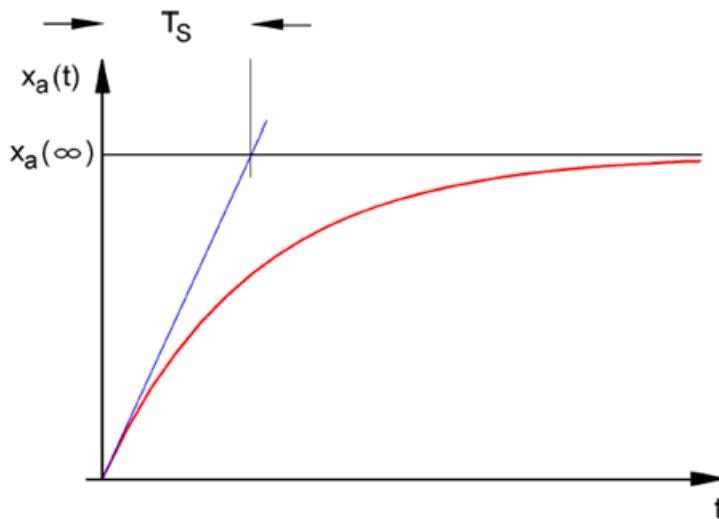
$$y_h = y_{\max} - y_{\min}$$

Control range:

$$x_h = x_{\max} - x_{\min}$$

4.4.2 Proportional system with time delay

This controlled system is called a P-T1 system for short.



Differential equation for a general input signal $x_e(t)$:

$$T_S \cdot x_a(t) + x_a(t) = K_{PS} \cdot x_e(t)$$

Solution of the differential equation for a step function at the input (step response)

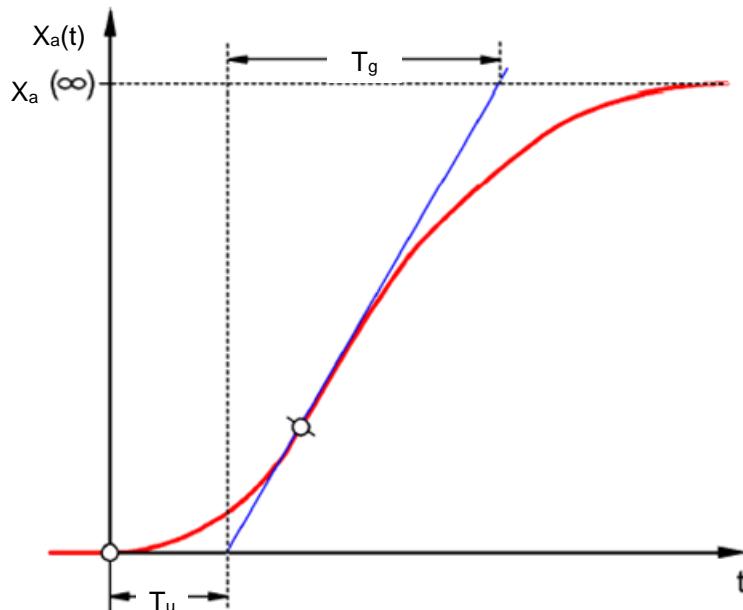
$$x_a(t) = K_{PS} (1 - e^{-t/T_S}) \cdot x_{eo}$$

$$x_a(t = \infty) = K_{PS} \cdot x_{eo}$$

T_S : Time constant

4.4.3 Proportional system with two time delays

This system is called a P-T2 system for short.



Tu: Delay time Tg: Compensation time

The system is generated through the reaction-free series connection of two P-T1 systems that have the time constants TS1 and TS2.

Controllability of P-Tn systems:

$$\frac{T_u}{T_g} < \frac{1}{10} \rightarrow \boxed{\text{easily controllable}} \quad \frac{T_u}{T_g} \approx \frac{1}{6} \rightarrow \boxed{\text{still controllable}} \quad \frac{T_u}{T_g} > \frac{1}{3} \rightarrow \boxed{\text{difficult to control}}$$

With the increasing ratio Tu/Tg, the system becomes less and less controllable.

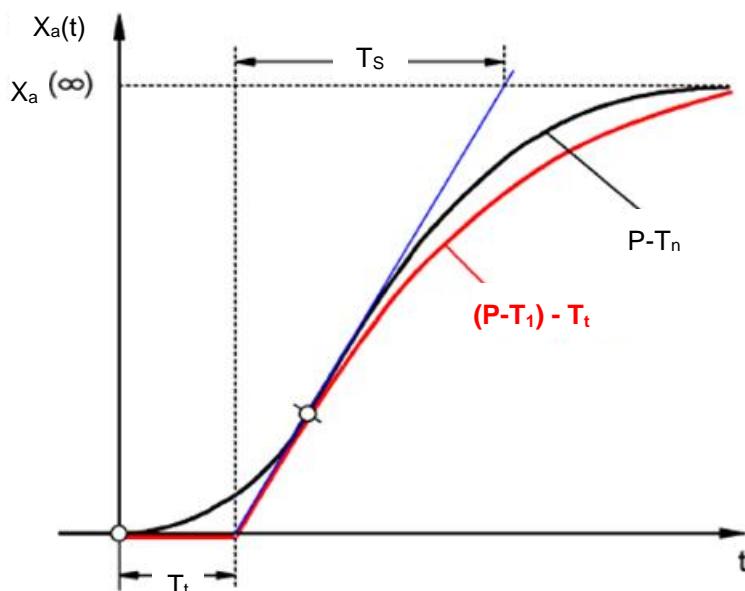
4.4.4 Proportional system with n time delays

This controlled system is called a P-T_n system for short.

The time response is described by an nth order differential equation. The step response characteristic is similar to that of the P-T₂ system. The time response is described by T_u and T_g.

Substitute: An approximate substitution for the system with many delays is the series connection of a P-T₁ system with a dead time system.

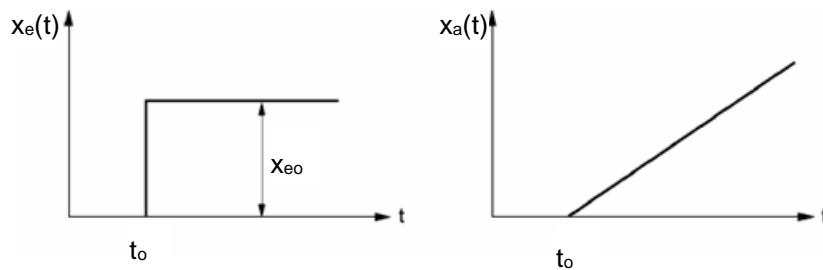
The following applies: T_t » T_u and T_S » T_g.



4.5 Systems without self-regulation

This controlled system is called an I system for short.

After a disturbance, the controlled variable continues increasing steadily without striving for a fixed final value.

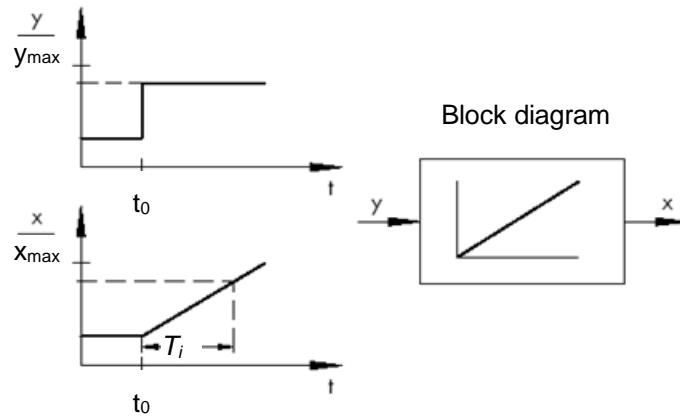


Example: Level control

For a tank with discharge outlet, whose incoming and outgoing flow rates are the same, there is a constant fill height. If the incoming or outgoing flow rate changes, the liquid level rises or falls. The level changes faster as the difference between the incoming flow rate and outgoing flow rate increases.

It is clear from this example that, in practice, the integral action has a limit in most cases. The controlled variable increases or decreases only until a system-inherent limit value is reached. A tank runs over or drains dry, pressure reaches the system maximum or minimum, etc.

The figure shows the time response of an I system to a step change in the input variable as well as the derived block diagram:



If the step function at the input changes to a function $x_e(t)$, then

$$x_a(t) = K_{IS} \int x_e(t) dt \Rightarrow \text{integrating controlled system}$$

K_{IS} : Integral coefficient of the controlled system

* Figure from SAMSON Technical Information - L102 Controllers and Controlled Systems, Edition: August 2000 (http://www.samson.de/pdf_en/l102en.pdf)

4.6 Basic types of continuous controllers

Discrete controllers that only switch one or two manipulated variables on and off have the advantage of simplicity. Both the controller itself and the actuator and final controlling element are simpler in nature and thus less expensive than continuous controllers.

Discrete controllers have several disadvantages, however. For one thing, when large loads such as large electric motors or cooling units must be switched, high load peaks may occur at switch-on and overload the power supply, for example. For this reason, these often do not switch between "Off" and "On" but instead between full power ("full load") and a significantly lower power of the actuator or final controlling element ("base load"). Still, even with this improvement, a discrete closed-loop control is unsuitable for numerous applications. Consider an automobile engine whose speed is discreetly controlled. There would then be nothing between idle and full throttle. Apart from the fact that it would probably be impossible to properly transfer the forces from a sudden full-throttle to the road via the tires, such a vehicle would probably be unsuitable for road traffic.

Continuous controllers are therefore used for such applications. Theoretically, hardly any limits are placed on the mathematical relationship that establishes the controlling element between the system error and controller output variable. In practice, however, three classic basic types are differentiated. These will be described in more detail in the following.

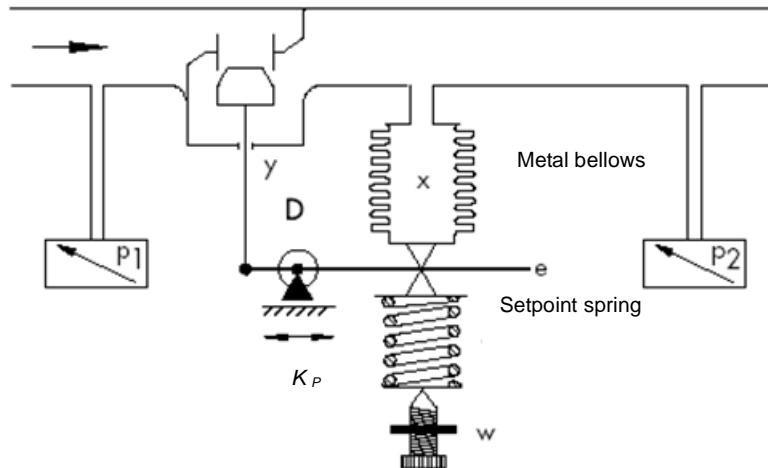
4.6.1 *The proportional controller (P controller)*

The manipulated variable y of a P controller is proportional to the measured error e . From this can be deducted that a P controller reacts to any deviation without lag and only generates a manipulated variable in case of system deviation.

The proportional pressure controller illustrated in the figure compares the force FS of the setpoint spring with the force FB created in the elastic metal bellows by the pressure p_2 . When the forces are off balance, the lever pivots about point D. This changes the position of the valve plug –and, hence, the pressure p_2 to be controlled –until a new equilibrium of forces is restored.

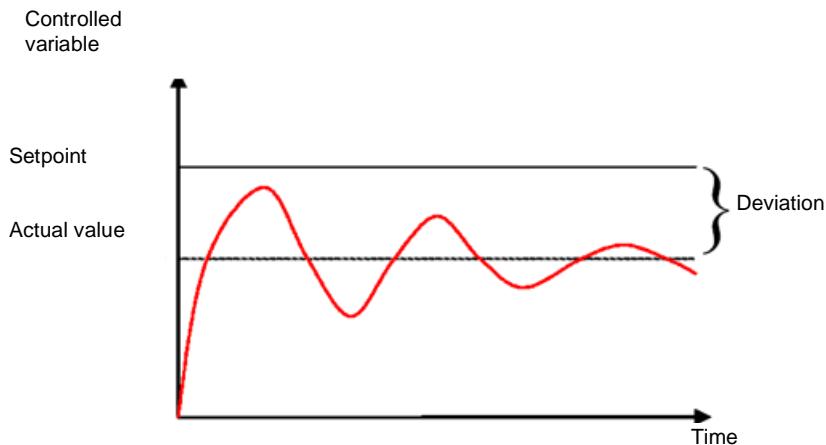
The dynamic behavior of the P controller after a step change in the error variable is shown in the figure. The amplitude of the manipulated variable y is determined by the error e and the proportional-action coefficient K_p .

To keep the control deviation as small as possible, as large a proportional-action coefficient as possible must be selected. An increase in the factor causes the controller to react faster, but if the value is too high there is a risk of overshooting and a large "hunting" tendency of the controller.



$$y = K_P \cdot e$$

You see the response of the P controller in the diagram.



The advantages of this controller type lie, on the one hand, in its simplicity (in the simplest case, it can be implemented electronically with just a resistor) and, on the other hand, in its very prompt reaction compared to other controller types.

The main disadvantage of the P controller is its permanent system deviation. That is, the setpoint is never fully reached even over the long term. This disadvantage as well as the not yet ideal response speed cannot be minimized to a satisfactory extent through a larger proportional-action coefficient, because this leads to overshooting by the controller, or in other words an overreaction. In the worst case, the controller goes into a permanent oscillation in which the controlled variable is periodically moved away from the setpoint by the controller itself instead of by the manipulated variable.

The problem of permanent control deviation is best solved by an additional integral controller.

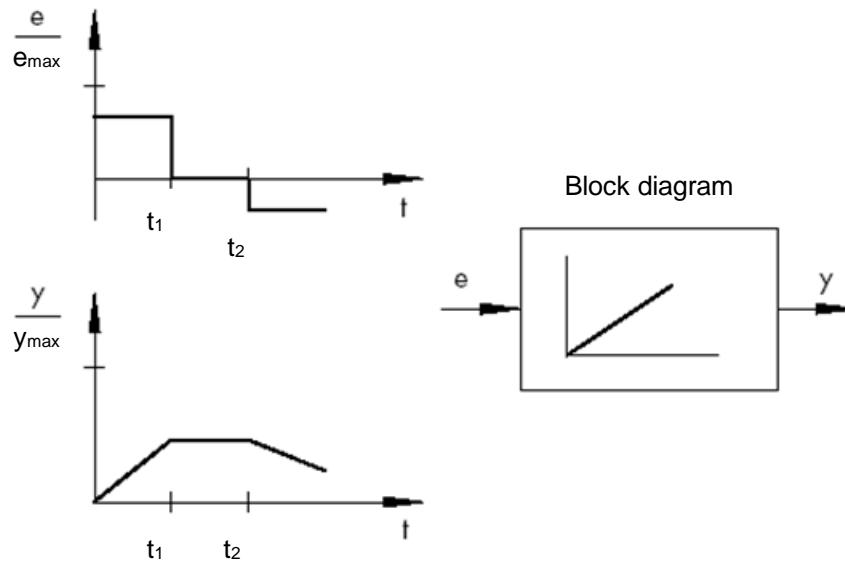
4.6.2 The integral controller (I controller)

Integral control action is used to fully correct system deviations at any operating point. As long as the error is nonzero, the integral action will cause the value of the manipulated variable to change. Only when reference variable and controlled variable are equally large –at the latest, though, when the manipulated variable reaches its system specific limit value (Umax, pmax, etc.)– is the control process balanced.

Mathematics expresses integral action as follows: the value of the manipulated variable is changed proportional to the integral of the error e.

$$y = K_i \int e dt \quad \text{with} \quad K_i = \frac{1}{T_n}$$

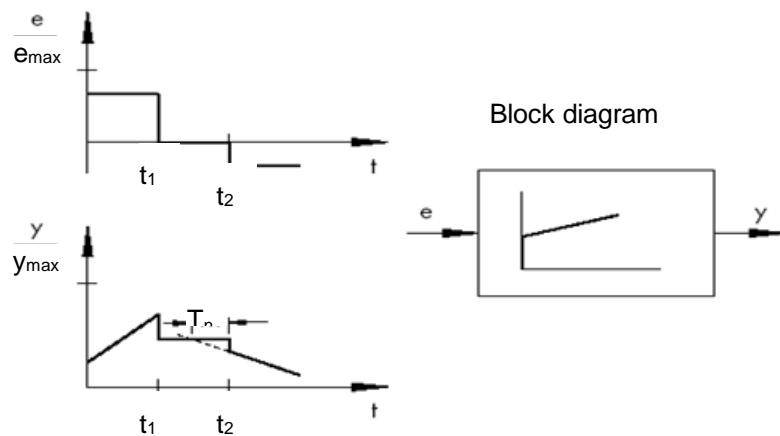
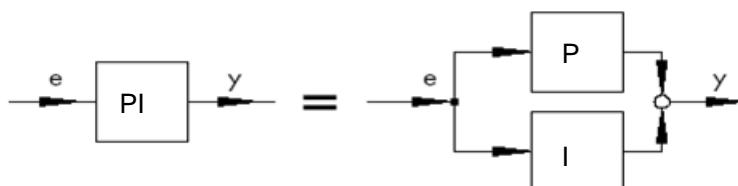
How rapidly the manipulated variable increases/decreases depends on the error and the integral time.



4.6.3 The PI controller

PI controllers are often employed in practice. In this combination, one P and one I controller are connected in parallel.

If properly designed, they combine the advantages of both controller types (stability and rapidity; no steady-state error), so that their disadvantages are compensated for at the same time.



The dynamic behavior is marked by the proportional-action coefficient K_p and the reset time T_n . Due to the proportional component, the manipulated variable immediately reacts to any error signal e , while the integral component starts gaining influence only after some time. T_n represents the time that elapses until the I component generates the same control amplitude that is generated by the P component (K_p) from the start. As with I controllers, the reset time T_n must be reduced if the integral-action component is to be amplified.

Controller dimensioning:

By adjusting the K_p and T_n values, oscillation of the controlled variable can be reduced, however, at the expense of control dynamics.

PI controller applications: Fast control loops allowing no steady-state error

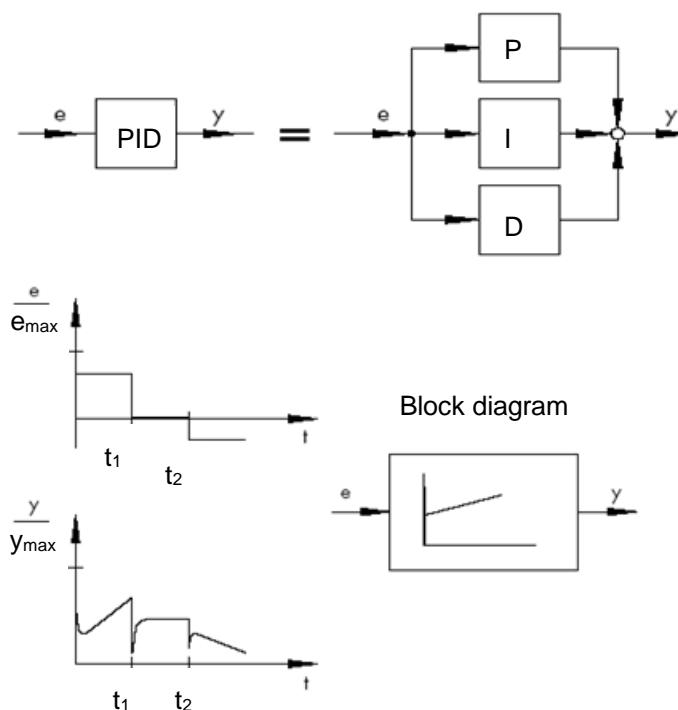
Examples: pressure, temperature, ratio control, etc.

4.6.4 The derivative controller (D controller)

D controllers generate the manipulated variable from the rate of change of the error and not—as P controllers—from their amplitude. Therefore, they react much faster than P controllers: even if the error is small, derivative controllers generate—by anticipation, so to speak—large control amplitudes as soon as a change in amplitude occurs. A steady-state error signal, however, is not recognized by D controllers, because regardless of how big the error, its rate of change is zero. Therefore, derivative-only controllers are rarely used in practice. They are usually found in combination with other control elements, mostly in combination with proportional control.

4.6.5 The PID controller

If a D component is added to PI controllers, the result is an extremely versatile PID controller. As with PD controllers, the added D component—if properly tuned—causes the controlled variable to reach its setpoint more quickly, thus reaching steady state more rapidly.



$$y = K_p \cdot e + K_i \int e dt + K_D \frac{de}{dt} \quad \text{with} \quad K_i = \frac{K_p}{T_n}; \quad K_D = K_p \cdot T_v$$

4.7 Controller tuning using the oscillation test

For a satisfactory control result, the selection of a suitable controller is an important aspect. It is even more important that the control parameters K_p , T_n and T_v be appropriately adjusted to the system response. Mostly, the adjustment of the controller parameters remains a compromise between a very stable, but also very slow control loop and a very dynamic, but irregular control response which may easily result in oscillation, making the control loop instable in the end.

For nonlinear systems that should always work in the same operating point, e.g. fixed setpoint control, the controller parameters must be adapted to the system response at this particular operating point. If a fixed operating point cannot be defined, such as with follow-up control systems ñ, the controller must be adjusted to ensure a sufficiently rapid and stable control result within the entire operating range.

In practice, controllers are usually tuned on the basis of values gained by experience.

Should these not be available, however, the system response must be analyzed in detail, followed by the application of several theoretical or practical tuning approaches in order to determine the proper control parameters.

One approach is a method first proposed by Ziegler and Nichols, the so-called ultimate method. It provides simple tuning that can be applied in many cases. This method, however, can only be applied to controlled systems that allow sustained oscillation of the controlled variable.

For this method, proceed as follows:

- At the controller, set K_p and T_v to the lowest value and T_n to the highest value (smallest possible influence of the controller).
- Adjust the controlled system manually to the desired operating point (start up control loop).
- Set the manipulated variable of the controller to the manually adjusted value and switch to automatic operating mode.
- Continue to increase K_p (decrease X_p) until the controlled variable encounters harmonic oscillation. If possible, small step changes in the setpoint should be made during the K_p adjustment to cause the control loop to oscillate.
- Take down the adjusted K_p value as critical proportional-action coefficient $K_{p,crit}$. Determine the time span for one full oscillation amplitude as T_{crit} , if necessary by taking the time of several oscillations and calculating their average.
- Multiply the values of $K_{p,crit}$ and T_{crit} by the values according to the table and enter the determined values for K_p , T_n and T_v at the controller.

	K_p	T_n	T_v
P	$0.50 \times K_{p,crit}$	-	-
PI	$0.45 \times K_{p,crit}$	$0.85 \times T_{crit}$	-
PID	$0.59 \times K_{p,crit}$	$0.50 \times T_{crit}$	$0.12 \times T_{crit}$

4.8 Controller tuning with T_u - T_g approximation

The tuning of the controlled systems will be performed here using the example of a PT2 system.

T_u - T_g approximation

The Ziegler-Nichols method and the Chien, Hrones and Reswick method are based on the T_u - T_g approximation in which the transfer coefficient of the system K_s , delay time T_u and balancing time T_g parameters are determined from the system step response.

The tuning rules, which are described below, are the result of experiments using analog computer simulations.

P-T_N systems can be described with sufficient accuracy with a so-called T_u - T_g approximation, that is, through approximation using a P-T₁-T_L system.

The starting point is the system step response with input step height K . The required parameters (transfer coefficient of the system K_s , delay time T_u and balancing time T_g) are determined as shown in the figure.

The transfer function must be measured up to the final steady-state value (K^*K_s) so that the transfer coefficient of the system K_s required for the calculation can be determined.

The main advantage of this method is that the approximation can also be used when an analytical description of the system is not possible.

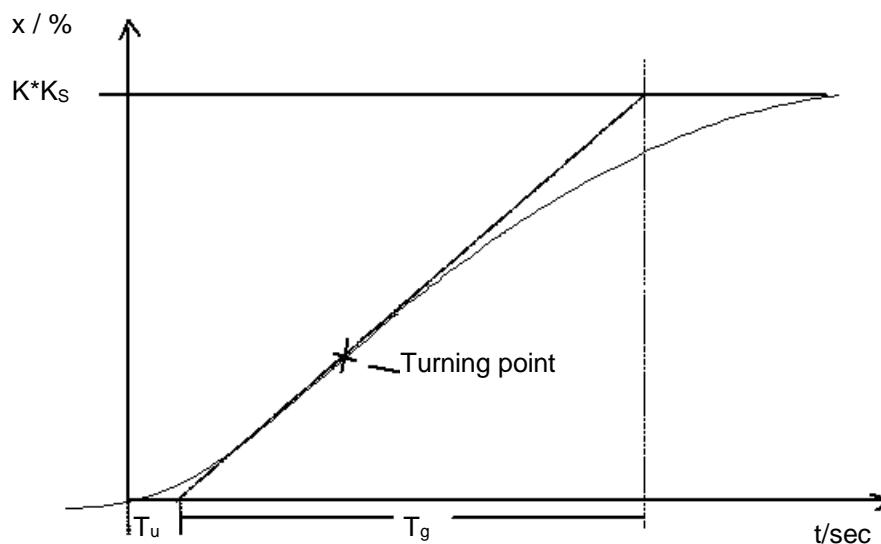


Figure: T_u - T_g -Approximation

4.8.1 Tuning the PI controller according to the Ziegler-Nichols method

Based on experiments on P-T₁-T_L systems, Ziegler and Nichols have identified the following optimal controller adjustments for fixed setpoint control:

$$K_{PR} = 0.9 \frac{T_g}{K_s T_u}$$

$$T_N = 3.33 T_u$$

Use of these tuning values generally results in very good response to disturbances.

4.8.2 Tuning the PI controller according to the Chien, Hrones and Reswick method

Both the response to disturbances and response to setpoint changes were examined in order to achieve the most favorable controller parameters. Different values are yielded for the two cases. In addition, two different adjustments are specified in each case that meets different control performance requirements.

This resulted in the following adjustments:

- For response to disturbances:

Aperiodic transient reaction
with the shortest duration

$$K_{PR} = 0.6 \quad \frac{T_g}{K_S T_u}$$

$$T_N = 4 T_u$$

20 % overshoot minimum
oscillation period

$$K_{PR} = 0.7 \quad \frac{T_g}{K_S T_u}$$

$$T_N = 2.3 T_u$$

- For response to setpoint changes:

Aperiodic transient reaction
with the shortest duration

$$K_{PR} = 0.35 \quad \frac{T_g}{K_S T_u}$$

$$T_N = 1.2 T_g$$

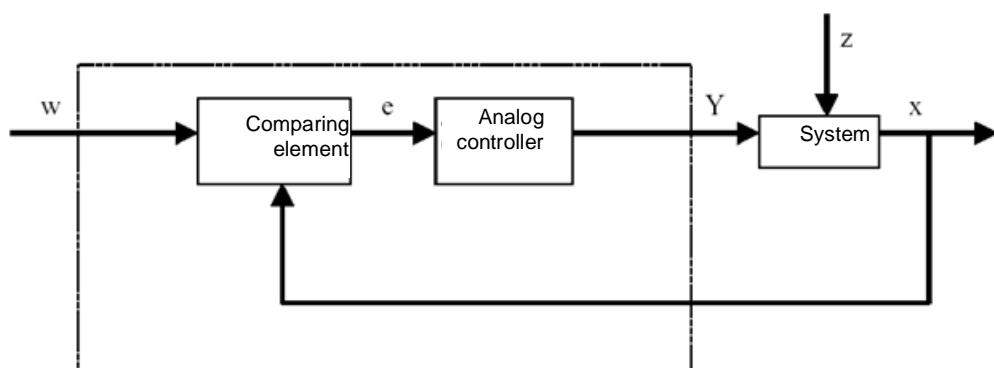
20 % overshoot minimum
oscillation period

$$K_{PR} = 0.6 \quad \frac{T_g}{K_S T_u}$$

$$T_N = T_g$$

4.9 Digital controllers

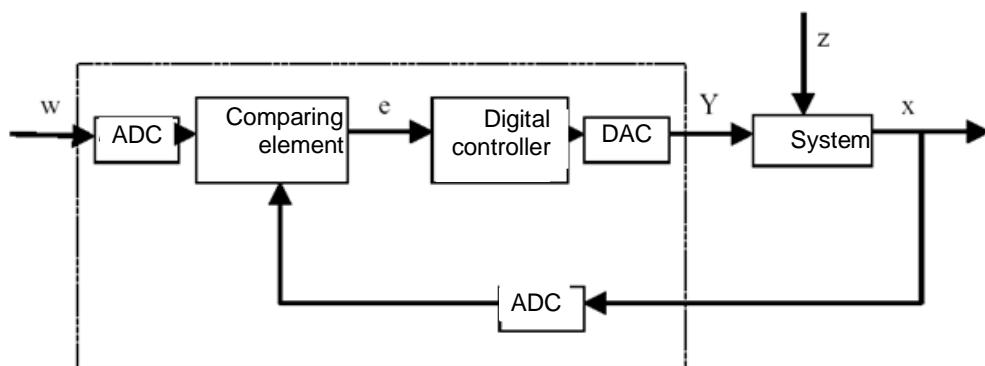
Up to now, the main focus was on analog controllers, in other words, controllers that use the system error, which exists as an analog value, to derive the controller output variable in an analog manner. The diagram of this type of control loop is now well-known:



Often, however, it is advantageous to perform the actual evaluation of the system error digitally. For one thing, the relationship between the system error and controller output variable can be defined much more flexibly when it can be defined by an algorithm or formula that can be used in each case to program a computer than when it has to be implemented in the form of an analog circuit. For another, digital technology enables significantly greater integration of circuits so that multiple controllers can be accommodated in the smallest space. Finally, by dividing the computing time when there is a sufficient amount of computing capacity, it is even possible to use an individual computer as a controller for multiple control loops.

To enable digital processing of the variables, both the reference variable and the feedback variable are first converted to digital values in an analog-to-digital converter (ADC). These are then subtracted from one another by a digital comparing element and the difference is passed to the digital controlling element. Its controller output variable is then converted back to an analog value in a digital-to-analog converter (DAC). From the outside, the combined unit of converters, comparing element and controlling element resembles an analog controller.

We will examine the structure of a digital controller based on a diagram:



The advantages resulting from digital implementation of the controller are accompanied by various problems. For this reason, the size of some variables related to the digital controller must be chosen large enough to prevent the accuracy of the closed loop control from suffering too much from digitization.

Quality criteria for digital computers are:

- The quantization resolution of the digital-to-analog converter

This specifies how fine the continuous value range is digitally mapped. The chosen resolution must be high enough that none of the finer points important for the closed loop control are lost.

- The sampling rate of the analog-to-digital converter.

This is the frequency at which the analog values present at the converter are measured and digitized. This must be high enough that the controller can also still respond to step changes in the controlled variable in a timely manner.

- The cycle time

Unlike an analog closed-loop controller, each digital computer works in clock cycles. The speed of the utilized computer must be high enough that a significant change of the controlled variable cannot occur during a single clock cycle (in which the output value is calculated and no input value is queried).

The performance of the digital controller must be high enough that its response is apparently as prompt and precise as an analog controller.

5 Task

In this chapter, a PID controller for speed control will be added to the program from chapter Analog. The call-up of the "MOTOR_SPEEDCONTROL" [FC10] function must be deleted for this.

6 Planning

The PID_Compact technology object is available in the TIA Portal for closed loop controls.

For closed-loop control of the motor speed, this technology object replaces the "MOTOR_SPEEDCONTROL" [FC10] block.

This will be carried out as an expansion of the "032-500_Analog_Values_" project. This project must be retrieved from the archive beforehand.

The call-up of the "MOTOR_SPEEDCONTROL" [FC10] function must be deleted in the "Main" [OB1] organization block before the technology object can be called and connected in a cyclic interrupt OB.

The PID_Compact technology object must then be configured and commissioned.

6.1 PID_Compact closed-loop control block

The PID_Compact technology object provides a PID controller with integrated tuning for proportional-action final controlling elements.

The following operating modes are possible:

- Inactive
- Pretuning
- Fine tuning
- Automatic mode
- Manual mode
- Substitute output value with error monitoring

Here, the connection, parameter assignment and commissioning of this controller will be for automatic mode

During commissioning we will use the integrated tuning algorithms and record the control response of the controlled system.

The PID_Compact technology object is always called from a cyclic interrupt OB whose fixed set cycle time is 50 ms here.

The speed setpoint is set as a constant at the "Setpoint" input of the PID_Compact technology object in revolutions per minute (range: +/- 50 rpm). The data type is 32-bit floating-point number (Real).

The actual speed value -B8 (sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm) will be entered at the "Input_PER" input.

The output of the controller "Output_PER" will then be connected directly with signal -U1 (manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm).

The controller will only be active as long as output -Q3 (conveyor motor -M1 variable speed) is set. If this is not set, the controller will be deactivated by connection of the "Reset" input.

6.2 Technology diagram

Here you see the technology diagram for the task.

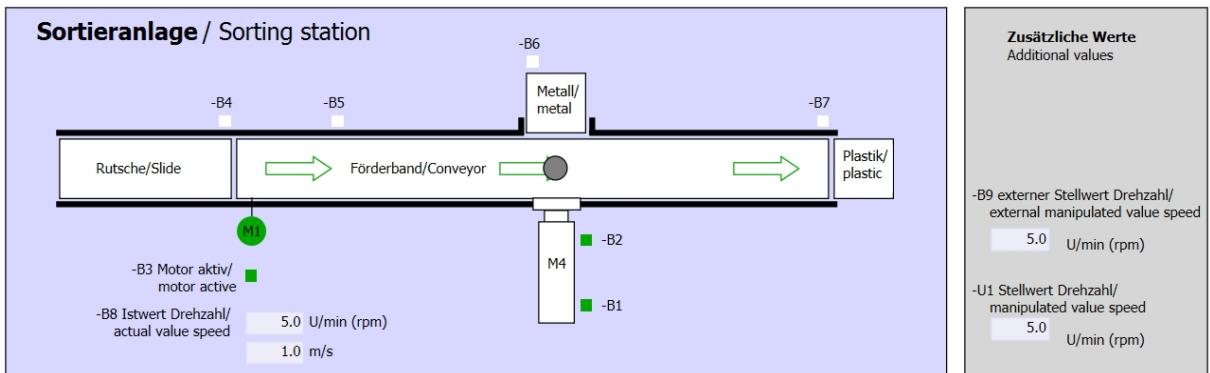


Figure 1: Technology diagram



Figure 2: Control panel

6.3 Reference list

The following signals are required as global operands for this task.

DI	Type	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop OK	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
IW64	BOOL	-B8	Sensor actual value speed of the motor +/-10V corresponds to +/- 50 rpm	

DO	Type	Identifier	Function	
Q 0.2	BOOL	-Q3	Conveyor motor -M1 variable speed	
QW 64	BOOL	-U1	Manipulated value speed of the motor in 2 directions +/- 10V corresponds to +/- 50 rpm	

Legend for reference list

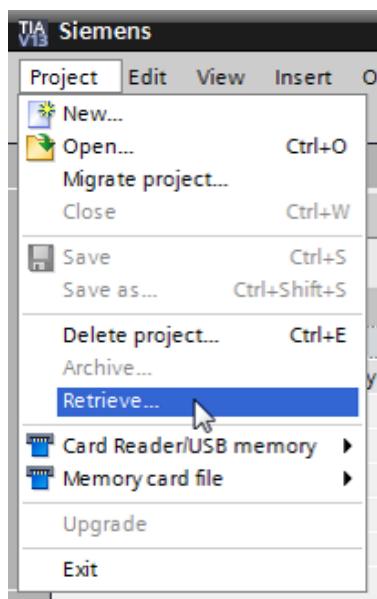
DI	Digital Input	DO	Digital Output
AI	Analog Input	AO	Analog Output
I	Input	Q	Output
NC	Normally Closed		
NO	Normally Open		

7 Structured step-by-step instructions

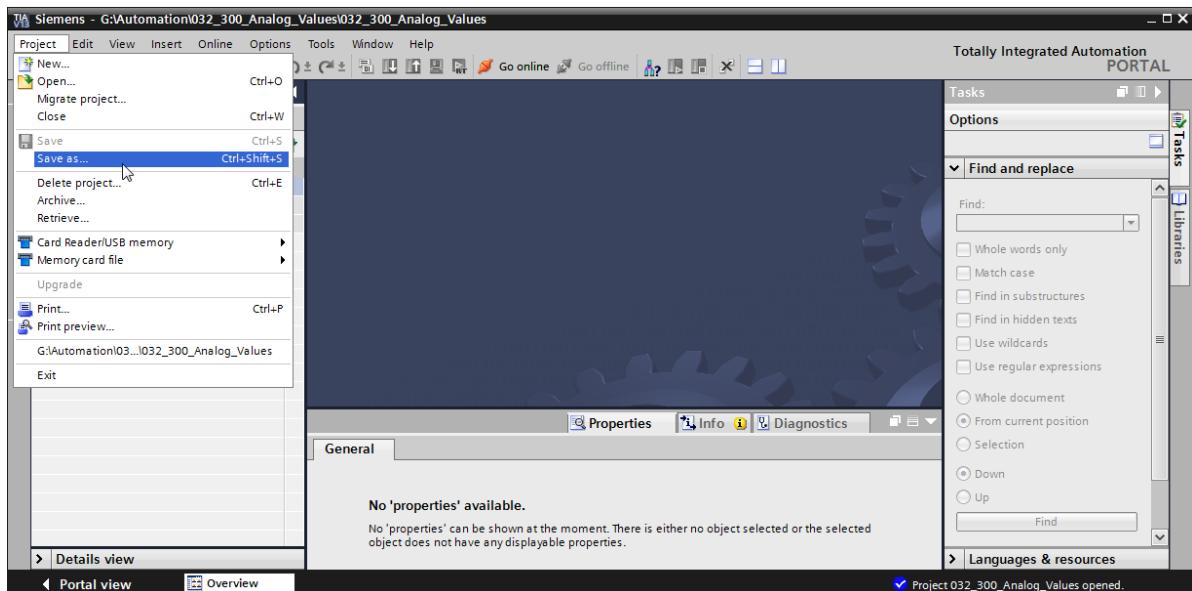
You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it will be sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

- Before we can expand the "SCE_EN_032-500_Analog_Values_R1508.zap13" project from chapter "SCE_EN_032-500_Analog_Values", we must retrieve this project from the archive. To retrieve an existing project that has been archived, you must select the relevant archive with → Project → Retrieve in the project view. Confirm your selection with Open.
- (→ Project → Retrieve → Select a .zap archive → Open)

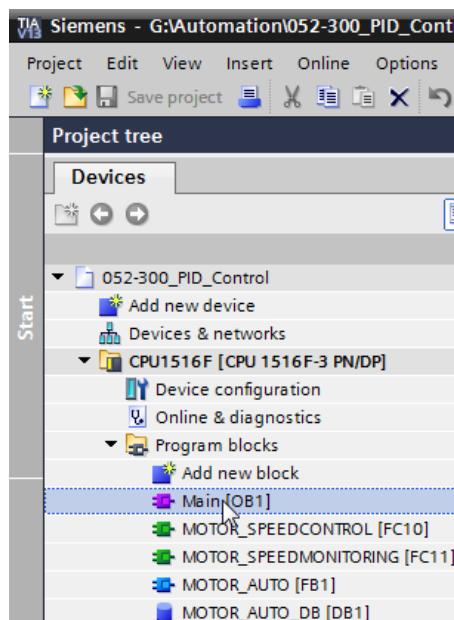


- The next step is to select the target directory where the retrieved project will be stored.
Confirm your selection with "OK".
(→ Target directory → OK)
- Save the opened project under the name 052-300_PID_Controller.
(→ Project → Save as ... → 052-300_PID_Controller → Save)

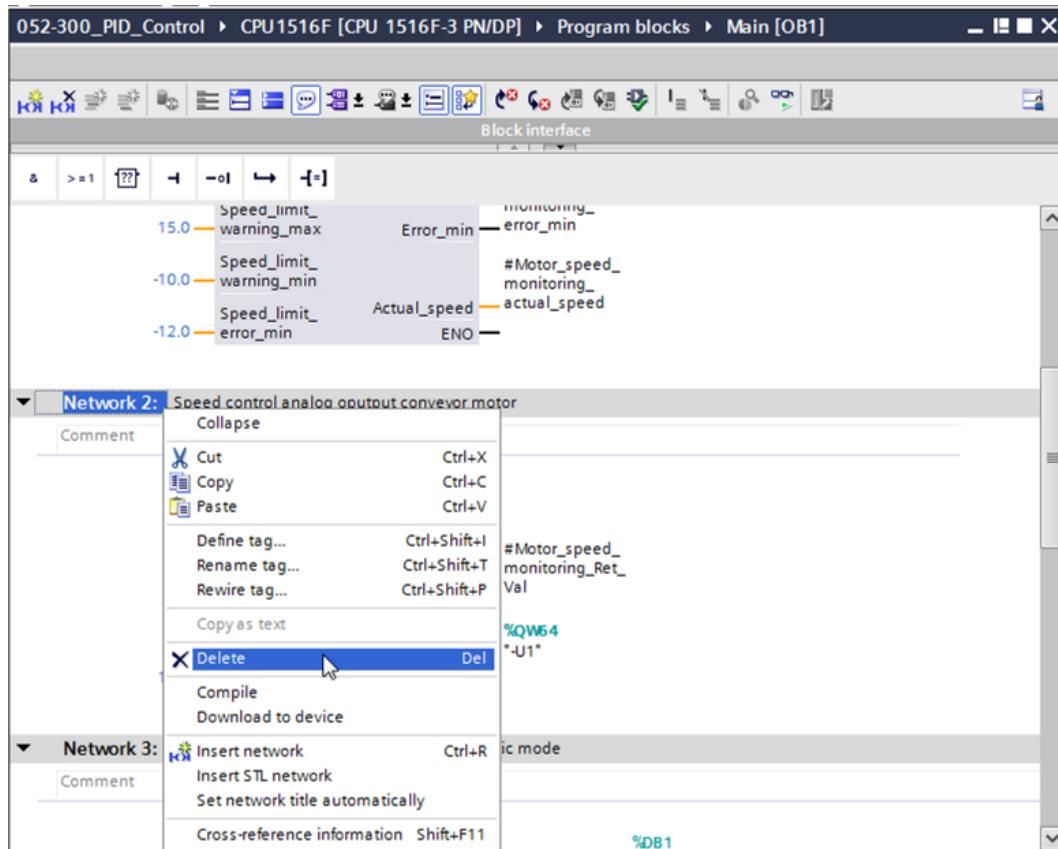


7.2 Call PID_Compact controller in a cyclic interrupt OB

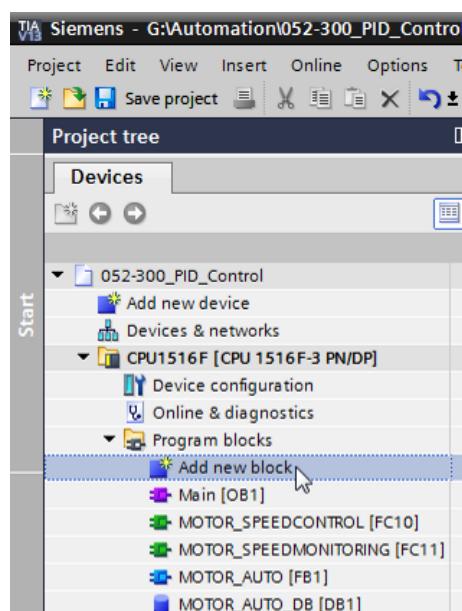
- Open the "Main [OB1]" organization block with a double-click.



- Delete Network 2 with the no longer needed call-up of the "MOTOR_SPEEDCONTROL" [FC10] function.
(→ Network 2 → Delete)

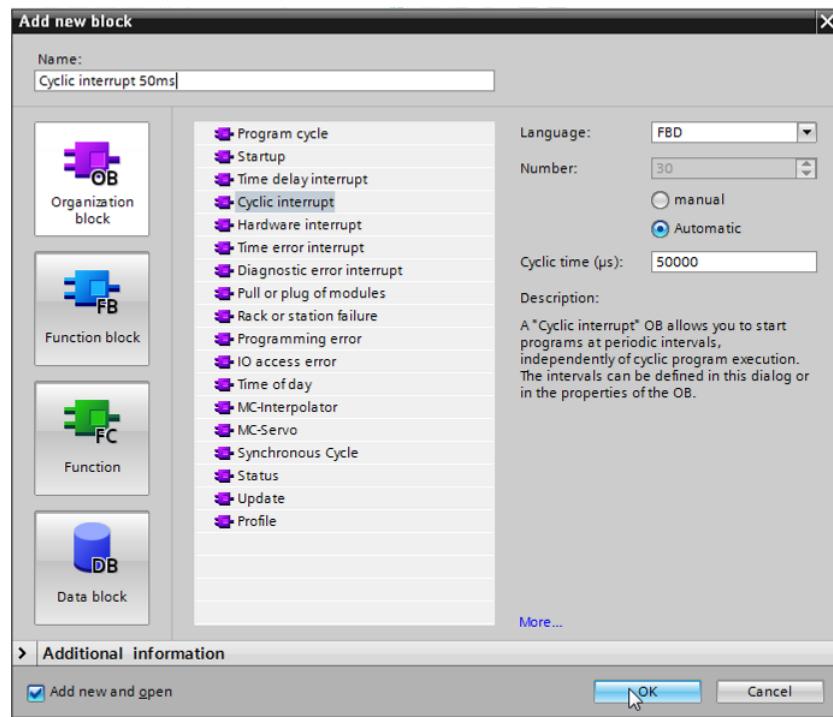


- We need a cyclic interrupt OB for calling the PID_Compact controller. Therefore, select the 'Add new block' item in the Program blocks folder.
- (→ Program blocks → Add new block)

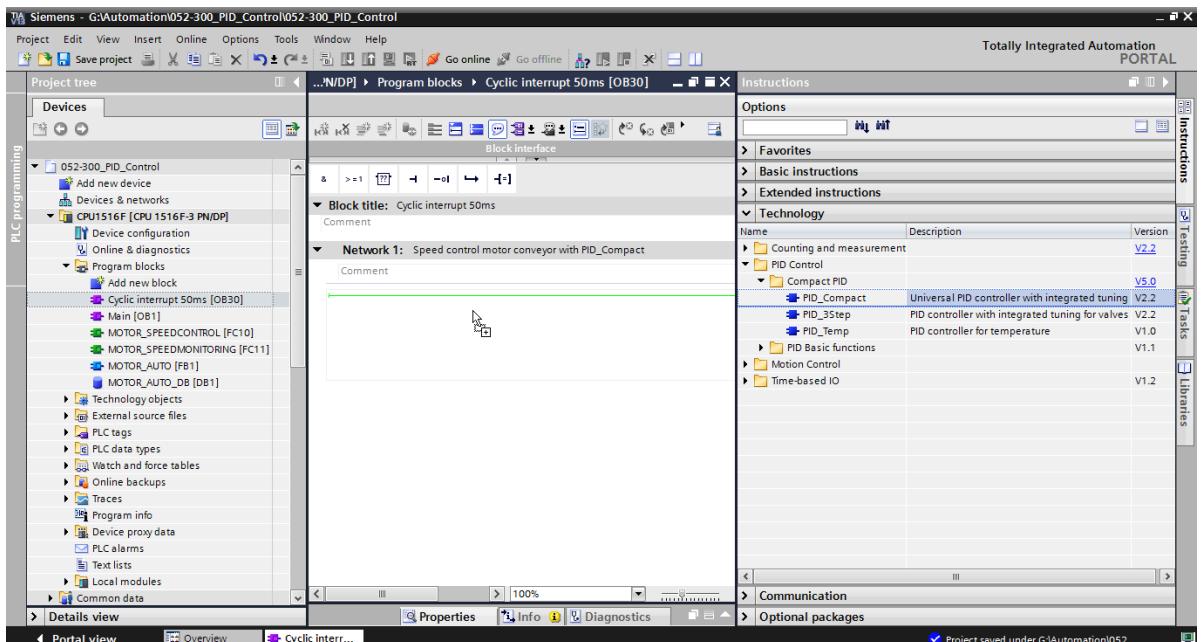


- Select in the next dialog and rename the cyclic interrupt OB to: "Cyclic interrupt 50ms". Set the language to FBD and assign "50000 µs" as the cyclic time. Select the "Add new and open" check box. Click "OK".

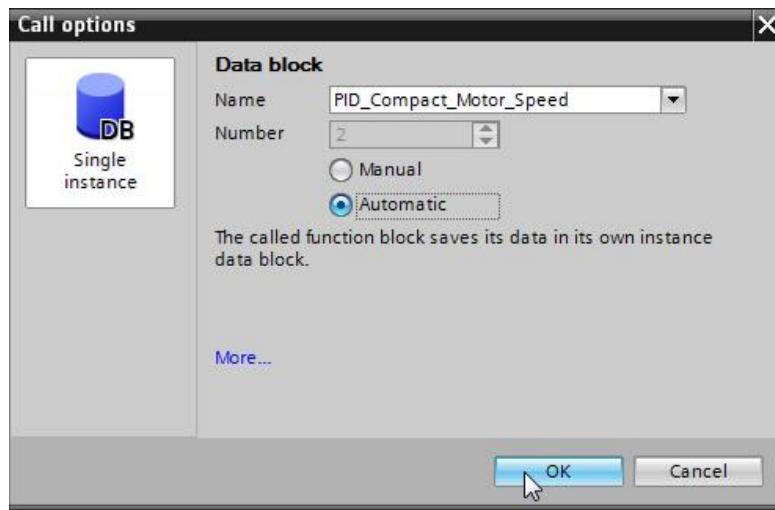
- OB → Name: Cyclic interrupt 50ms → Language: FBD → Cyclic time (ms): 50000 →
 Add new and open → OK)



- The block is then directly opened. Enter meaningful comments and move the 'PID_Compact' technology object to Network 1 using drag-and-drop.
 (→ Technology → PID Control → Compact PID → PID_Compact)

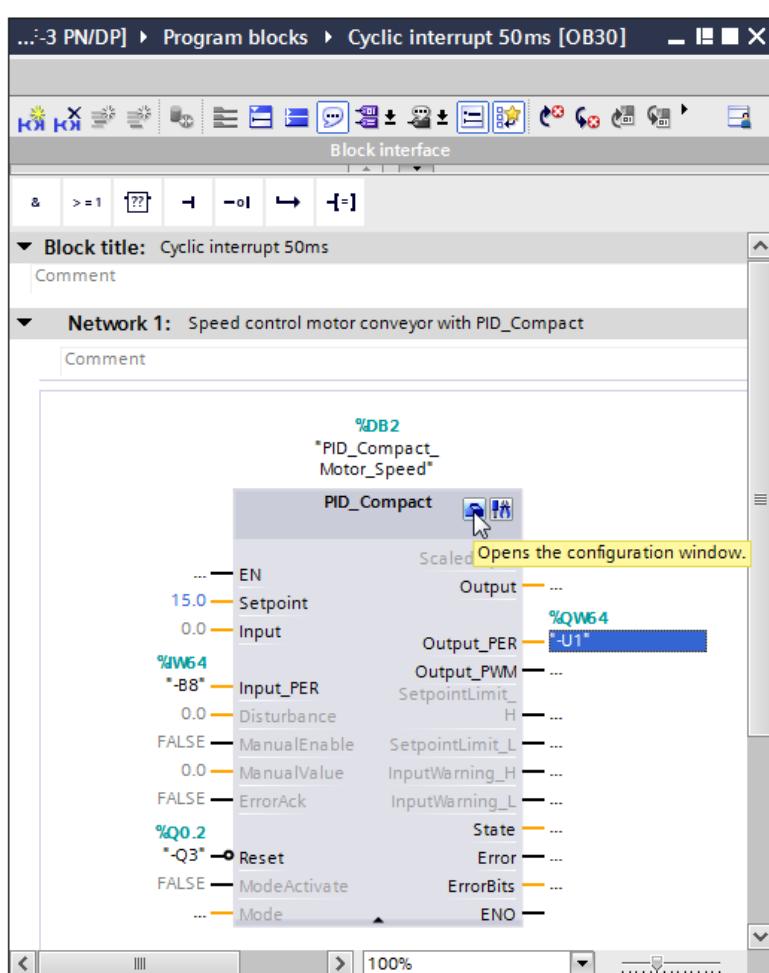


- Assign a name for the instance data block and apply it with OK.
 (→ PID_Compact_Motor_Speed → OK)



- Expand the view of the block by clicking the '▲' arrow. Interconnect this block as shown here with setpoint (constant: 15.0), actual value (global tag "-B8"), manipulated variable (global tag "-U1") and Reset input for deactivating the controller (global tag "-Q3"). Negate the 'Reset' input. The configuration mask '↙' of the controller can then be opened.

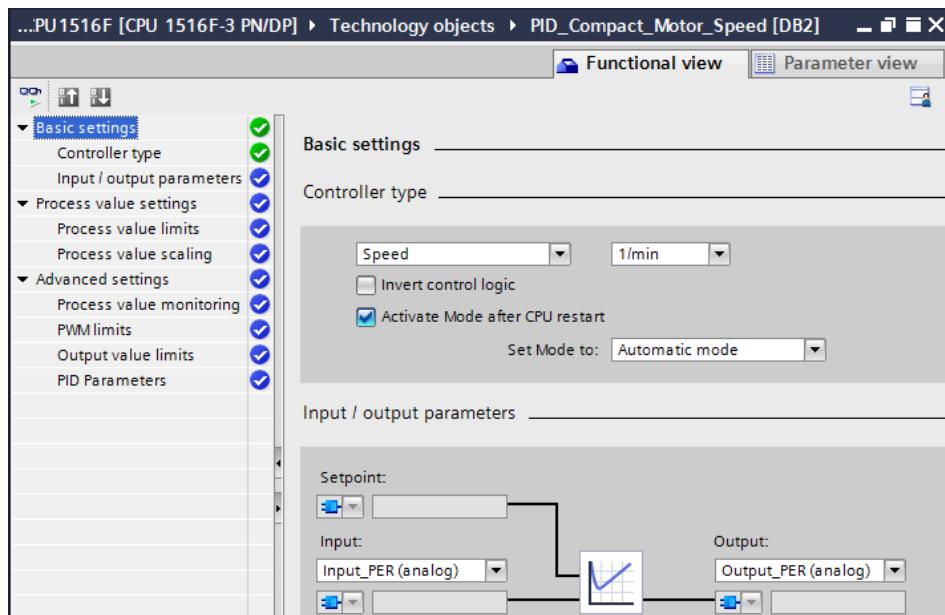
(→ ▲ → 15.0 → "-B8" → "-U1" → -Q3 → $\neg\text{Q3}$ → ↗)



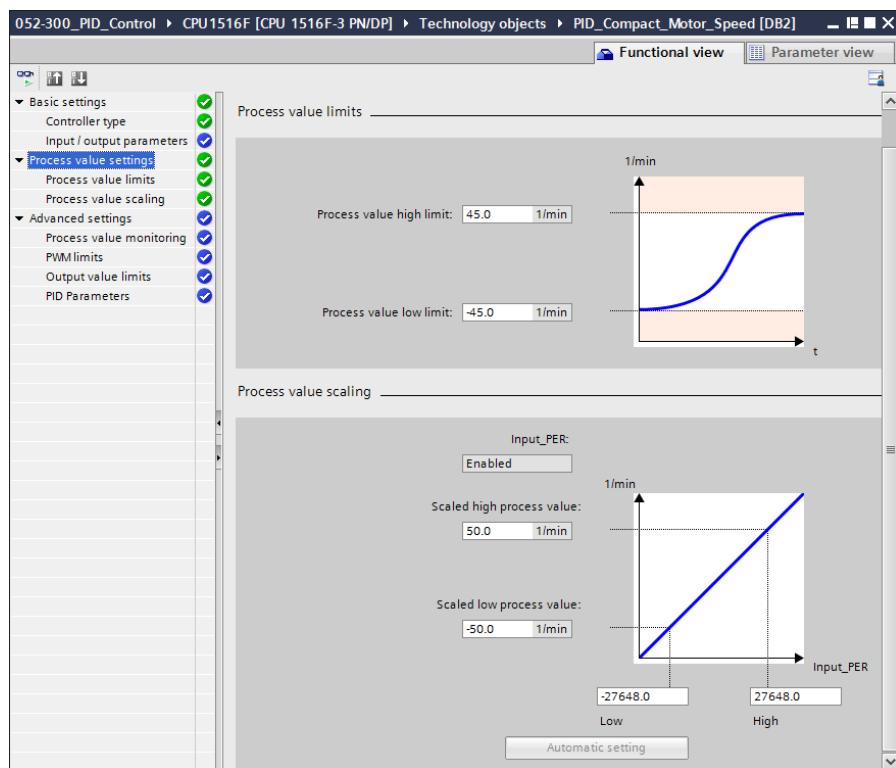
- There are 2 views for configuration of the controller: Parameter view and Functional view.
Here we will use the easier-to-understand 'Functional view'.
(→ Functional view)

Name in functional view	Name in DB	Start value project	Minimum value	Maximum value	Comment
Physical quantity	PhysicalQuantity	Speed			Selection of physical quantity.
Unit of measurement	PhysicalUnit	1/min			Selection of unit of measurement.
	PhysicalUnit	0			Selection of unit of measurement.
Invert control logic	..InvertControl	FALSE			Enables inversion of control logic.
Activate Mode after CPU restart	RunModeByStartup	TRUE			Activates the operating mode selected after a power failure.
Set Mode to	Mode	Automatic mode	0	4	Selection of operating mode.
	Mode	3			Selection of operating mode.

- In the 'Basic settings', the 'Controller type' and the interconnection of the 'Input / output parameters' are entered. Set the values as shown here.
(→ Basic settings → Controller type → Input / output parameters)

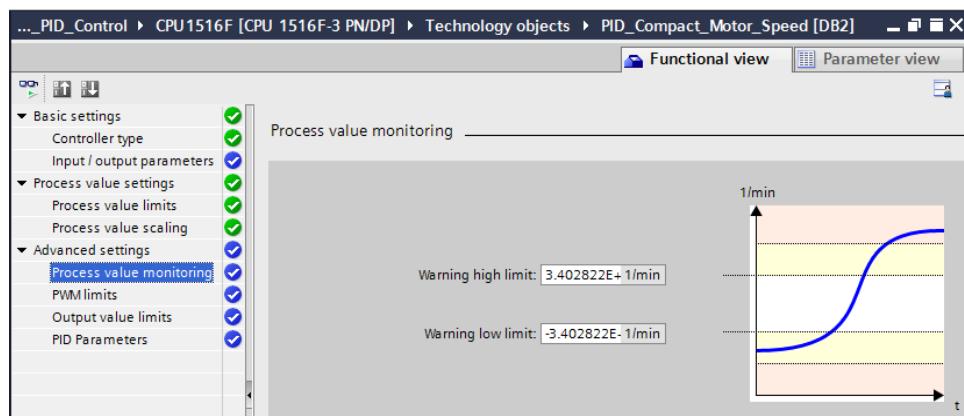


- In 'Process value settings' we scale to the range +/- 50 rpm and define the 'Process value limits' of +/- 45 rpm.
(→ Process value settings → Process value limits → Process value scaling)



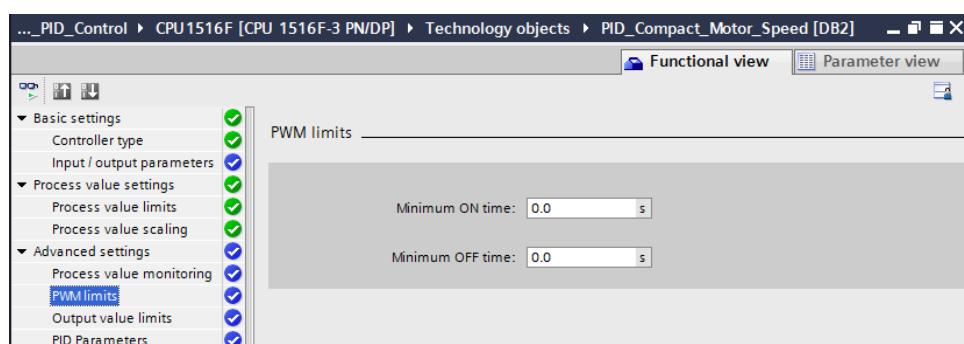
- In the 'Advanced settings', a process value monitoring would be possible but we don't want to deal with that here.

(→ Advanced settings → Process value monitoring)

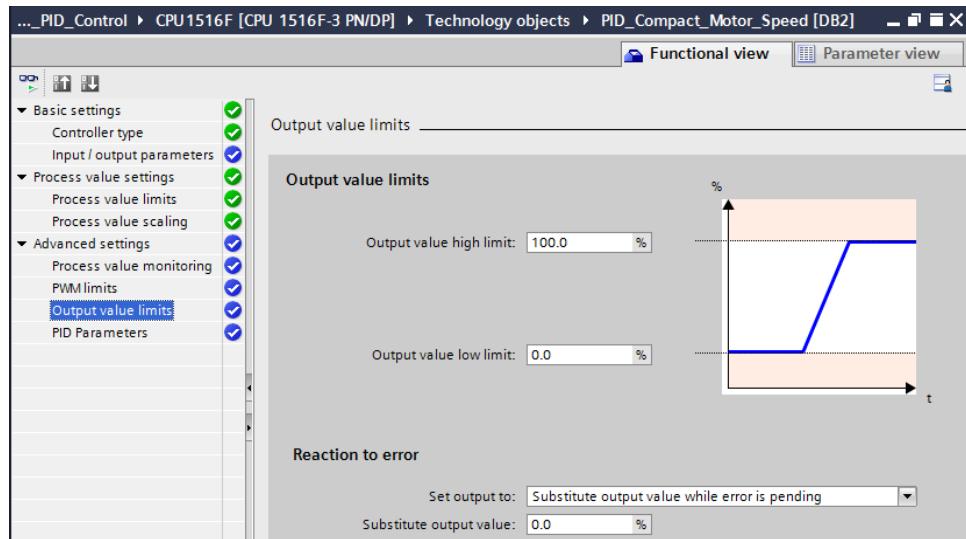


- In the 'Advanced settings' for 'PWM' (pulse width modulation), we will leave the default values since the output for this is not needed in our project.

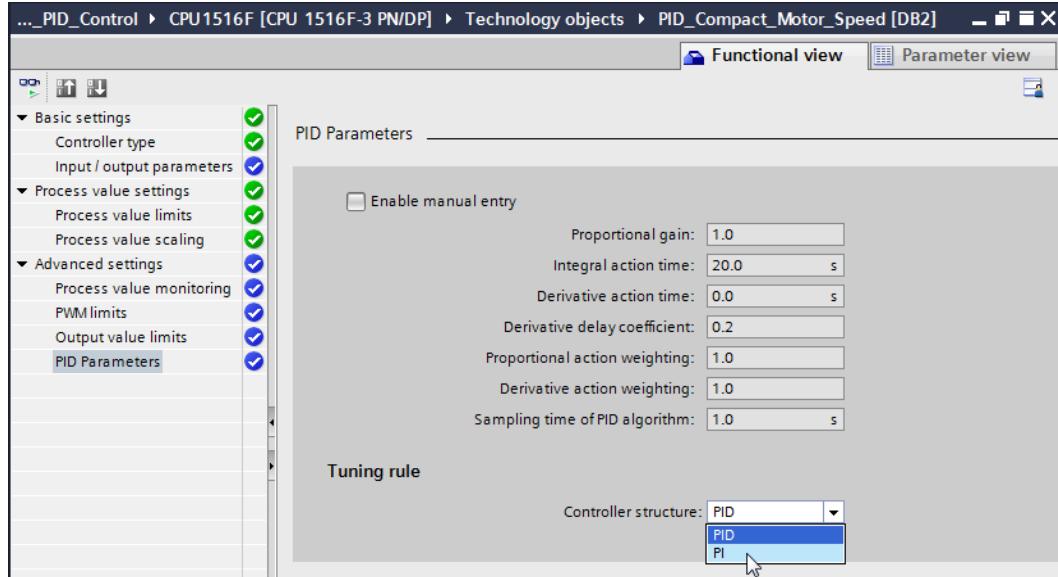
(→ Advanced settings → PWM)



- In the 'Advanced settings', we define the 'Output value limits' of 0.0 % to 100.0 %.
 (→ Advanced settings → Output value limits)

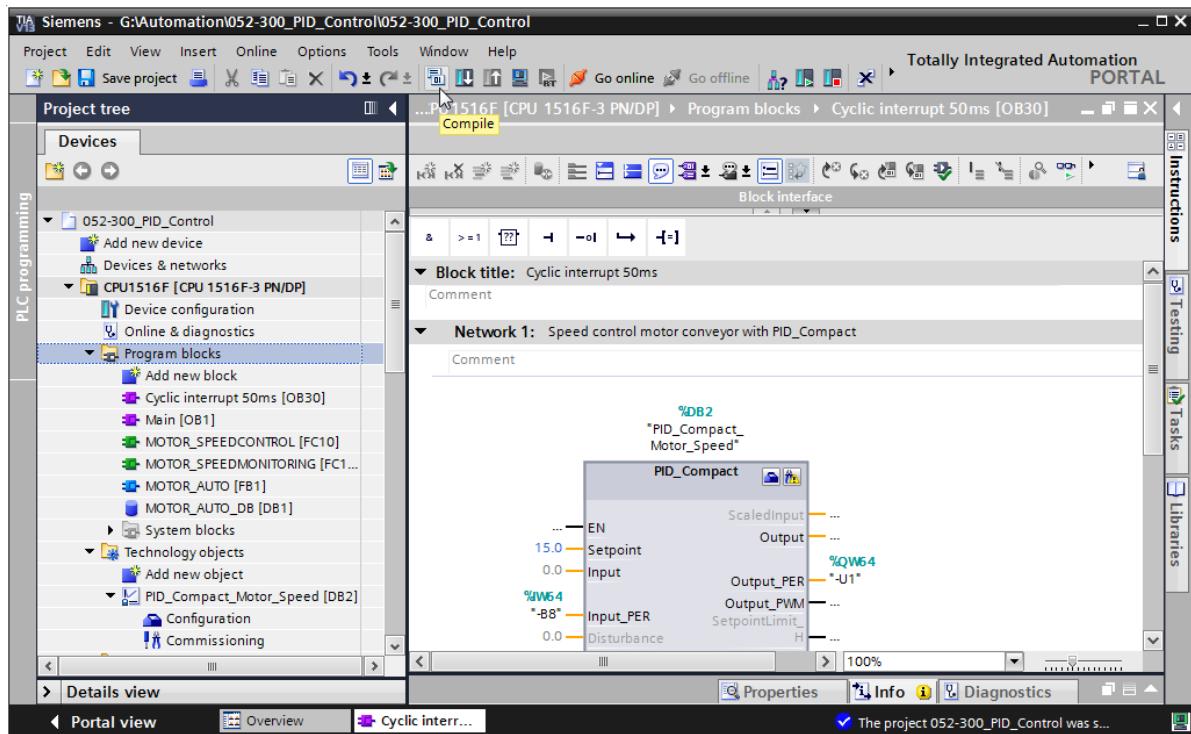


- In the 'Advanced settings', you will now also find a manual setting of the 'PID parameters'. Once we have changed the controller structure to 'PI', the configuration window is closed by clicking **X** and we receive a finished product with a functional PID controller. This should, however, still be commissioned and tuned online during operation.
- (→ Advanced settings → PID Parameters → Controller structure: PI → **X**)



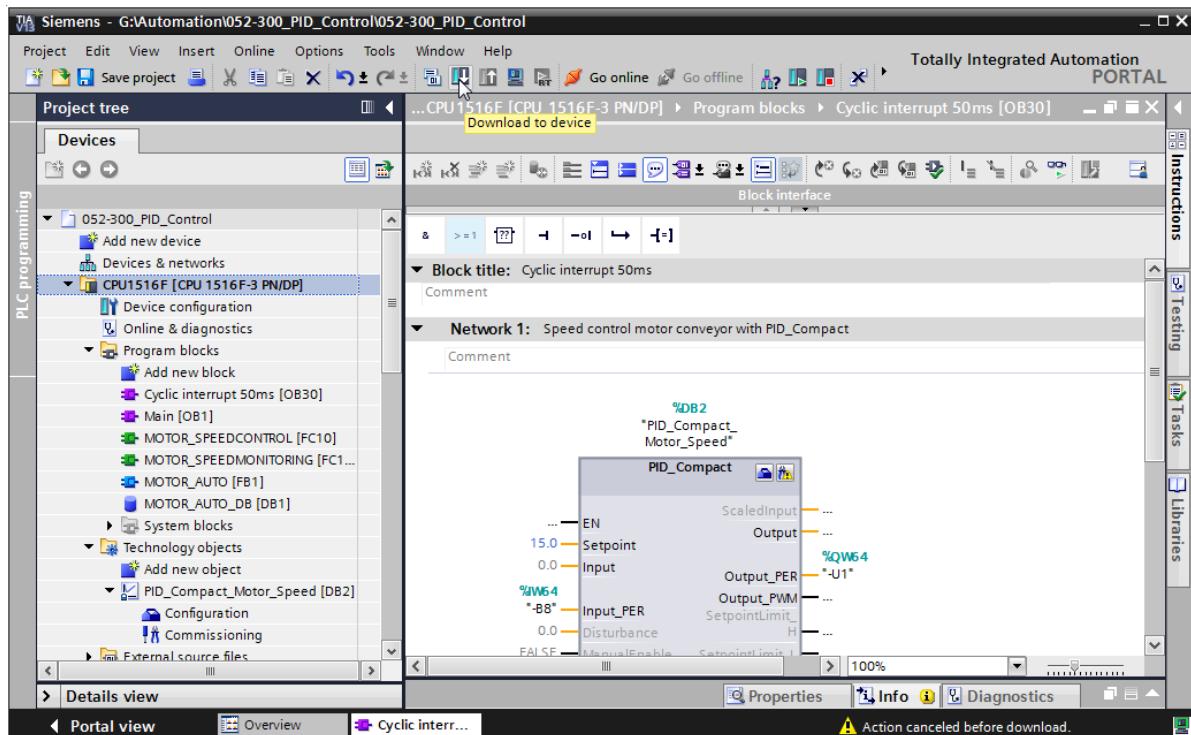
7.3 Save and compile the program

- To save your project, click the **Save project** button in the menu. To compile all blocks, click the "Program blocks" folder and select the **Compile** icon for compiling in the menu.
 (→ **Save project** → Program blocks → **Compile**)



7.4 Download the program

- After successful compilation, the complete controller with the created program including the hardware configuration can, as described in the previous modules, be downloaded.



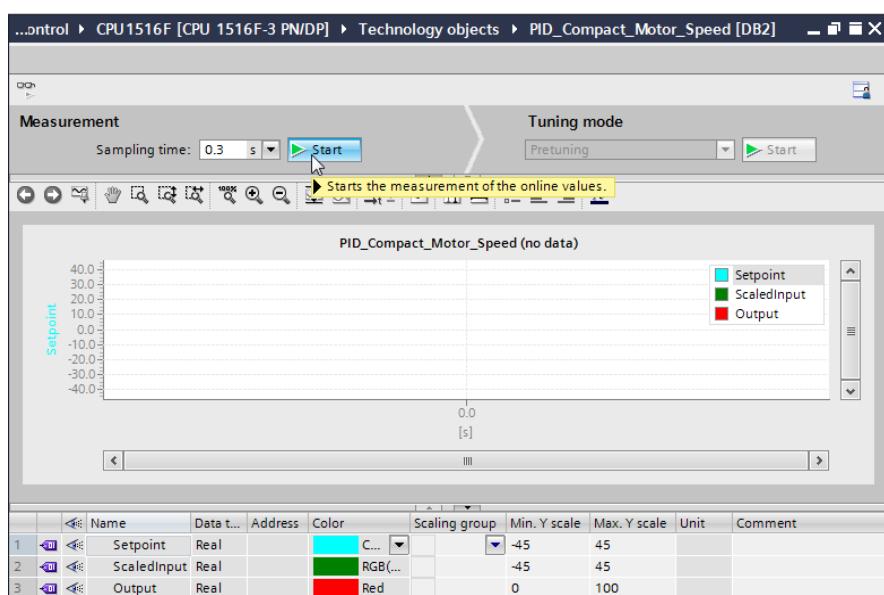
7.5 Monitor PID_Compact

→ Click the Monitoring on/off icon  to monitor the state of the blocks and tags when testing the program. At the first start of the CPU, however, the 'PID_Compact' controller is not yet tuned. We still have to start the tuning by clicking the ' Commissioning' icon.

(→ Cyclic interrupt 50ms [OB30] →  → PID_Compact → 

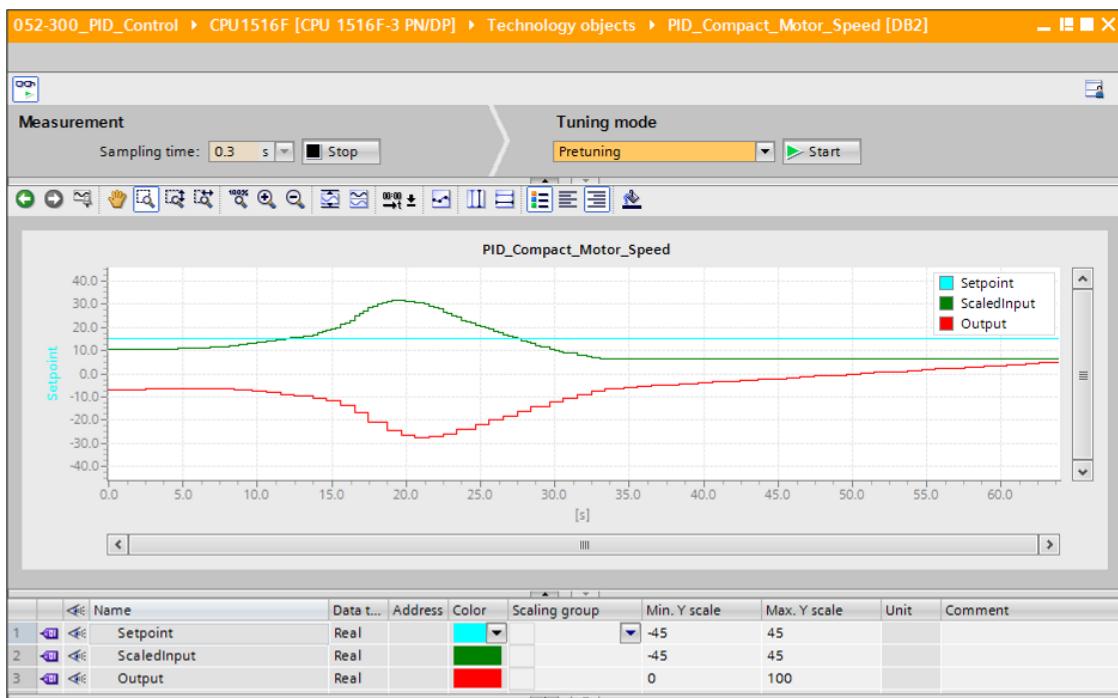
→ If we click  under 'Measurement', the values of the setpoint (Setpoint), actual value (ScaledInput) and manipulated variable (Output) can be displayed and monitored in a diagram.

(→ )



→ The measurement can be stopped again by clicking .

(→ )



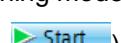
7.6 PID_Compact pretuning

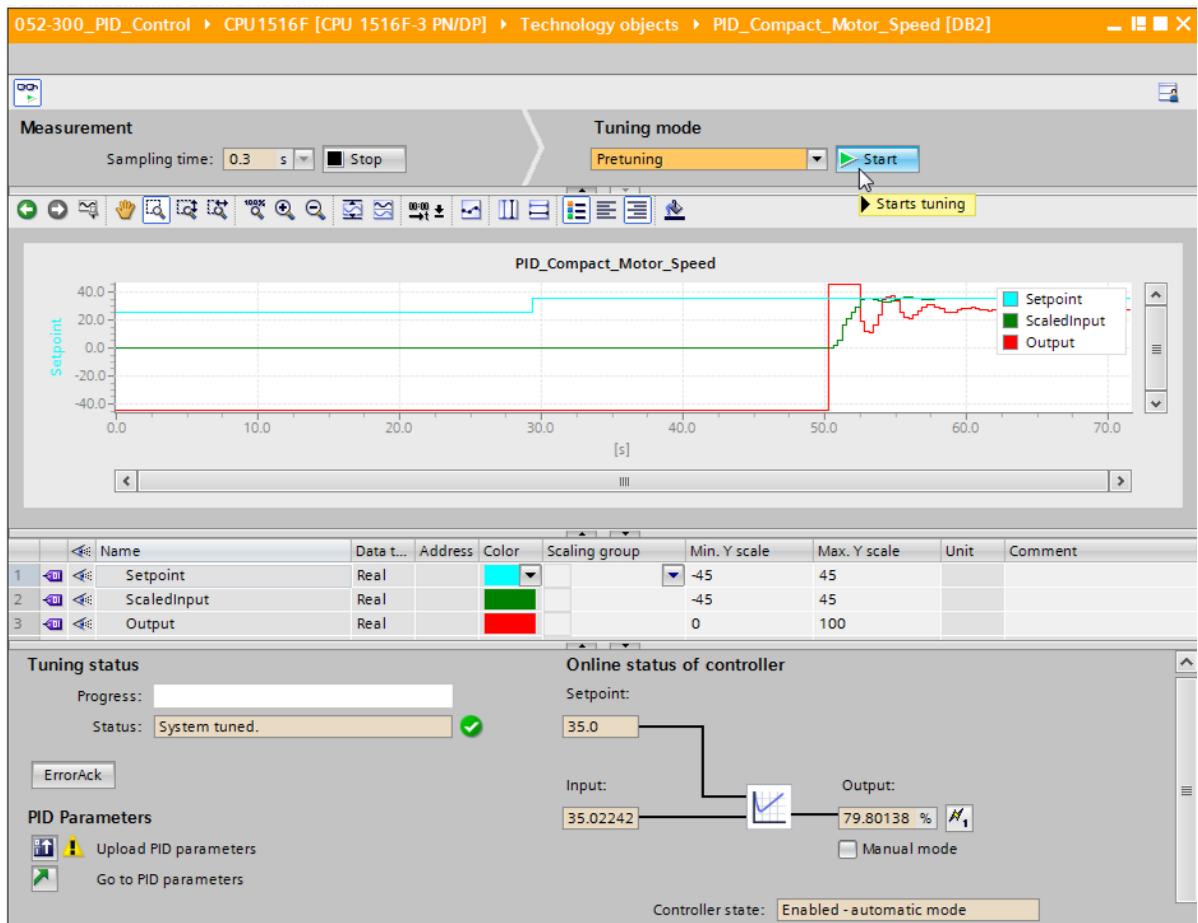
The pretuning determines the process response to a step change of the output value and searches for the turning point. The PID parameters are calculated from the maximum slope and the dead time of the controlled system. The optimal PID parameters are obtained when you perform pretuning and fine tuning.

The more stable the actual value is, the easier and more accurately the PID parameters can be determined. Actual value noise is acceptable as long as the actual value rise is significantly greater than the noise. This is most likely the case in "Inactive" or "Manual mode" operating mode. The PID parameters are backed up before they are recalculated.

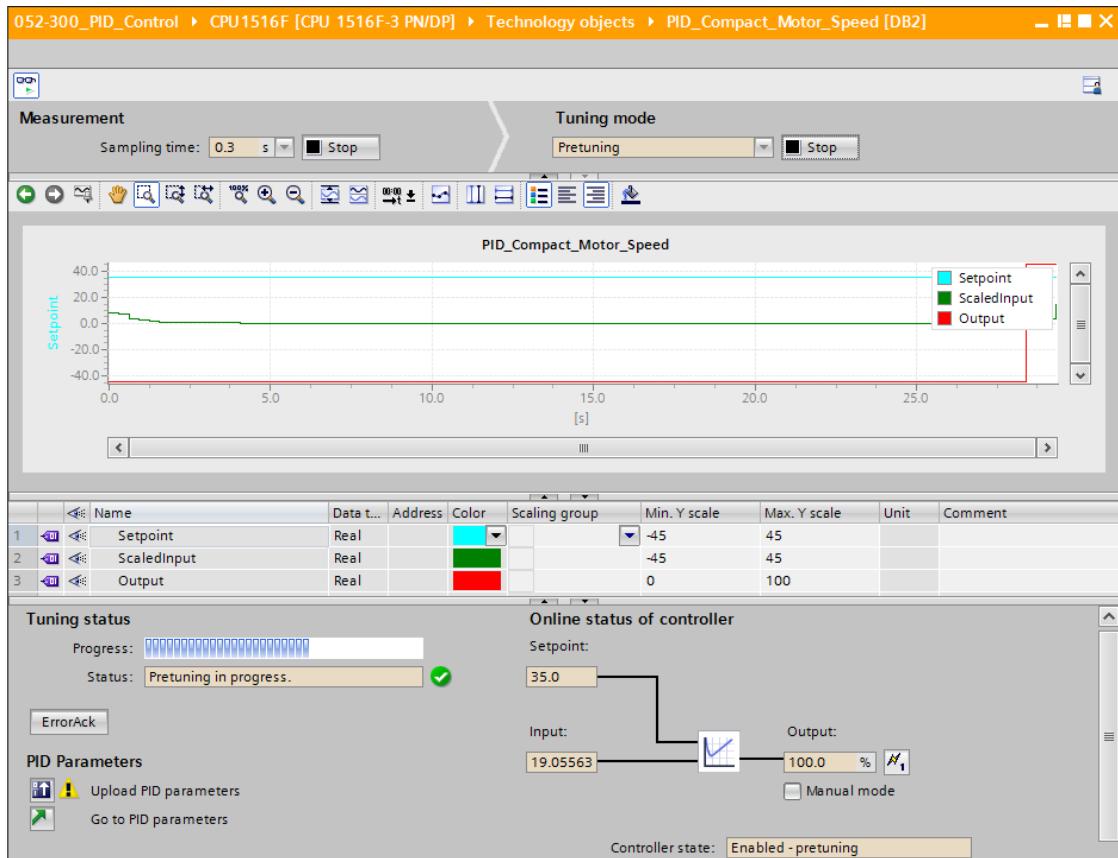
The following requirements must be met:

- The "PID_Compact" instruction is called in a cyclic interrupt OB.
 - ManualEnable = FALSE
 - Reset = FALSE
 - PID_Compact is in "Manual mode", "Inactive" or "Automatic mode" operating mode.
 - The setpoint and actual value are within the configured limits (see "Process value monitoring" configuration).
 - The difference between setpoint and actual value is greater than 30 % of the difference between the process value high limit and low limit.
 - The difference between setpoint and actual value is > 50 % of the setpoint.
- 'Pretuning' is selected as the 'Tuning mode' and this is then started.

(→ Tuning mode → Pretuning → )



- The pretuning starts. The current work steps and any errors that occur are shown in the "Tuning status" field. The progress bar shows the progress of the current work step.



7.7 PID_Compact fine tuning

The fine tuning generates a constant, limited oscillation of the actual value. The PID parameters are optimized for the operating point based on the amplitude and frequency of this oscillation. All PID parameters are recalculated from the results. The PID parameters resulting from fine tuning generally produce a better response to setpoint changes and disturbances than the PID parameters from pretuning. The optimal PID parameters are obtained when you perform pretuning and fine tuning.

PID_Compact automatically attempts to generate an oscillation that is greater than the actual value noise. The fine tuning is influenced only slightly by the stability of the actual value. The PID parameters are backed up before they are recalculated.

The following requirements must be met:

- The "PID_Compact" instruction is called in a cyclic interrupt OB.
- ManualEnable = FALSE
- Reset = FALSE
- The setpoint and actual value are within the configured limits.
- The control loop is stable at the operating point. The operating point is reached when the actual value is equal to the setpoint.
- No disturbances are expected.
- PID_Compact is in "Manual mode", "Inactive" or "Automatic mode" operating mode.

The fine tuning runs as follows when started in automatic mode:

When you want to improve the existing PID parameters by tuning them, start the fine tuning from automatic mode.

PID_Compact uses the existing PID parameters for controlling until the control loop is stable and the requirements for fine tuning are met. Only then does the fine tuning start.

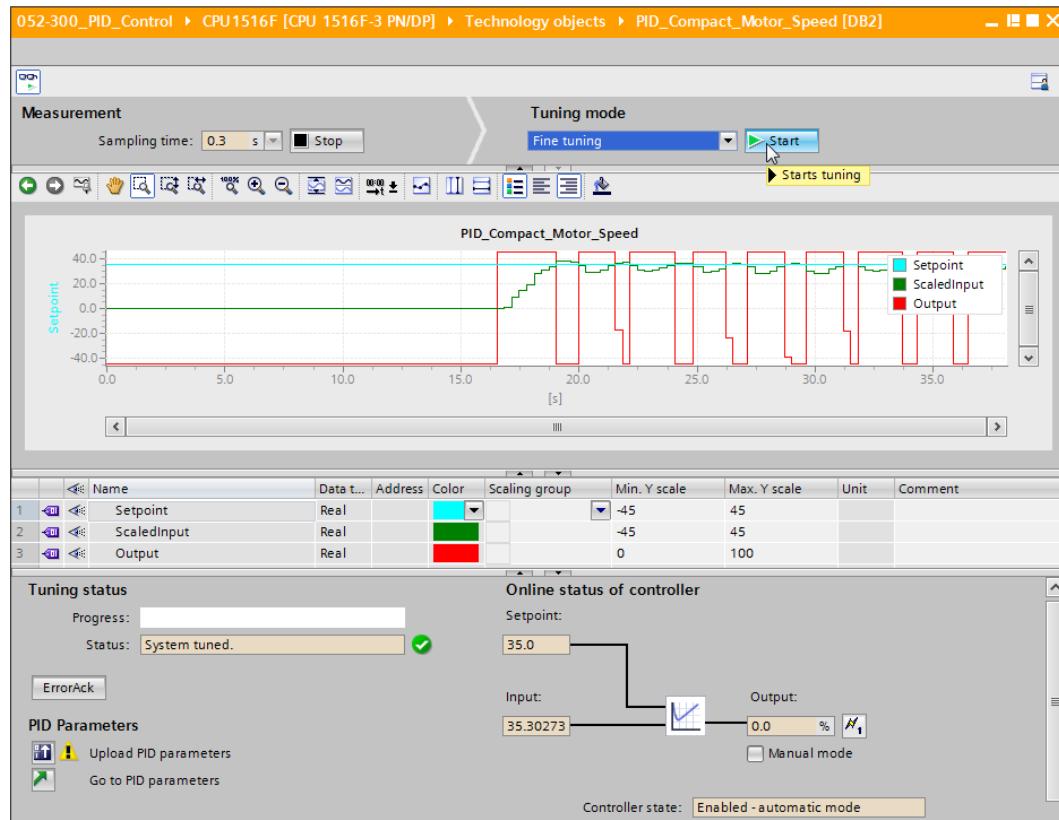
The fine tuning runs as follows when started in inactive or manual mode:

When the requirements for pretuning are met, pretuning is started. PID_Compact uses the determined PID parameters for controlling until the control loop is stable and the requirements for fine tuning are met. Only then does the fine tuning start. If pretuning is not possible, PID_Compact responds as configured in Response to error.

If the actual value is already too close to the setpoint for pretuning, an attempt is made to reach the setpoint with minimum or maximum output value. This can cause increased overshoot.

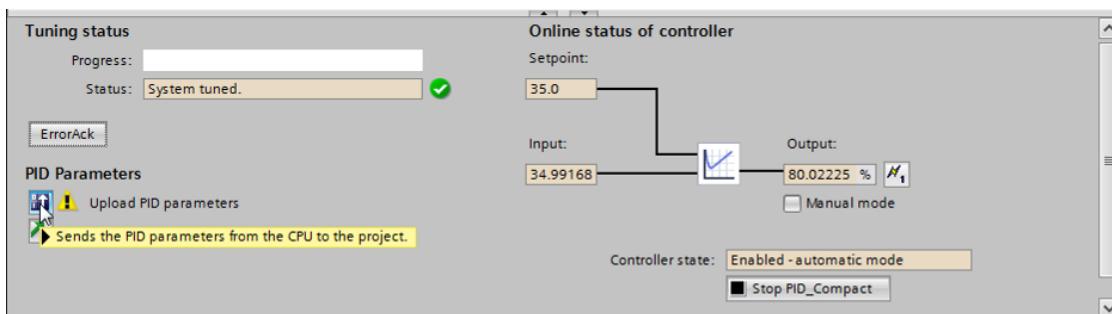
→ 'Fine tuning' is selected as the 'Tuning mode' and this is then started.

(→ Tuning mode → Fine tuning → Start)



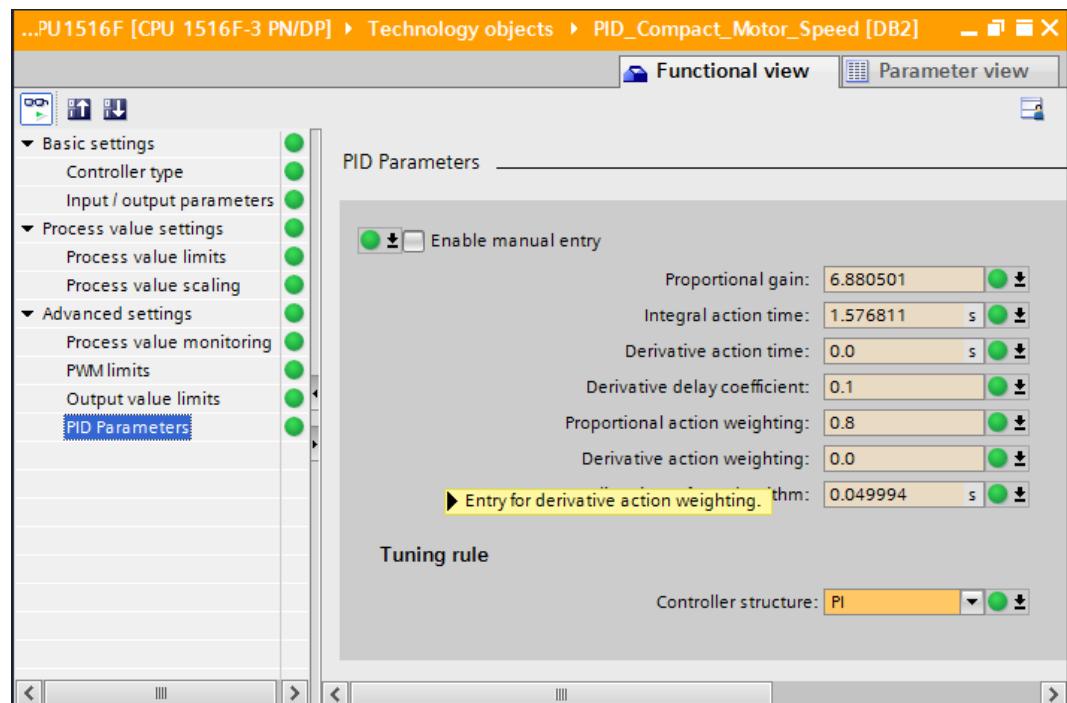
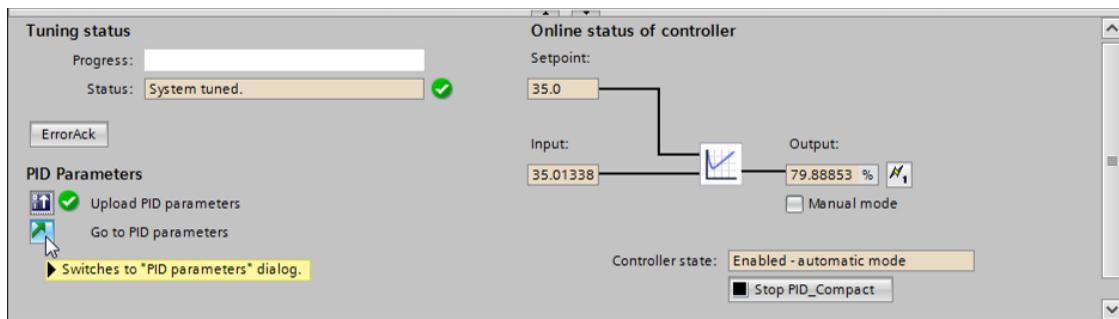
→ The fine tuning starts. The current work steps and any errors that occur are shown in the "Tuning status" field. If the self-tuning was completed without error message, the PID parameters have been tuned. The PID controller switches to automatic mode and uses the tuned parameters. The tuned PID parameters are retained at a Power ON and restart of the CPU. You can download the PID parameters from the CPU to your project with the button.

(→



→ The PID parameters in the configuration can be displayed by clicking .

(→



- As the final step, the online connection should be disconnected and the complete project should be saved.

(→ Go offline → Save project)

8 Checklist

No.	Description	Completed
1	Cyclic interrupt OB Cyclic interrupt 50ms [OB30] successfully created.	
2	PID_Compact controller in cyclic interrupt OB Cyclic interrupt 50ms [OB30] called and connected.	
3	Configuration of the PID_Compact controller performed.	
4	Compiling successful and without error message	
5	Download successful and without error message	
6	Pretuning successful and without error message	
7	Fine tuning successful and without error message	
8	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Pushbutton automatic stop not actuated (-S2 = 1) Briefly press the automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) then Conveyor motor -M1 variable speed (-Q3 = 1) switches on and stays on. The speed corresponds to the speed setpoint in the range +/- 50 rpm	
9	Sensor part at end of conveyor activated (-B7 = 1) → -Q3 = 0 (after 2 seconds)	
10	Briefly press the automatic stop pushbutton (-S2 = 0) → -Q3 = 0	
11	Activate EMERGENCY OFF (-A1 = 0) → -Q3 = 0	
12	Manual mode (-S0 = 0) → -Q3 = 0	
13	Switch off station (-K0 = 0) → -Q3 = 0	
14	Cylinder not retracted (-B1 = 0) → -Q3 = 0	
15	Speed > Motor_speed_monitoring_error_max → -Q3 = 0	
16	Speed < Motor_speed_monitoring_error_min → -Q3 = 0	
17	Project successfully archived	



Training Curriculum

Siemens Automation Cooperates with Education | 05/2017

TIA Portal Module 013
Frequency converter G120 on PROFINET
with SIMATIC S7-1500

Table of contents

1	Objective	479
2	Requirement	479
3	Required hardware and software	479
4	Theory	481
4.1	SINAMICS G120 frequency converter	481
4.2	Components for configuring SINAMICS G120	481
4.2.1	Control Units CU250S-2	481
4.2.2	Operator Panels	482
4.2.3	Memory cards for Control Unit (optional)	483
4.2.4	Brake Relay	483
4.2.5	Safe Brake Relay	483
4.2.6	PM240-2 Power Modules	483
4.2.7	PM250 Power Modules	484
4.2.8	Line filter	484
4.2.9	Line reactor	484
4.2.10	Output reactor	485
4.2.11	Sine-wave filter	485
4.2.12	Braking resistor	485
4.3	Parameter assignment of the SINAMICS G120	486
4.3.1	Display parameters	486
4.3.2	Adjustable parameters	486
4.3.3	P0010 Drive commissioning parameter filter	486
4.3.4	P0015 Macro drive unit	487
4.3.5	Changeability depending on the converter state	487
4.3.6	BICO technology	488
4.3.7	Control Data Set (CDS) and Drive Data Set (DDS)	489
4.4	Commissioning of the SINAMICS G120 frequency converter	489
4.4.1	Restoring factory settings through a parameter reset	490
4.4.2	Basic commissioning	490
4.5	PROFINET interface of the SINAMICS G120, CU250S-2 PN Vector	490
4.5.1	Telegrams	491
4.5.2	Assignment of the process data (PZD) for the SINAMICS G120 with Standard Telegram 1	491
4.5.3	Control word 1 (STW1)	491
4.5.4	Status word 1 (ZSW1)	492
4.5.5	Main setpoint (HSW/NSOLL_A; 16-bit)	493
4.5.6	The main actual value (HIW/NIST_A; 16-bit)	493
4.5.7	Layout of the request telegram in double-word format	494

4.5.8	Layout of the response telegram in double-word format	494
4.6	SINAMICS Startdrive commissioning tool for SINAMICS G120	495
4.6.1	Resetting frequency converters and setting the IP address.....	495
5	Task.....	498
6	Planning.....	499
6.1	Technology schematic diagram.....	500
6.2	Reference table	500
7	Structured step-by-step instructions.....	501
7.1	Retrieving an existing project	501
7.2	Creating a frequency converter in the TIA Portal	502
7.3	Assigning parameters with the commissioning wizard	510
7.4	Testing and commissioning with control panel.....	517
7.5	Creating a program for controlling the frequency converter	520
7.6	Loading the program in SIMATIC S7 CPU 1516F-3 PN/DP	528
7.7	Diagnostics of SIMATIC S7 CPU 1516F-3 PN/DP	529
7.8	Diagnostics with SINAMICS Startdrive	530
7.9	Checklist.....	533

Frequency Converter G120 with Control Unit CU250S-2 PN Vector on PROFINET with SIMATIC S7-1500

1 Objective

In this chapter you learn how a frequency converter SINAMICS G120 with the Control Unit CU250S-2 PN Vector and together with a CPU1516F-3 PN/DP on PROFINET is put into operation.

The module explains the basic commissioning of the frequency converter SINAMICS G120 with the SINAMICS Startdrive software in the TIA Portal.

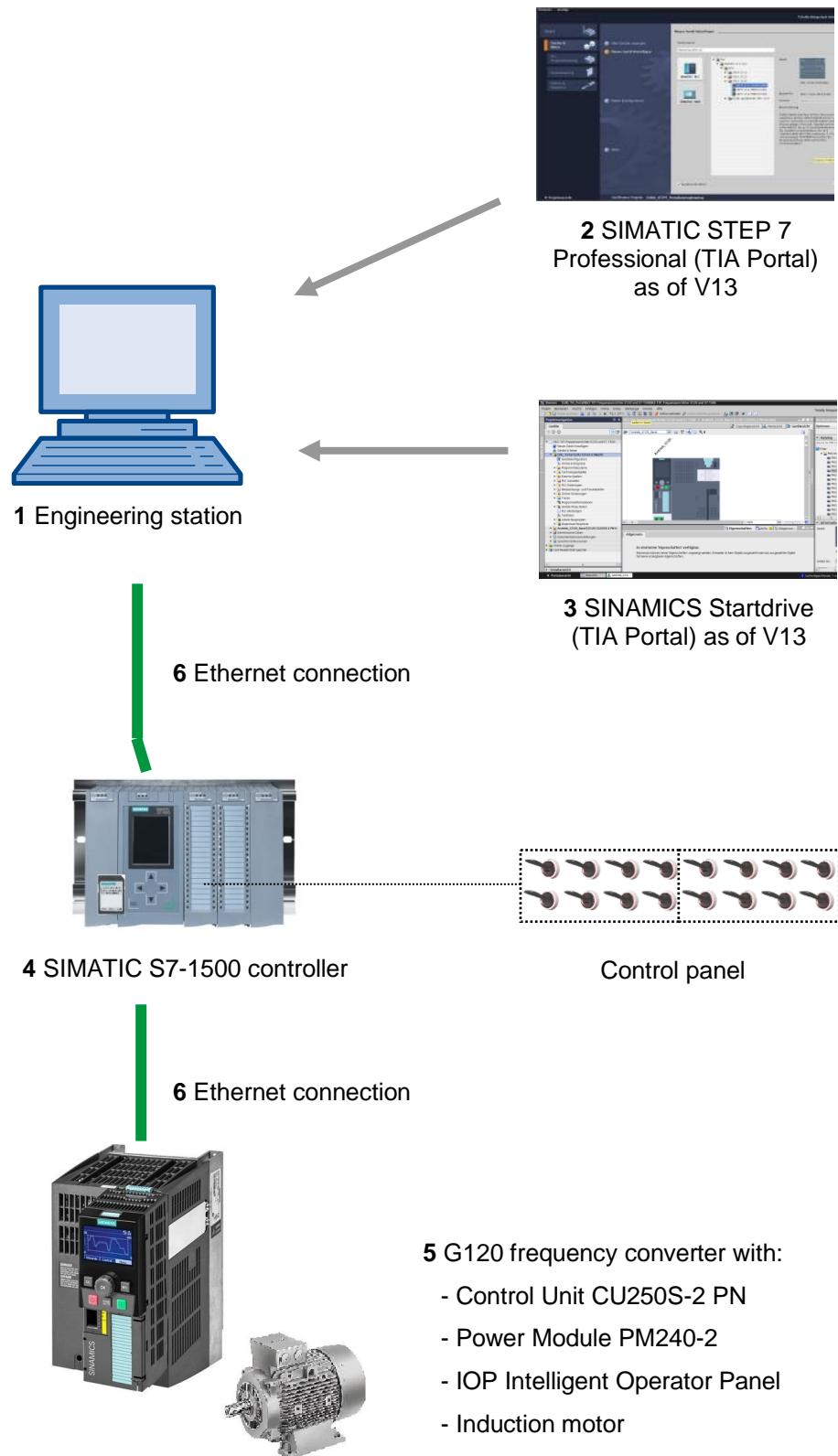
Subsequently we show step-by-step how the frequency converter SINAMICS G120 can be controlled and monitored from the program of the CPU1516F-3 PN/DP.

2 Requirement

This chapter is based on chapter "Global data blocks for SIMATIC S7 CPU1516F-3 PN/DP". In order to carry out this chapter you can for example, use the following project: Data Block.

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system (for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Professional software in TIA Portal – as of V1X
- 3 SINAMICS Startdrive software in TIA Portal – as of V1X
- 4 SIMATIC S7-1500/S7-1200/S7-300 controller, e.g. CPU 1516F-3 PN/DP – Firmware as of V1.6 with memory card and 16DI/16DO
Note: The digital inputs should be fed out to a control panel.
- 5 SINAMICS G120 frequency converter with:
 - Control Unit CU250S-2 PN as of Firmware 4.6
 - Power Module PM240-2
 - IOP Intelligent Operator Panel
 - Induction motor
- 6 Ethernet connection between engineering station and controller and between controller and frequency converter

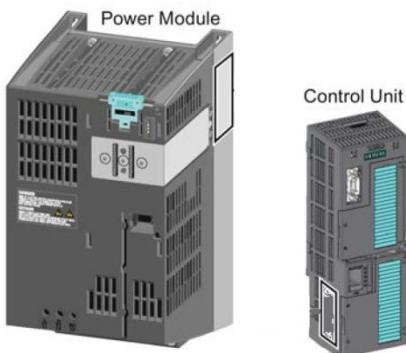


4 Theory

4.1 SINAMICS G120 frequency converter

Each SINAMICS G120 converter consists of a Control Unit (CU) and a Power Module (PM).

- The Control Unit controls and monitors the Power Module and the connected motor.
- The Power Modules contain rectifiers and converters for motors in a power range of 0.37 kW to 250 kW.



Note: More detailed information about the frequency converter G120 with Control Unit CU250S-2 PN Vector is available in the manuals. In this module the frequency converter SINAMICS G120 is used as a PROFINET IO device.

4.2 Components for configuring SINAMICS G120

4.2.1 Control Units CU250S-2



The Control Units CU250S-2 differ with regard to their type of fieldbus connections. There are Control Units CU250S-2 with:

- RS485 interface for USS, Modbus RTU
- PROFIBUS interface
- RS485 interface for PROFINET, Ethernet/IP
- CANopen interface

All the Control Units have an **EEPROM** in order for power-failure-proof storage of the configuration data.

The used Control Unit CU250S-2 Vector has a **PROFINET interface** with two ports that supports the **PROFIdrive**, **PROFIsafe** and **PROFInergy** profiles.

In addition, for example, **HTL or TTL encoders and temperature sensors** can be connected directly to a 15-pin encoder interface and **DRIVE-CLiQ-compatible encoders** as well as sensor modules to a DRIVE-CLiQ interface of the Control Unit.

The Control Unit supports the following functions of **Safety Integrated** (SIL 3, PL e, Cat. 3):

- Safe Torque Off (STO)
 - Safe Stop 1 (SS1) with and without speed monitoring
 - Safe Brake Control (SBC)
 - Safely Limited Speed (SLS)
 - Safe Direction (SDI)
- Safe Speed Monitor (SSM)
- PROFIsafe communication to a higher-level control unit

Various **control methods** are available in order to meet the wide range of requirements in drive technology:

- U/f characteristic curves
- Flux current control
- Vector regulation with and without encoders

The following **special functions** can be used with this Control Unit:

- Basic positioning function with EPOS
- Energy recovery capability through Efficient Infeed Technology (only PM250 Power Modules)

Terminals with **digital** and **analog** as well as **safe inputs and outputs** are available.

4.2.2 Operator Panels

The Operator Panels are used to commission, diagnose and control the converter as well as to back up and transfer the converter settings.



The **Intelligent Operator Panel (IOP)** is available for snapping onto the Control Unit or as a hand-held unit with a connecting line to the Control Unit. The IOP enables operator control and diagnostics of the converter.



The **BOP-2** is an Operator Panel for snapping onto the Control Unit. The BOP-2 has a two-line display for diagnostics and operator control of the converter.

Note: For further information on the Operator Panels, please refer to the manuals:

4.2.3 Memory cards for Control Unit (optional)

The SD or MMC memory cards can be optionally used to back up the converter settings.

It is possible to store up to 100 parameter sets. This can be done by using the SINAMICS Startdrive software.

A firmware update/downgrade is only possible by using a memory card.

If you use the "Basic positioner" function or the extended safety functions, a memory card with a valid license has to be inserted into the Control Unit.

Note: A memory card is not required during operation.

4.2.4 Brake Relay



The Brake Relay provides a switch contact (NO contact) to control the motor brake solenoid.

4.2.5 Safe Brake Relay



The Safe Brake Relay controls a 24-V motor brake and monitors the brake control for short-circuits and wire breaks.

4.2.6 PM240-2 Power Modules

PM240-2 Power Modules have a brake chopper (four-quadrant applications) and are suitable for a wide range of applications in general mechanical engineering. The PM240-2 Power Modules are available without a filter or with integrated Class A line filter.



The PM240-2 Power Module is available for the following voltage and power range:

- 1-phase/3-phase 200 VAC ... 240 VAC 0.55 kW ... 4.0 kW
- 3-phase 200 VAC ... 240 VAC 5.5 kW ... 7.5 kW

- 3-phase 380 VAC ... 480 V 0.55 kW ... 250 kW
- 3-phase 500 VAC ... 690 VAC 11 kW ... 132 kW

Note: If frequency converters are not put into operation for a longer period, the DC link capacitors have to be formed in accordance with the specifications in the operating instructions.

4.2.7 PM250 Power Modules

PM250 Power Modules are suitable for identical applications as the PM240 Power Modules. Any brake energy occurring can be fed back directly into the power network (four-quadrant applications – no brake chopper required). The PM250 Power Modules are available without a filter or with integrated Class A line filter.



The PM250 Power Module is available for the following voltage and power range:

- 3-phase 380 VAC - 480 VAC $\pm 10\%$ 7.5 kW to 90 kW

Note: If frequency converters are not put into operation for a longer period, the DC link capacitors have to be formed in accordance with the specifications in the operating instructions.

4.2.8 Line filter



A line filter allows the converter to reach a higher radio interference category. An external filter is not required for converters with built-in line filter.

4.2.9 Line reactor



The line reactor supports overvoltage protection, flattens the harmonics in the power network and bridges commutation notches.

4.2.10 Output reactor



Output reactors reduce the voltage load of the motor windings as well as the load of the converter through capacitive charge/discharge currents in the lines. An output reactor is required for shielded motor lines greater than 50 m or unshielded motor lines greater than 100 m.

4.2.11 Sine-wave filter



The sine-wave filter at the output of the converter limits the voltage gradient and the peak voltages at the motor motor winding. The maximum permissible motor supply line length increases to 300 m. An output reactor becomes superfluous.

4.2.12 Braking resistor



The braking resistor allows rapid braking of loads with a high moment of inertia.

The Power Module controls the braking resistor through its integrated brake chopper.

4.3 Parameter assignment of the SINAMICS G120

There are two main types of parameters:

- Display parameters
- Adjustable parameters

4.3.1 ***Display parameters***

Display parameters allow the reading of the internal measured quantities of the converter and motor. The Operator Panel and SINAMICS Startdrive represent the display parameters with a preceding "r". For example, r0027 is the parameter for the output current of the converter.

4.3.2 ***Adjustable parameters***

Adjustable parameters are the parameters that you use to adjust the converter to your application. When you change the value of an adjustable parameter, you also change the behavior of the converter. Adjustable parameters are represented with a preceding "p". For example, p1082 is the parameter for adjusting the maximum speed of the motor.

The following section displays some particularly important adjustable parameters.

Note: Further information on the parameters is available in the list manual.

4.3.3 ***P0010 Drive commissioning parameter filter***

Parameter P0010 filters parameters so that only the parameters assigned to a specific function group can be selected. This means, for example, that the parameters required for quick commissioning are displayed in order. The following settings are available:

- P0010 = 0: Ready

In order to start up the converter, the P0010 has to be set to 0.

- P0010 = 1: Quick commissioning
- P0010 = 2: Power unit startup
- P0010 = 3: Motor startup
- P0010 = 4: Encoder startup
- P0010 = 5: Technological application/units
- P0010 = 11: Function modules
- P0010 = 15: Data records
- P0010 = 17: Basic positioning startup
- P0010 = 25: Position control startup
- P0010 = 29: Only Siemens-internal
- P0010 = 30: Parameter reset
- P0010 = 39: Only Siemens-internal
- P0010 = 49: Only Siemens-internal
- P0010 = 95: Safety Integrated startup

By setting p3900 unequal to 0, the quick commissioning is complete, and this parameter is set automatically to 0.

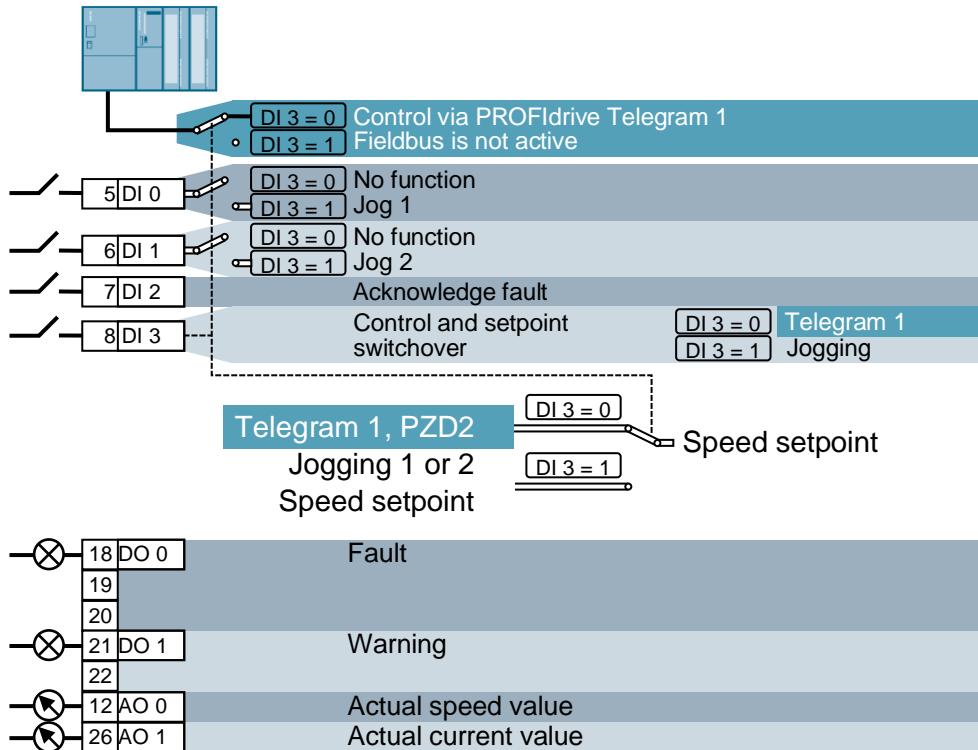
4.3.4 P0015 Macro drive unit

With the parameter P0015 you select command and setpoint sources of the converter by executing the corresponding macro files.

After the value has changed, the further changing of parameters is blocked as long as the macro is being executed. The status is displayed in r3996. Changing is not possible until r3996 = 0 again.

When a specific macro is executed, the correspondingly programmed settings are carried out and become effective.

For example, Macro 7: "Fieldbus with data record changeover"



Note: Information about further macros is available in the operating instructions of the respective Control Unit.

4.3.5 Changeability depending on the converter state

"P"-parameters can furthermore only be changed depending on the status of the converter.

For example, the parameter p1120 Ramp-function generator ramp-up time (with the attribute "C(1), U, T" in the parameter list) can only be changed in the quick commissioning "C", when P0010 = 1, in the ready state "T" or during operation "U".

Status	Description
C(*)	Quick commissioning (P0010 = *)
U	Operation (drive running)
T	Drive ready-to-start

4.3.6 BICO technology

A converter corresponding to the latest state-of-the-art has to offer the possibility to freely interconnect internal and external signals (setpoints or actual values as well as control and status signals).

This interconnection has to offer a high degree of flexibility so that the converter can be easily adapted to new applications.

The BICO technology and macros are used to meet these requirements.

By using the BICO technology the process data can be interconnected freely while using the "default" parameter assignment of the converter.

Here all the values that can be interconnected freely are defined as "connectors", for example, frequency setpoint, actual frequency value, actual current value, etc.

All the digital signals that can be interconnected freely are defined as "binectors", for example, status of a digital input, ON/OFF, message function at limit violations, etc.

A converter contains numerous input and output variables as well as variables within the control system that can be interconnected. Therefore it is possible to adapt the converter to the various requirements by using the BICO technology.

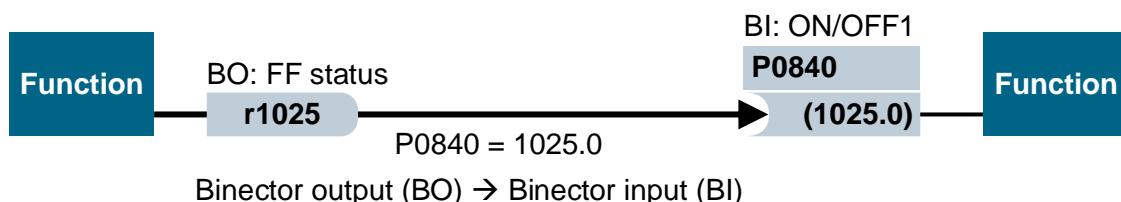
Binectors

A binector is a digital (binary) signal without any units that can have either the value 0 or 1. Binectors always reference functions. They are divided into binector inputs (BI) and binector outputs (BO).

The binector input is always identified with a "P"-parameter (for example, P0840 BI: ON/OFF1), whereas the binector output is always represented with an "r"-parameter (for example, r1025 BO: FF status).

Example

Combination of the command ON/OFF1 with selection of a fixed frequency.



When a fixed frequency is selected, the fixed frequency status bit (r1025) is changed internally from 0 to 1.

The source for the command ON/OFF1 is the parameter P0840 (default DI0). When the fixed frequency status bit is connected as the source for P0840 (P0840 = 1025), the converter starts by activating a fixed frequency and stops with OFF1 for deactivation of the fixed frequency.

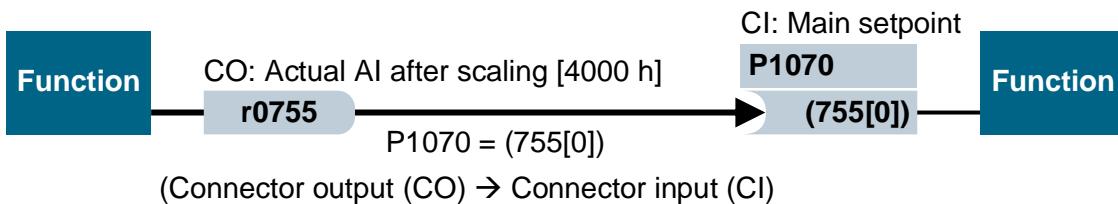
Connectors

A connector (16 or 32 bits) has a value that can contain a normalized variable (dimensionless) or also a variable with assigned units.

Connectors always reference functions. They are divided into connector inputs (CI) and connector outputs (CO). In essence the same applies as for binectors: The connector inputs are identified by a "P"-parameter (for example P0771 CI: AO (analog output)), whereas the connector outputs are always represented with an "r"-parameter (for example r0021 CO: Actual frequency).

Example

Interconnection of the parameter r0755 (display analog input) with an internal value (main frequency setpoint). To this purpose the CO parameter r0755 (scaled analog input) has to be interconnected with the CI parameter P1070 (main setpoint).



Note: For further details please refer to the list manual.

4.3.7 Control Data Set (CDS) and Drive Data Set (DDS)

Drive engineering has applications in which simultaneous changeover of multiple parameters with external signals is needed during operation.

To enable this, certain parameters have been organized into groups. These so-called data sets are:

- Control Data Set (CDS)
- Drive Data Set (DDS)

Note: For more details, refer to the list manual and the operating instructions.

4.4 Commissioning of the SINAMICS G120 frequency converter

A converter of the type G120 always consists of the Power Module and the Control Unit. After the initial latching in of the Control Unit at the Power Module and switching on of the supply voltage, the Power Module is recognized by the Control Unit. If it is a compatible Power Module, the data are stored in the Control Unit.

Commissioning of the converter G120 is usually carried out in the following steps:

- Resetting to factory settings
- Basic commissioning
- Quick commissioning
- Calculation of the motor/control data
- Optimization of the speed control
- Further settings for commissioning
- Optional: Motor data identification
- Startup of the application
- Commissioning of fail-safe functions (only with fail-safe applications)

4.4.1 Restoring factory settings through a parameter reset

The factory setting can be effected via the SINAMICS Startdrive software, via a menu function in the Intelligent Operator Panel (IOP) or via a direct parameter input.

Procedure for "Reset parameters":

p0010 = 30

p0970 = 1

P0970 = 0 is automatically set at the end of the calculations.

Through a factory setting via P0970, the original values of all the converter parameters can be restored. These values are designated with "Factory Setting" in the list manual.

The following parameters remain unchanged after a reset to factory settings:

- P0014 Storage mode
- Communication parameters (for example PROFIBUS and PROFINET settings)
- Power-Module-dependent data

4.4.2 Basic commissioning

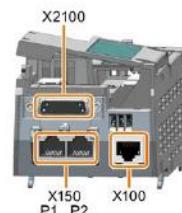
Basic commissioning should always be carried out by using the commissioning wizard via the SINAMICS Startdrive software or the Intelligent Operator Panel (IOP).

Alternatively, quick commissioning (P0010 = 1) can also be carried out by direct entry of the parameters. However, this procedure is not advisable.

Notes: Commissioning by using the commissioning wizard via the SINAMICS Startdrive software is described in Chapter 6 of this document.

For information about carrying out commissioning by using the commissioning wizard via the Intelligent Operator Panel (IOP) please refer to the operating instructions of the IOP.

4.5 PROFINET interface of the SINAMICS G120, CU250S-2 PN Vector



The frequency converter can be integrated into an Ethernet network at the PROFINET interface X150 with the two ports P1 and P2. Now:

- The parameter assignment and diagnostics of the frequency converter via Ethernet can be carried out by using the SINAMICS Startdrive software in the TIA Portal.
- The converter can be integrated into a PROFINET network.

In PROFINET IO operation, the converter supports the following functions:

- IO-RT: Real-time communication (as used in this document.)
- IO-IRT: Isochronous real-time communication
- MRP: Media redundancy when used in a network with ring topology
- MRPD: Media redundancy requirement: IRT when used in a network with ring topology
- Diagnostic interrupts in accordance with the error classes specified in the PROFIdrive profile

4.5.1 Telegrams

Various telegrams, whose process data lengths and contents differ, are available for selection for IO-RT communication with the frequency converter.

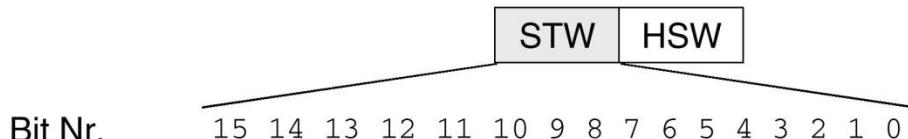
The simplest telegram, set as the standard, is the Standard Telegram 1.

4.5.2 Assignment of the process data (PZD) for the SINAMICS G120 with Standard Telegram 1

Control words and setpoints (PLC -> SINAMICS) and status words and actual values (SINAMICS -> PLC) can be transferred with the process data. The structure of the PZD area is as follows for Telegram 1, for a coupling via PROFINET:

	PZD1	PZD2
Request telegram (PLC -> SINAMICS)	Control word (STW1)	Main setpoint (NSOLL_A)
Response telegram (SINAMICS -> PLC)	Status word (ZSW1)	Main actual value (NIST_A)

4.5.3 Control word 1 (STW1)

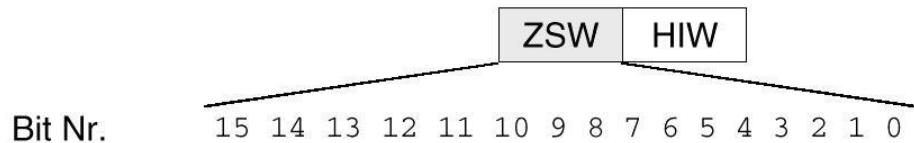


Bit	Significance		Comments	Signal interconnection in the inverter
	Telegram 20	All other telegrams		
0	0 = OFF1		The motor brakes with the ramp-down time p1121 of the ramp-function generator. The inverter switches off the motor at standstill.	P0840[0] = r2090.0
	0 → 1 = ON		The inverter goes into the "ready" state. If, in addition bit 3 = 1, then the inverter switches on the motor.	
1	0 = OFF2		Switch off the motor immediately, the motor then coasts down to a standstill.	P0844[0] = r2090.1
	1 = No OFF2		The motor can be switched on (ON command).	
2	0 = Quick stop (OFF3)		Quick Stop: The motor brakes with the OFF3 rampdown time p1135 down to standstill.	P0848[0] = r2090.2
	1 = No Quick Stop (OFF3)		The motor can be switched on (ON command).	
3	0 = Inhibit operation		Immediately switch-off motor (cancel Pulses).	P0852[0] = r2090.3
	1 = Enable operation		Switch-on motor (pulses can be enabled).	
4	0 = Disable RFG sperren		The inverter immediately sets its ramp-function generator output to 0.	p1140[0] = r2090.4
	1 = Do not disable RFG		The ramp-function generator can be enabled.	
5	0 = Stop RFG		The output of the ramp-function generator stops at the actual value.	P1141[0] = r2090.5
	1 = Enable RFG		The output of the ramp-function generator follows the setpoint.	
6	0 = Inhibit setpoint		The inverter brakes the motor with the ramp-down time p1121 of the ramp-function generator.	P1142[0] = r2090.6
	1 = Enable setpoint		Motor accelerates with the ramp-up time p1120 to the setpoint.	
7	0 → 1 = Acknowledge faults		Acknowledge fault. If the ON command is still active, the inverter switches to „closing	p2103[0] = r2139.7

		lockout" state.	
8, 9	Reserved		
10	0 = No control via PLC	Inverter ignores the process data from the fieldbus.	P0854[0] = r2090.10
	1 = Control via PLC	Control via fieldbus, inverter accepts the process data from the fieldbus.	
11	1 = Direction reversal	Invert setpoint in the inverter.	p1113[0] = r2090.11
12	Not used		
13	---1)	1 = MOP up	P1035[0] = r2090.13
14	---1)	1 = MOP down	P1036[0] = r2090.14
15	CDS bit 0	Reserved	P0810 = r2090.15

1) If you change over from another telegram to telegram 20, then the assignment of the previous telegram is kept.

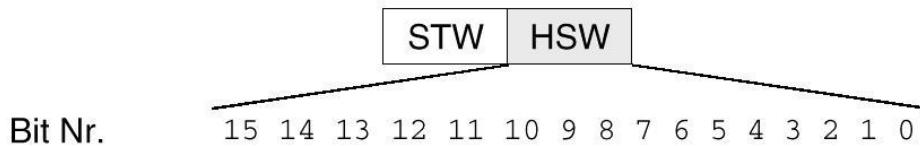
4.5.4 Status word 1 (ZSW1)



Bit	Significance		Comments	Signal interconnection in the inverter
	Telegram 20	All other telegrams		
0	1 = Ready to start		Power supply switched on; electronics initialized; pulses locked.	P2080[0] = r0899.0
1	1 = Ready		Motor is switched on (ON/OFF1 = 1), no fault is active. With the command "Enable operation" (STW1.3), the inverter switches on the motor	p2080[1] = r0899.1
2	1 = Operation enabled		Motor follows setpoint. See control word 1, bit 3	p2080[2] = r0899.2
3	1 = Fault active		The inverter has a fault. Acknowledge fault using STW1.7.	p2080[3] = r2139.3
4	1 = OFF2 inactive		Coast down to standstill is not active.	p2080[4] = r0899.4
5	1 = OFF3 inactive		Quick Stopp is not active	p2080[5] = r0899.5
6	1 = Closing lockout active		It is only possible to switch on the motor after an OFF1 followed by ON.	p2080[6] = r0899.6
7	1 = alarm active		Motor remains switched on; no acknowledgement is necessary.	p2080[7] = r2139.7
8	1 = Speed deviation within the tolerance range		Setpoint/actual value deviation within the tolerance range.	p2080[8] = r2197.7
9	1 = Master control requested		The automation system is requested to accept the inverter control.	p2080[9] = r0899.9
10	1 = Comparison speed reached or exceeded		Speed is greater than or equal to the corresponding maximum speed.	p2080[10] = r2199.1
11	1 = current or torque limit reached	1 = torque limit reached	Comparison value for current or torque has been reached or exceeded.	p2080[11] = r0056.13/ r1407.7
12	---1)	1 = Holding brake open	Signal to open and close a motor holding brake.	p2080[12] = r0899.12
13	0 = Alarm, motor overtemperature		—	p2080[13] = r2135.14
14	1 = Motor rotates clockwise	0 = Motor rotates counterclockwise	Internal inverter actual value > 0 Internal inverter actual value < 0	p2080[14] = r2197.3
15	1 = CDS display	0 = Alarm, inverter thermal overload		p2080[15] = r0836.0/ r2135.15

- 2) If you change over from another telegram to telegram 20, then the assignment of the previous telegram is kept.

4.5.5 Main setpoint (HSW/NSOLL_A; 16-bit)



The main setpoint is a 16-bit word in which the required speed is transferred to the converter.

The setpoint is transferred as an integer with preceding sign (-32768 to 32767). The value 16384 (4000 Hex) corresponds to +100%.

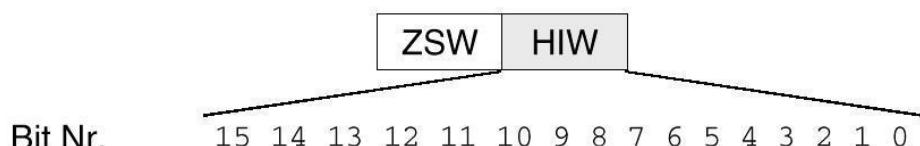
The value 100% is defined at a particular rotary speed by means of the parameter P2000 (reference speed). The speed that is to correspond to a setpoint of 100% via the interface is entered in this parameter.

The speed of the converter is calculated as follows:

$$n = (\text{HSW} \times \text{P2000})/16384$$

Note: The parameter P2000 (reference speed) is automatically calculated for Drive Data Set 0 during motor startup and set to the value of parameter P1082 (maximum speed).

4.5.6 The main actual value (HIW/NIST_A; 16-bit)



The main actual value is a 16-bit word through which the actual speed of the converter is transferred. The normalization of this value corresponds to that of the setpoint.

$$n = (\text{HIW} \times \text{P2000})/16384$$

Note: The parameter P2000 (reference speed) is automatically calculated for Drive Data Set 0 during motor startup and set to the value of parameter P1082 (maximum speed).

4.5.7 Layout of the request telegram in double-word format

The request telegram is sent to the SINAMICS G120 in double-word format.

The layout of the bits is shown in the table.

Control word																Main setpoint															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QB 256								QB 257								QB 258								QB 259							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0

4.5.8 Layout of the response telegram in double-word format

The response telegram is returned by the SINAMICS G120 in double-word format.

The layout of the bits is shown in the table.

Status word																Main actual value															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IB 256								IB 257								IB 258								IB 259							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0

Note: A data block in which the data are stored temporarily is used for the request telegram and for the response telegram in the control program. The telegrams are mapped there respectively in a structure that is created by means of the PLC data types.

4.6 SINAMICS Startdrive commissioning tool for SINAMICS G120

The most recent version of the SINAMICS Startdrive commissioning software can be downloaded from the Website:

support.industry.siemens.com.

SINAMICS Startdrive is a tool integrated in TIA Portal and corresponds to the familiar TIA Portal in its structure and handling.

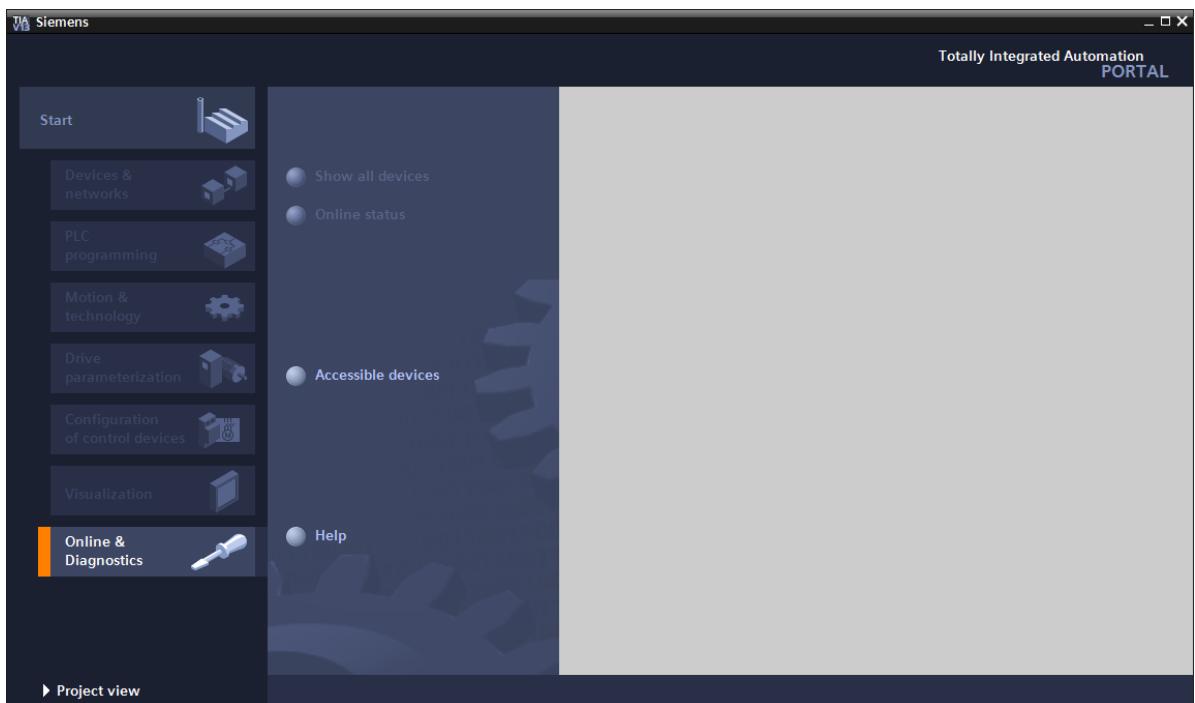
The SINAMICS Startdrive extension contains the data and views for the SINAMICS G120 frequency converters already supported there.

This enables easy parameter assignment and commissioning of the frequency converters. A wide range of functions and aids are available for diagnostics and troubleshooting.

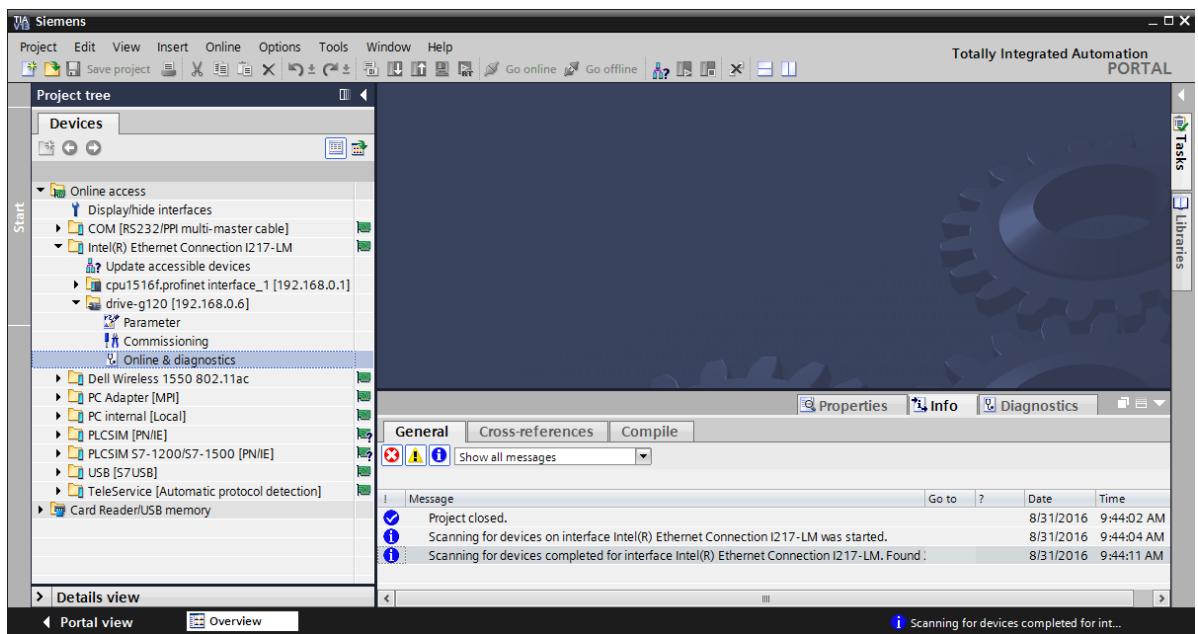
4.6.1 Resetting frequency converters and setting the IP address

A new IP address can be directly assigned to the Control Unit of the frequency converter with SINAMICS Startdrive in TIA Portal. The Control Unit can now be reset.

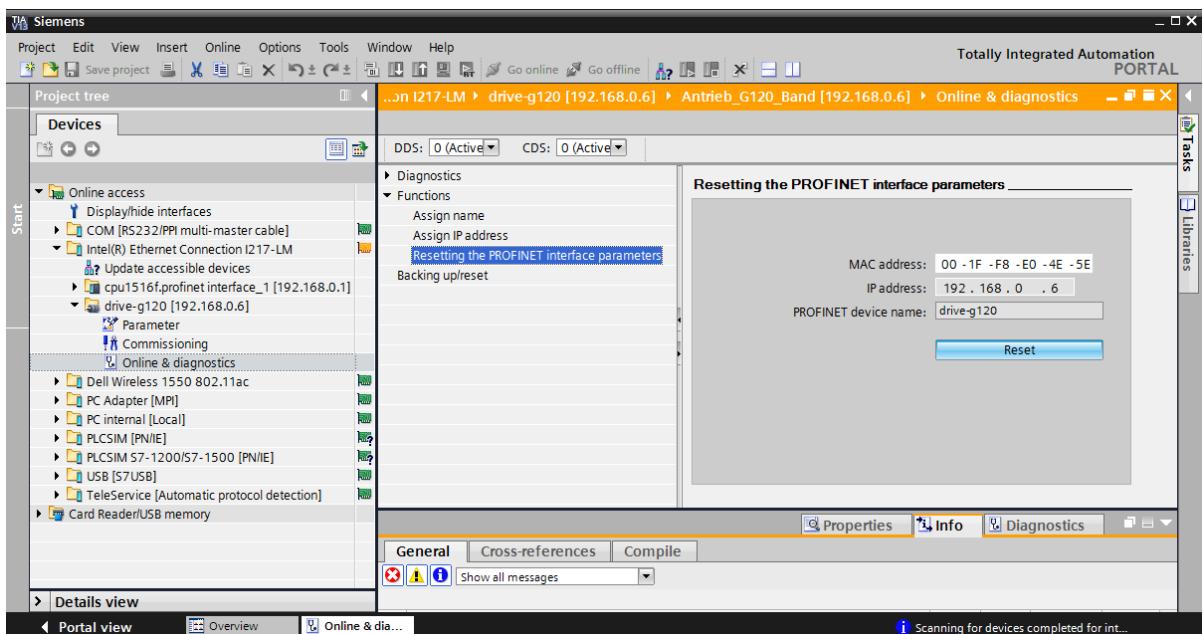
- To do so select the Totally Integrated Automation Portal, which is opened with a double-click. (→ TIA Portal V1X)
- Then select the item → "Online & Diagnostics" and open the → "Project view".



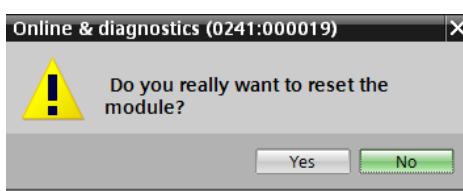
- In the project tree select the network card of your computer under → "Online access". When you click → "Update accessible devices", you see the IP address (if already set) or the MAC address (if the IP address has not yet been assigned) of the Control Unit of the connected SINAMICS G120 frequency converter. Select → "Online & diagnostics".



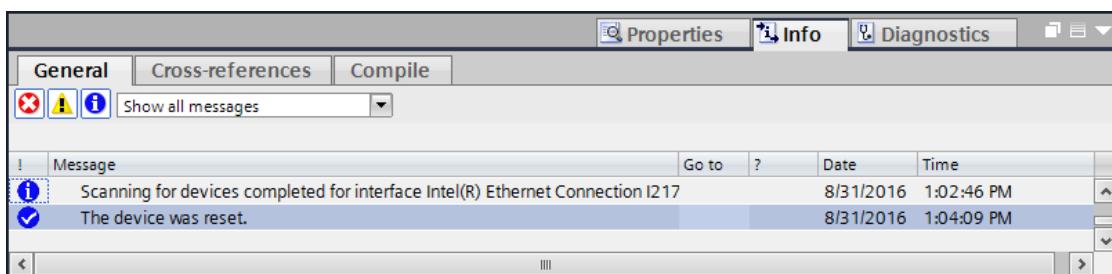
- Before you reassigned the IP address, we recommend that you first reset the PROFINET interface parameters. To do so select the function → "Resetting the PROFINET interface parameters" and click → "Reset".



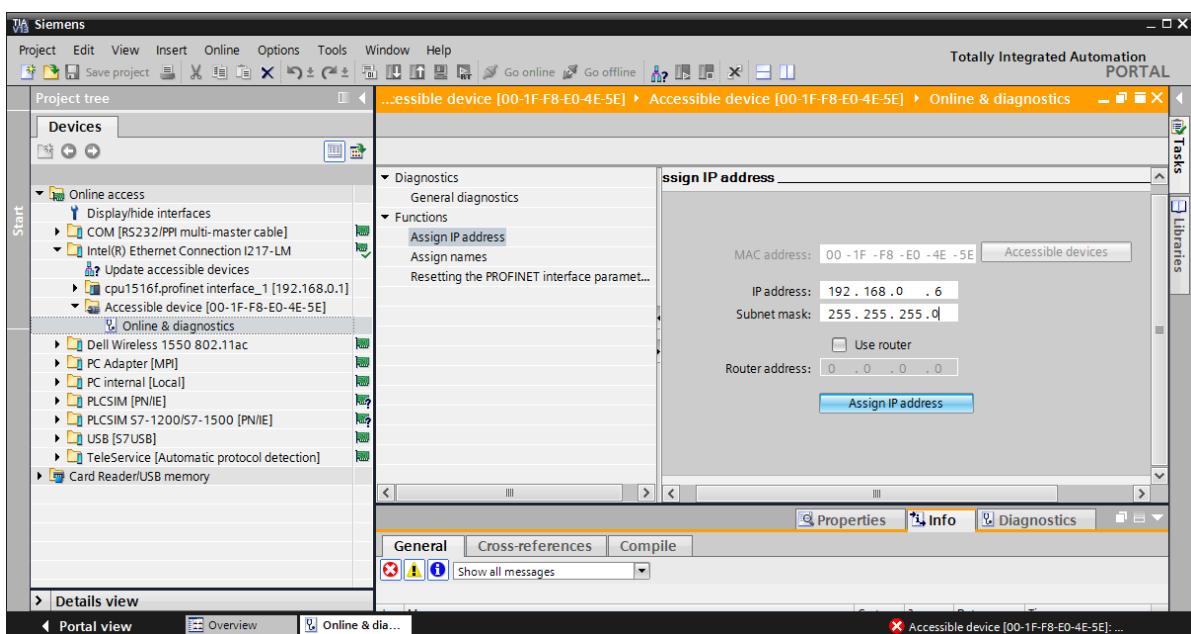
- Answer the prompt whether you really want to reset with → "Yes".



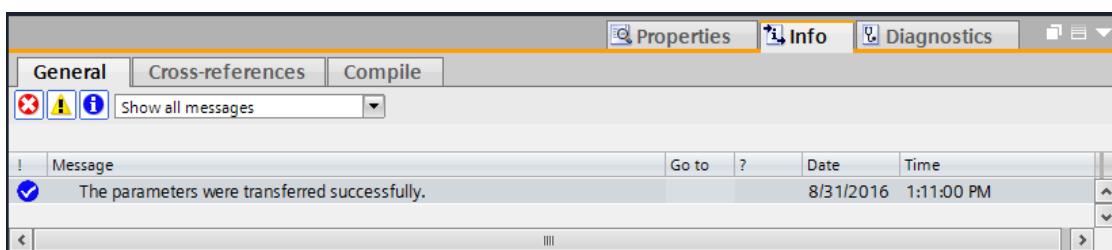
- Successful resetting can be checked in the messages in the → "Info" window → "General".



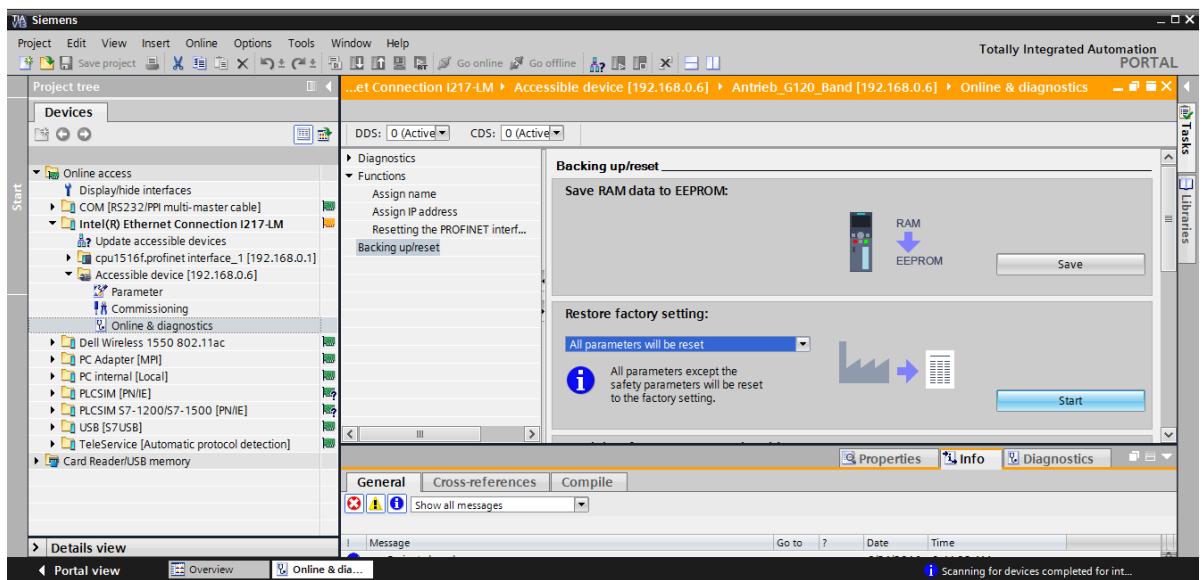
- Then once again select → "Update accessible devices" and then → "Online & diagnostics" of your frequency converter. To assign the IP address, select the function → "Assign IP address". Enter the following IP address at this point: → IP address: 192.168.0.6 → Subnet mask: 255.255.255.0. Click → "Assign IP address" and this new address is assigned to the Control Unit of your frequency converter.



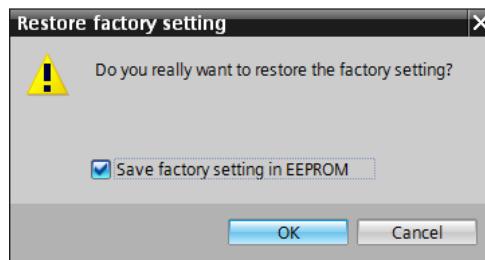
- The successful assignment of the IP address is shown again as a message in the window → "Info" → "General".



- Before you can now carry out resetting of the frequency converter to the factory setting you have to once again select → "Update accessible devices" and the → "Online & Diagnostics" of your frequency converter. In order to reset the frequency converter to factory settings, select → "Restore factory resetting" under → "Backing up/reset" and click → "Start".



- Select the option "Save factory setting in EEPROM" so that the parameters of the factory setting are loaded from the EEPROM into the RAM of the device after switching off and on - and not the data of an old project. Confirm the prompt whether you really want to reset with → "OK".



Note: The communication settings such as the IP address and the subnet mask are retained when the frequency converter is set to the factory setting.

5 Task

In the following section the project from the chapter "SCE_EN_032-600_Global_Data_Blocks" is to be supplemented by a frequency converter G120 with Control Unit CU250S-2 PN.

Controlling of the belt motor via analog values is now replaced by the controlling of the frequency converter via PROFINET. Monitoring of the actual speed value is also effected via PROFINET.

6 Planning

The conveyor belt driven by an induction motor will now be controlled via a frequency converter with a variable speed.

This frequency converter has to be created, configured and commissioned in the project.

The parameter assignment of the frequency converter is done offline with the SINAMICS Startdrive software, whereby the commissioning wizard is used.

Here the motor data of the induction motor are taken from the rating plate of the motor and entered manually.

In this project the following induction motor is wired in Delta mode and operated single-phase with 230V.

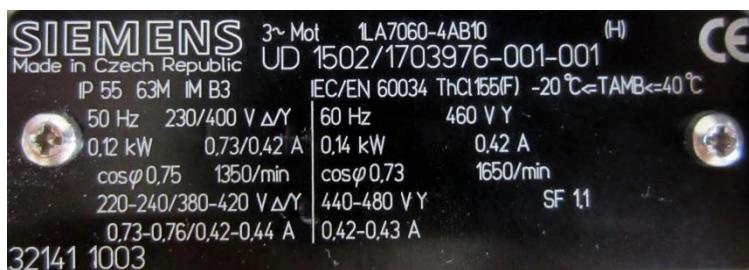


Figure 1: Rating plate of induction motor

A diagram of the two connection types can be found on the inside of the terminal box cover of most motors:

- Star connection (Y)
- Delta connection (Δ)

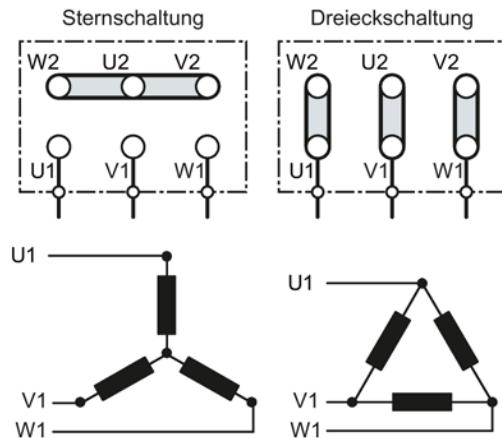


Figure 2: Star connection/delta connection

The frequency converter SINAMICS G120 receives the start command and speed specification in the following via PROFINET from the SIMATIC S7-1500. The actual speed value is also read out of the SINAMICS G120 frequency converter via PROFINET and is monitored for the high and low limits in the SIMATIC S7-1500.

A "Frequency converter" data block [DB4] is created in the control program in which the data are stored temporarily for the request telegram and the response telegram. The telegrams are created there by means of the PLC data types and are mapped respectively in a structure.

In the "Main" organization block [OB1] you copy the actual values from the converter into the "Frequency converter" data block [DB4] and the setpoints from the data into the converter.

Finally the data created in the "Frequency converter" data block [DB4] can be accessed when calling up the functions and function blocks.

6.1 Technology schematic diagram

At this point you see the technology schematic diagram for the task.

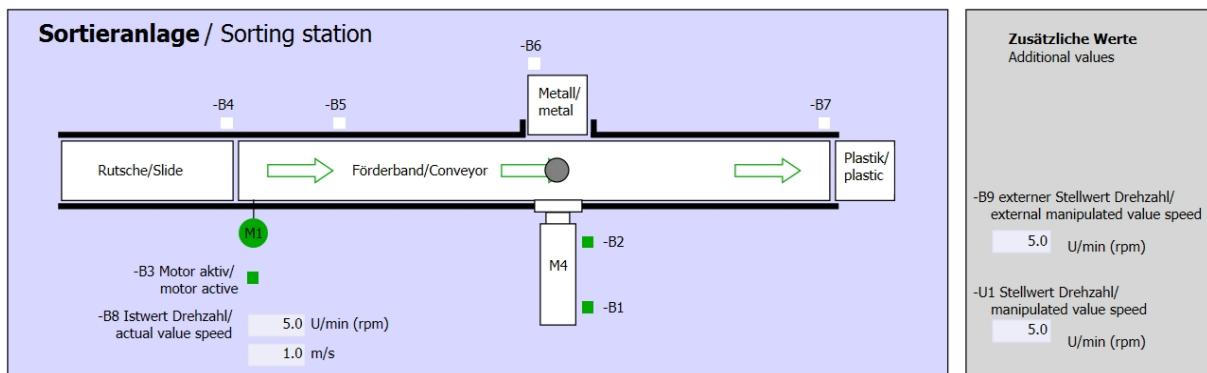


Figure 3: Technology schematic diagram



Figure 4: Operator panel

6.2 Reference table

The following signals are required as global operands for this task.

DI	Type	ID	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop ok	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
I 0.2	BOOL	-S0	Mode selector manual (0)/automatic (1)	Manual = 0 Auto=1
I 0.3	BOOL	-S1	Pushbutton automatic start	NO
I 0.4	BOOL	-S2	Pushbutton automatic stop	NC
I 0.5	BOOL	-B1	Sensor cylinder -M4 retracted	NO
I 1.0	BOOL	-B4	Sensor part at slide	NO
I 1.3	BOOL	-B7	Sensor part at end of conveyor	NO
ID256	STRUCT	PZD_IN_G120_01	Telegram 1 receive process data from G120 conveyor1	

DO	Type	ID	Function	
OD256	STRUCT	PZD_OUT_G120_01	Telegram 1 send process data to G120 conveyor1	

Legend for reference list

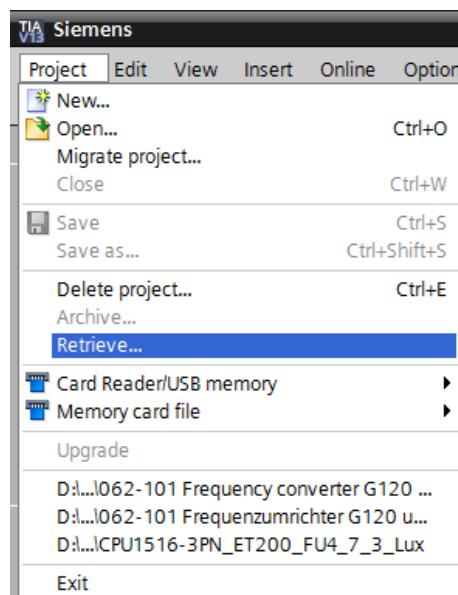
DI	Digital input	DO	Digital output
AI	Analog input	AO	Analog output
I	Input	O	Output
NC	Normally Closed		
NO	Normally Open		

7 Structured step-by-step instructions

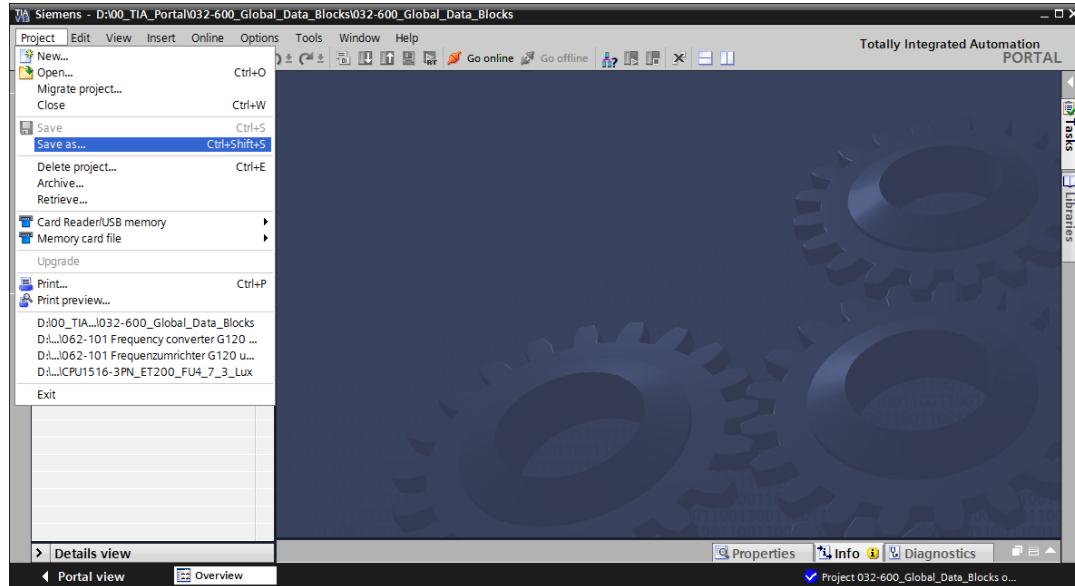
You can find instructions on how to perform planning below. If you already have a good understanding of everything, it is sufficient to focus on the numbered steps. Otherwise, simply follow the steps of the instructions illustrated below.

7.1 Retrieving an existing project

- Before we can extend the project: Data Block " from the chapter Data Block, we have to retrieve it. To retrieve an existing project you have to select the respective archive in the project view under → Project → Retrieve. Confirm your selection with Open.
(→ Project → Retrieve → Selection of a .zap archive → Open)

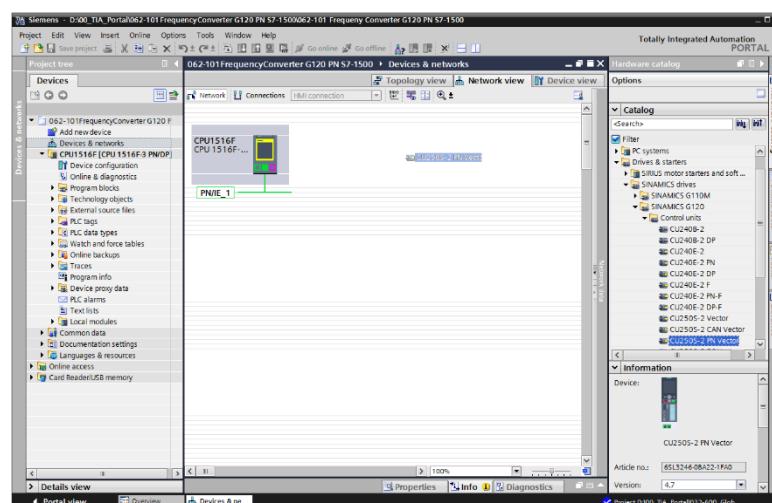


- Next, the target directory in which the retrieved project is to be stored can be selected.
Confirm your selection with "OK".
(→ Target directory → OK)
- Save the opened project under the name 062-101 Frequency converter G120 and S7-1500.
(→ Project → Save as ... → 062-101 Frequency converter G120 and S7-1500 → Save)



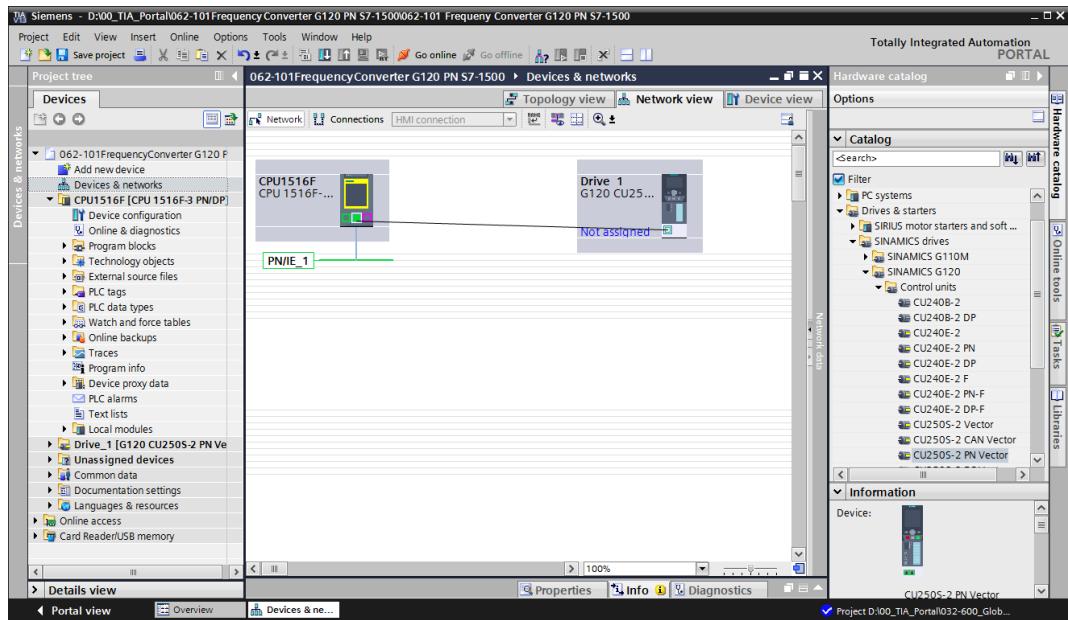
7.2 Creating a frequency converter in the TIA Portal

- In order to network the Control Unit of the SINAMICS G120 with the CPU1516F-3 PN/DP you have to change to the 'Network view'. At this point the desired 'CU250S-2 PN Vector' can be dragged-and-dropped into the network view.
(→ Devices & networks → Network view → Drives & starters → SINAMICS drives → SINAMICS G120 → Control units → CU250S-2 PN Vector → Article No.: 6SL3246-0BA22-1FA0 → Version 4.7).



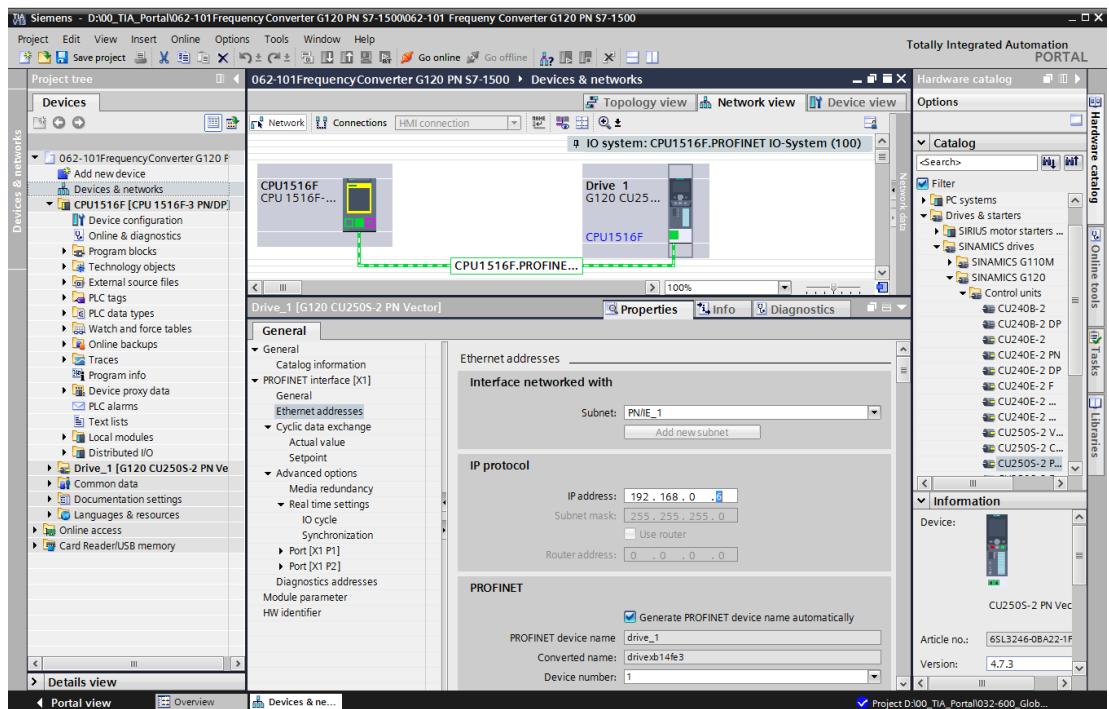
→ Connect the Ethernet interfaces of the Control Unit of the G120 and the CPU1516F-3 PN

with the mouse. (→ Ethernet → Ethernet)



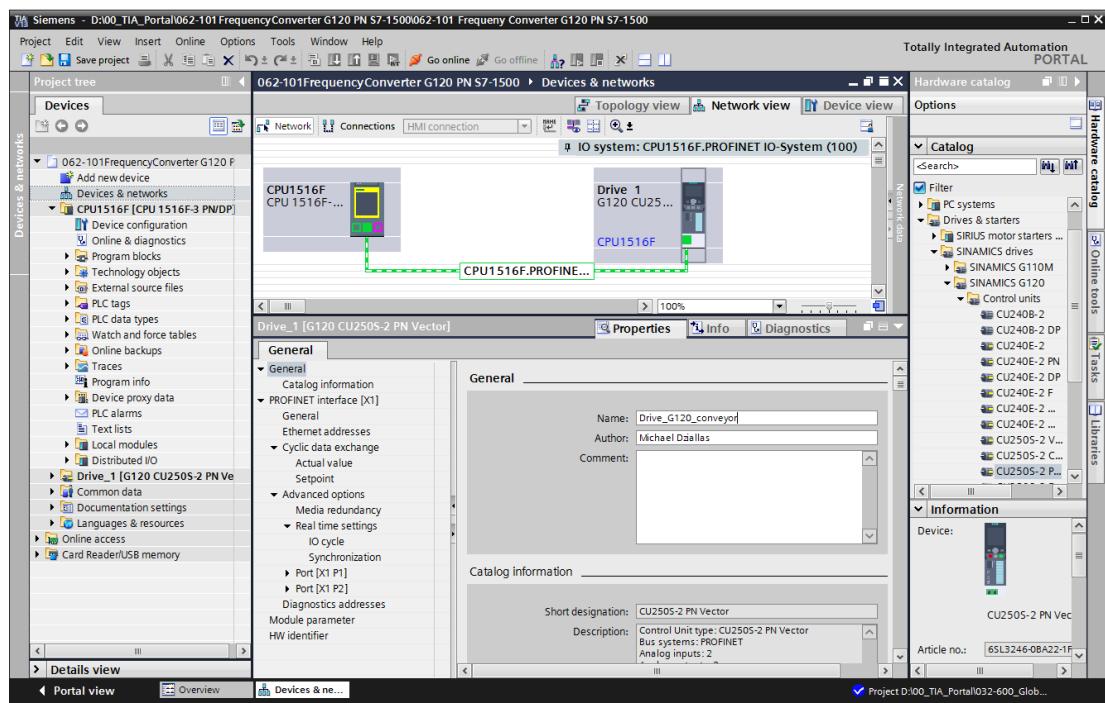
→ Set an IP address suitable for the CPU in the properties of the 'PROFINET interface [X1]' of the 'G120'.

(→ G120 CU250S-2 PN Vector → PROFINET interface [X1] → Properties → Ethernet addresses → IP protocol → IP address: 192.168.0.6)



→ The device name is entered under 'General'.

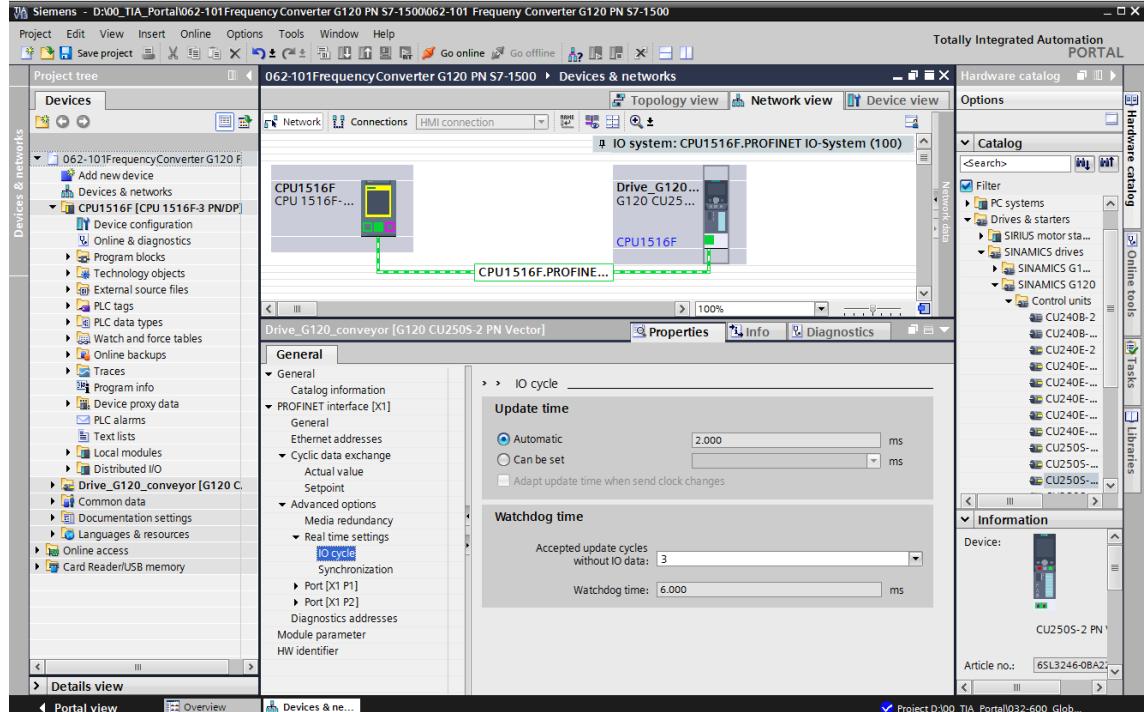
(→ General → Name: Drive_G120_conveyor)



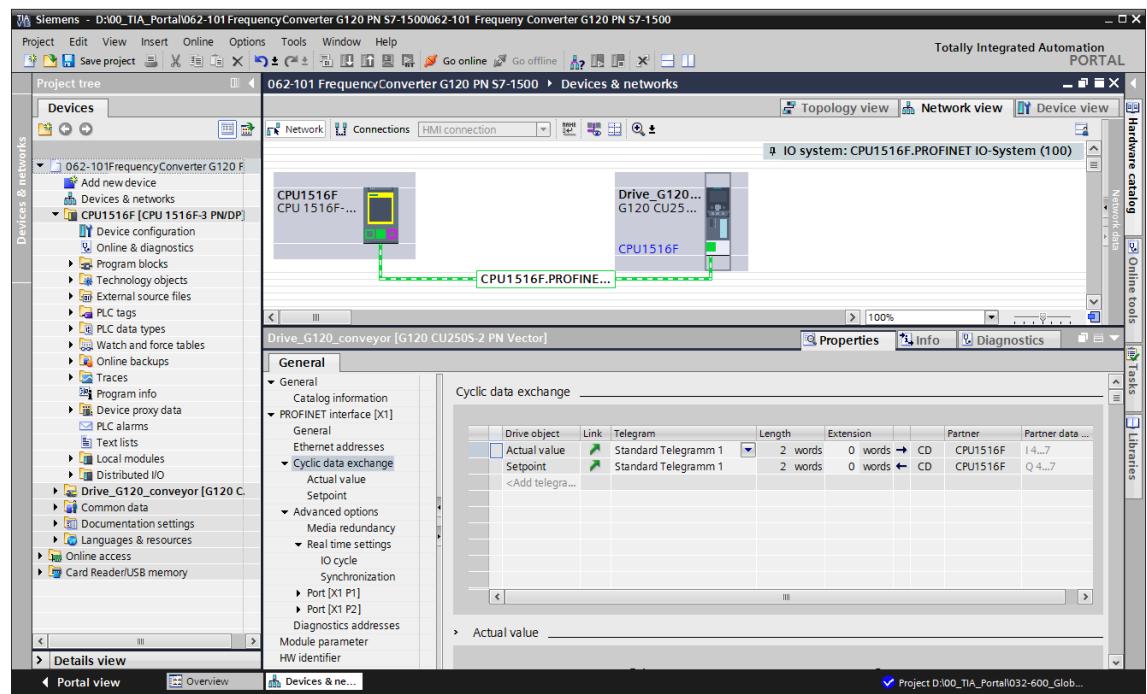
Note: This name is applied automatically as the PROFINET device name under the 'PROFINET' point for the 'PROFINET interface' of the 'G120 CU250S-2 PN-Vector'.

Settings for the 'IO cycle' such as the 'Update time' and 'Watchdog time' can also be set for this device.

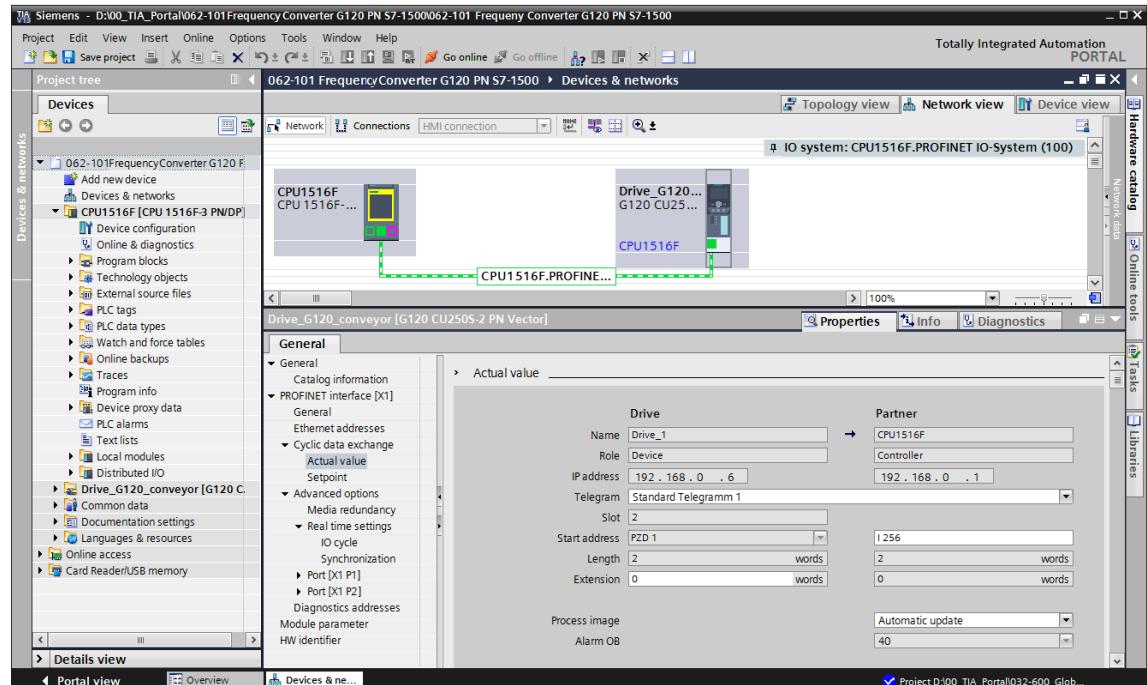
(→ Advanced options → Real time settings → IO cycle → Update time → Watchdog time)

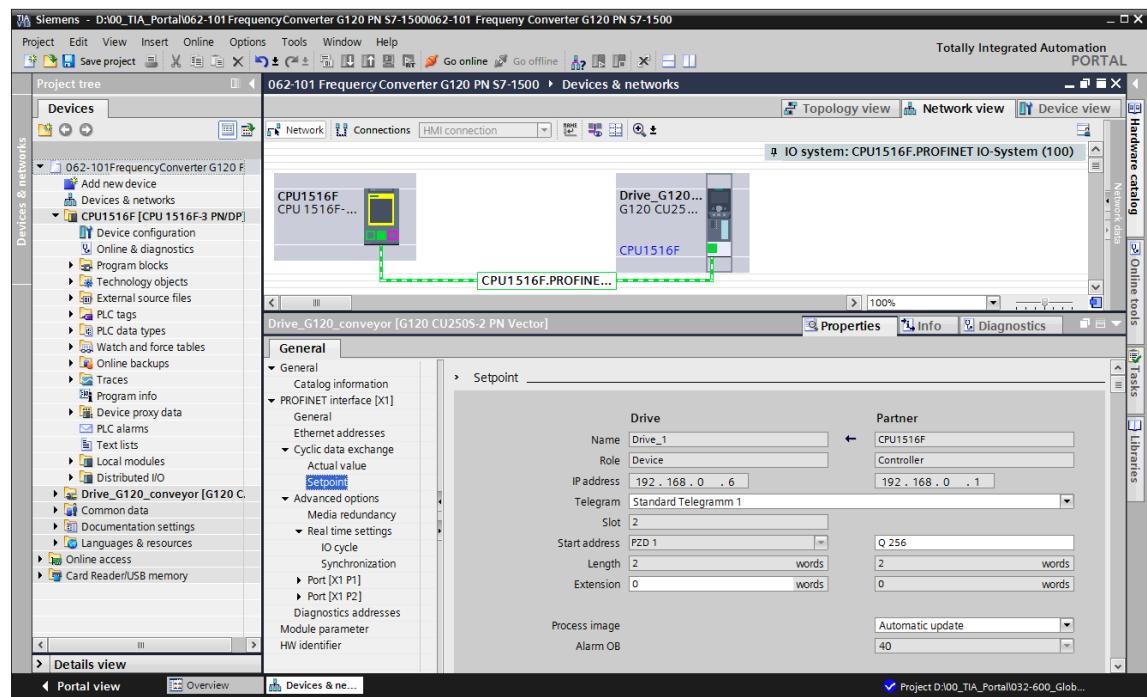


- The 'Standard Telegram 1' is specified for the 'Cyclic data exchange' between the PLC and the frequency converter.
(→ PROFINET interface [X1] → Cyclic data exchange → Actual value: Standard Telegram 1 → Setpoint: Standard Telegram 1)

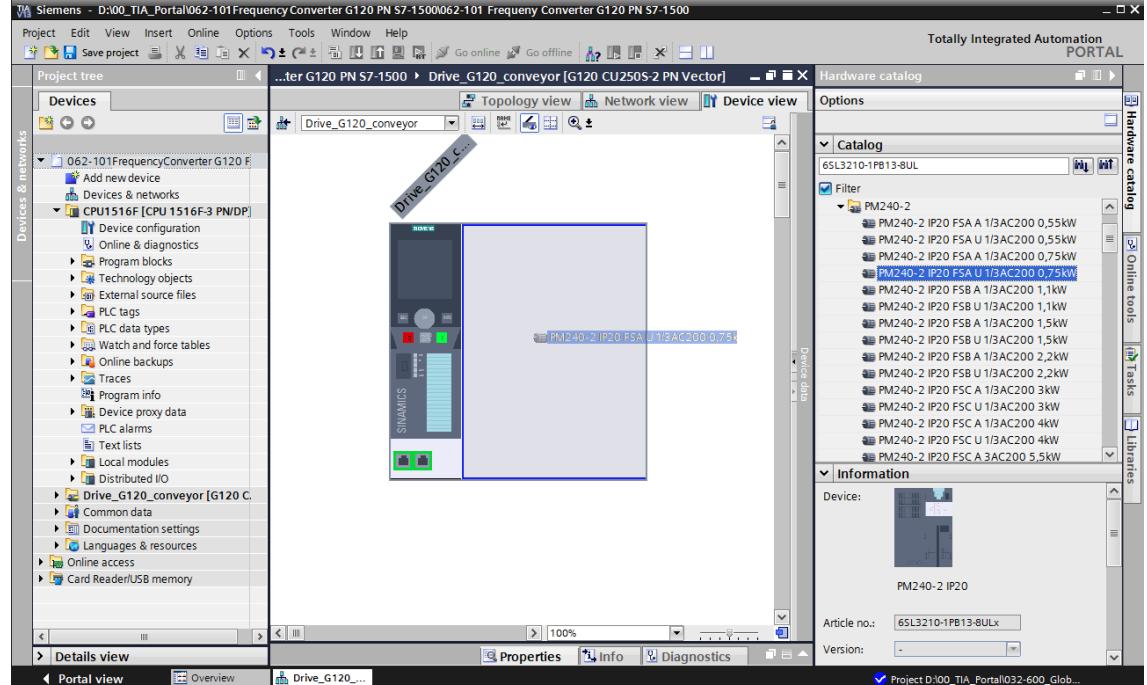


- For the address ranges, select 'I 256...259' and 'O 256 ... 259'.
 (→ PROFINET interface [X1] → Cyclic data exchange → Actual value → Start address I 256 → Setpoint → Start address O 256)



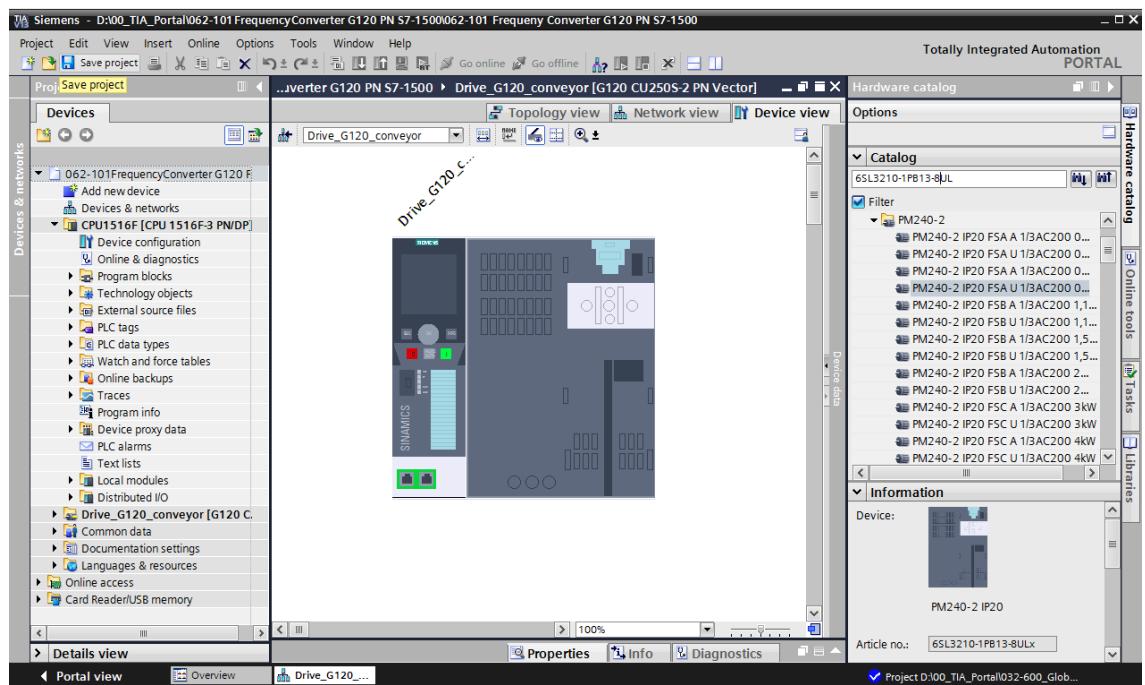


- Change to the 'Device view' from 'Drive_G120_conveyor'. There the used Power Module, for example 'PM240-2 IP20 FSA U 1/3 AC200 0.75kW', is selected and assigned to the 'Drive_G120_conveyor'.
 (→ Device view → Drive_G120_conveyor → PM 240-2 IP20 FSA U 1/3 AC200 0.75kW)

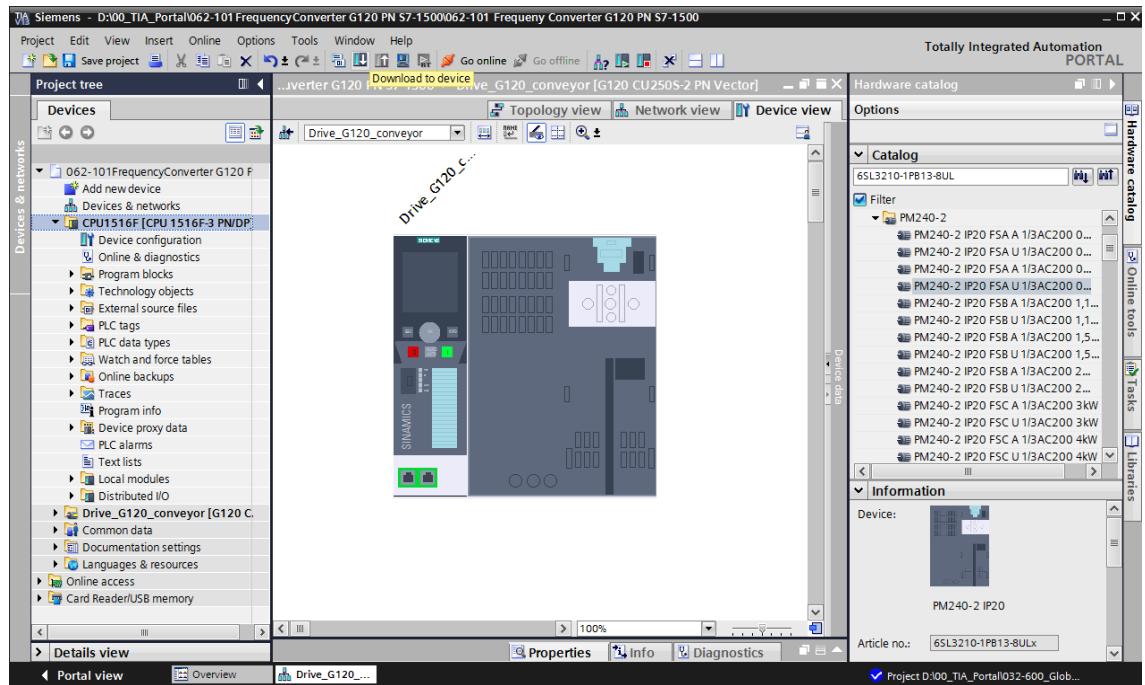


- Save the project with the existing settings.

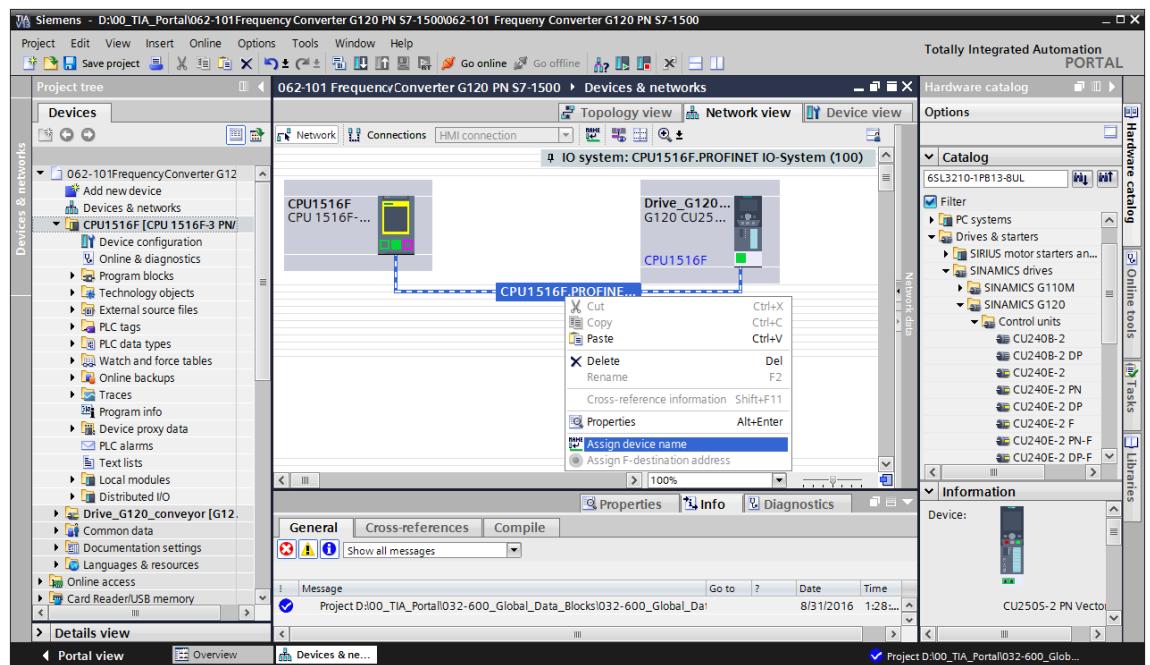
(→ Save project)



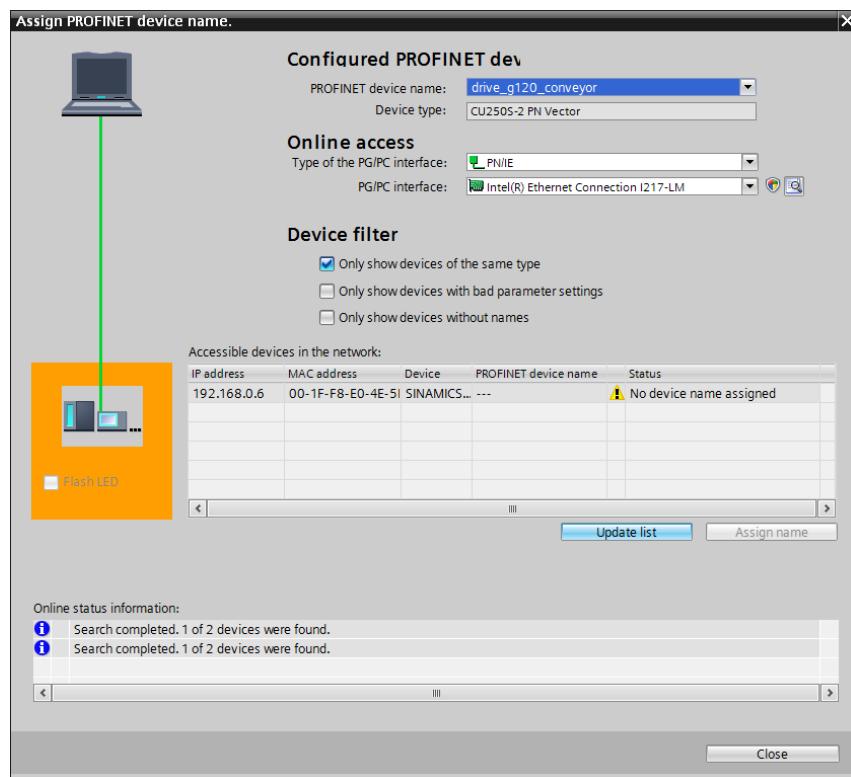
- Download the device configuration with the frequency converter G120 as the device to the 'CPU_1516F [CPU1516F-3 PN/DP]' by clicking the 'Download to device' icon.
(→CPU_1516F [CPU1516F-3 PN/DP] →)

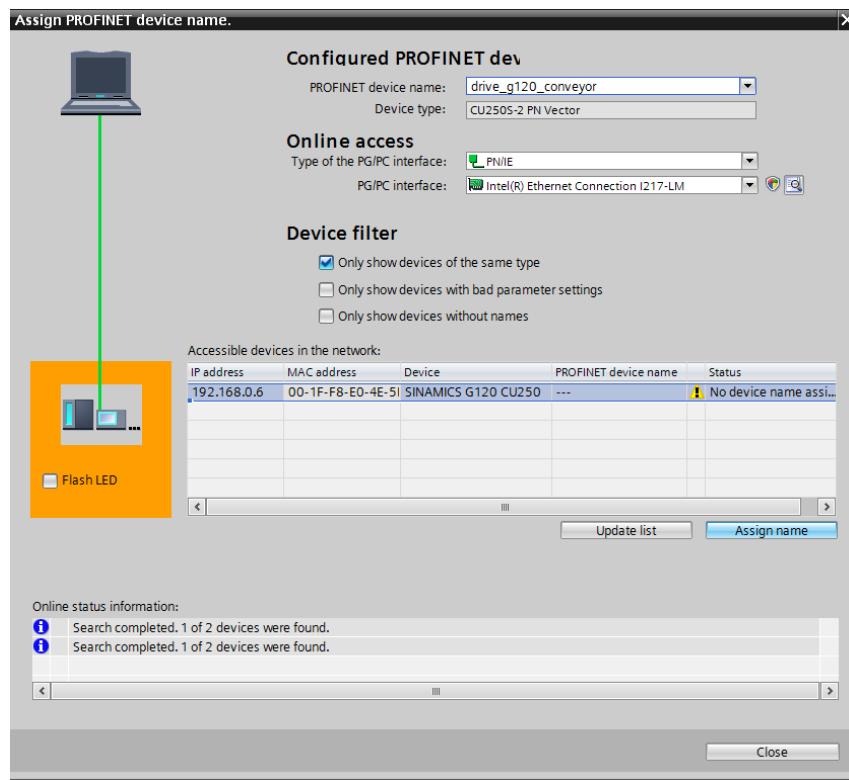


- The device name still has to be assigned to the frequency converter G120 as an IO device of the CPU_1516F. To do so, select the 'PN/IE_1' network and select 'Assign device name'.
(→ PN/IE_1 → Assign device name)



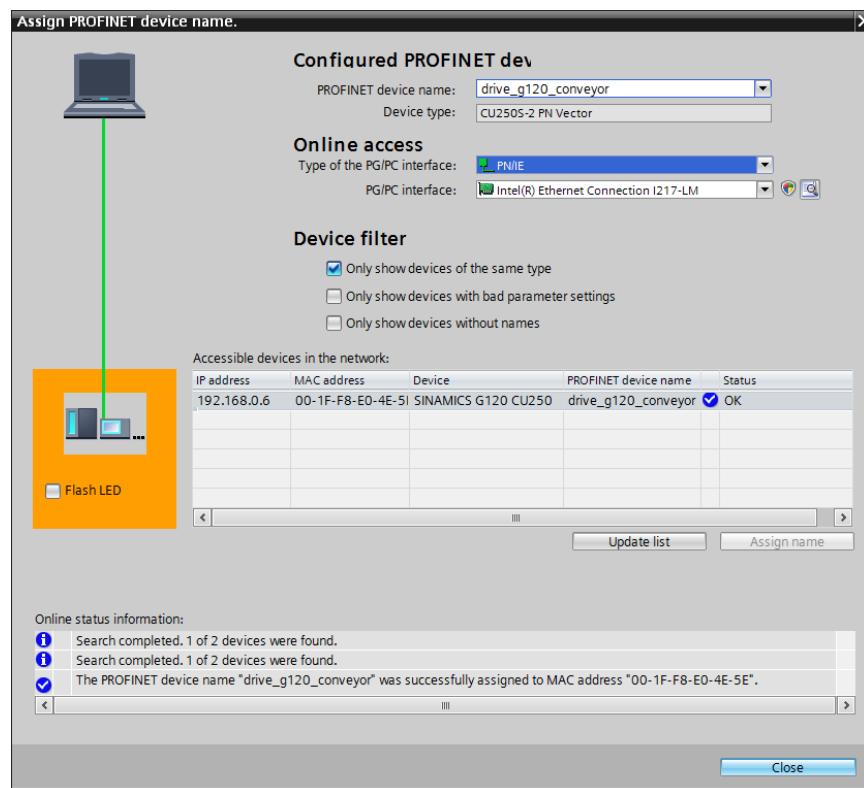
- In the subsequent dialog, the "PG/PC interface" can be selected, before we select the 'Drive_G120_conveyor' and 'Assign name'.
 (→ PROFINET device name: Drive_G120_conveyor → SINAMICS G120 CU250S → Assign device name)





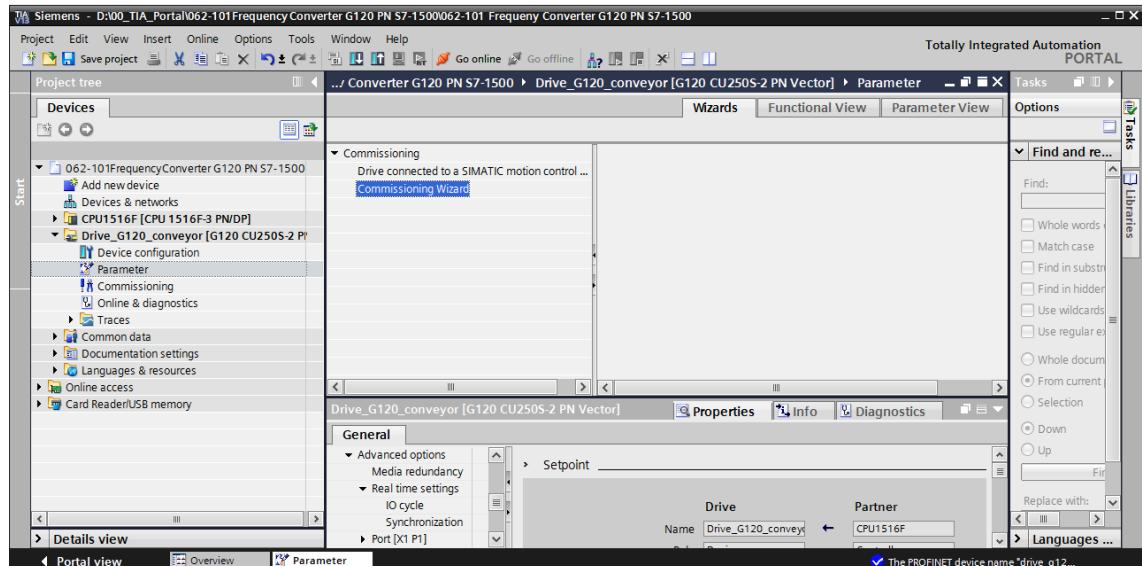
Note: If several IO devices exist in the network, the device can be identified on the basis of the imprinted MAC address.

- If too many components are displayed, the view can be filtered by clicking on 'Only show devices of the same type'. If the device name was assigned successfully, this is indicated by 'OK' in the status. (→ Close)

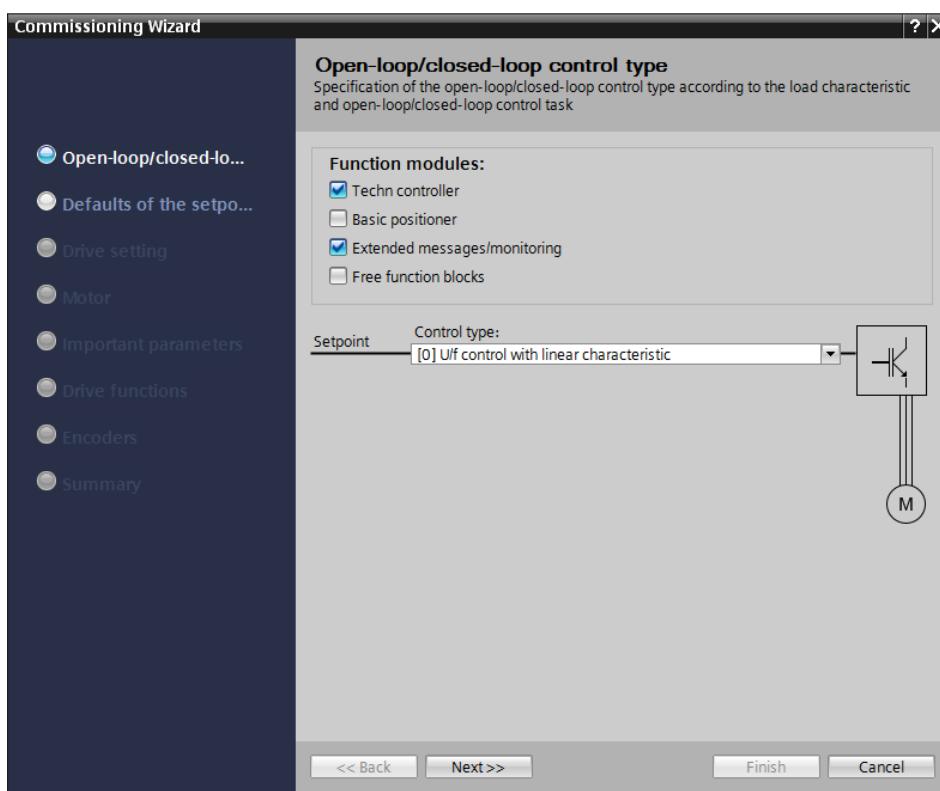


7.3 Assigning parameters with the commissioning wizard

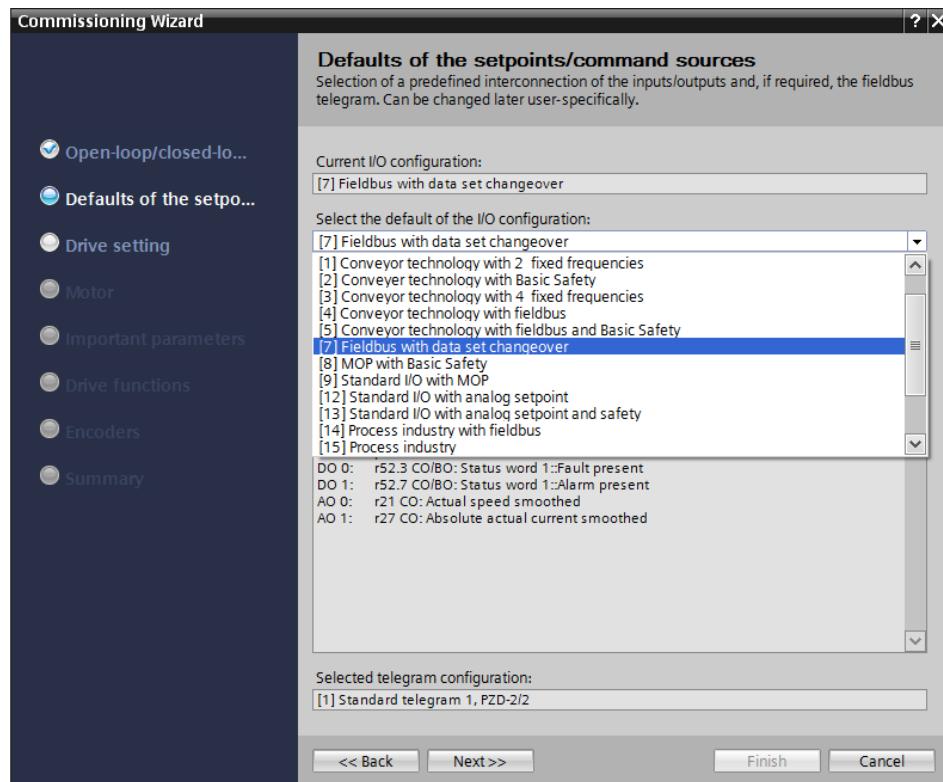
- To assign the parameters of the frequency converter, double-click 'Parameter' of 'Drive_G120_conveyor' to open the parameters and start the 'Commissioning Wizard'.
(→ Drive_G120_conveyor → Parameter → Commissioning Wizard)



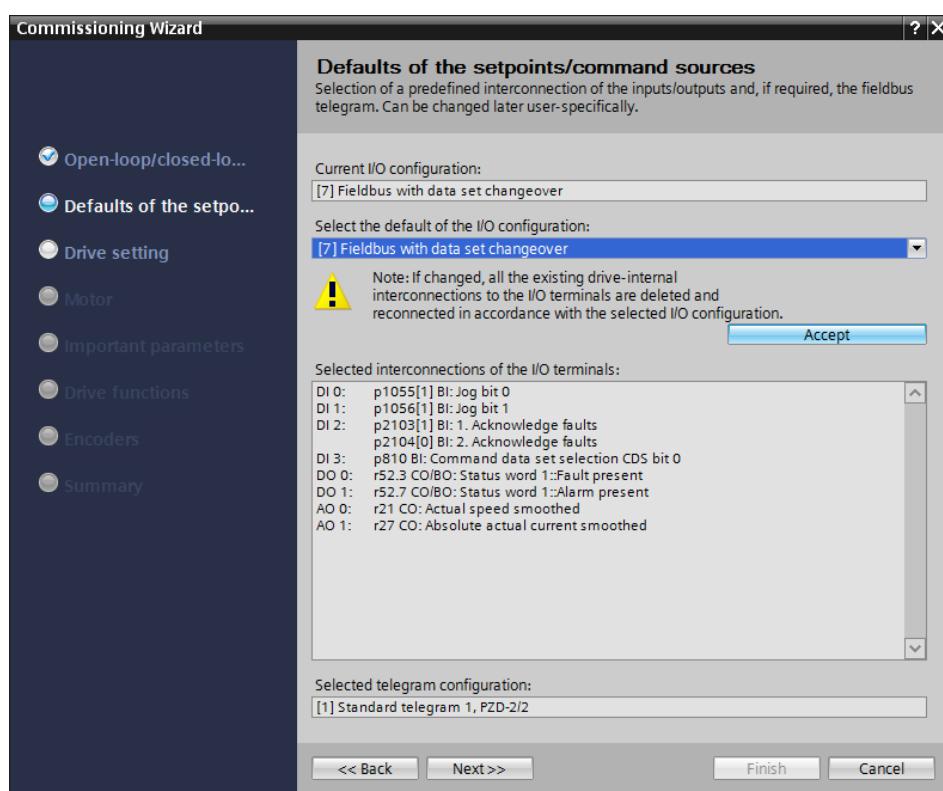
- In the subsequent dialog, select 'U/f control with linear characteristic' as the control type.
Keep the default selection for the function modules.
(→ U/f control with linear characteristic → Next)



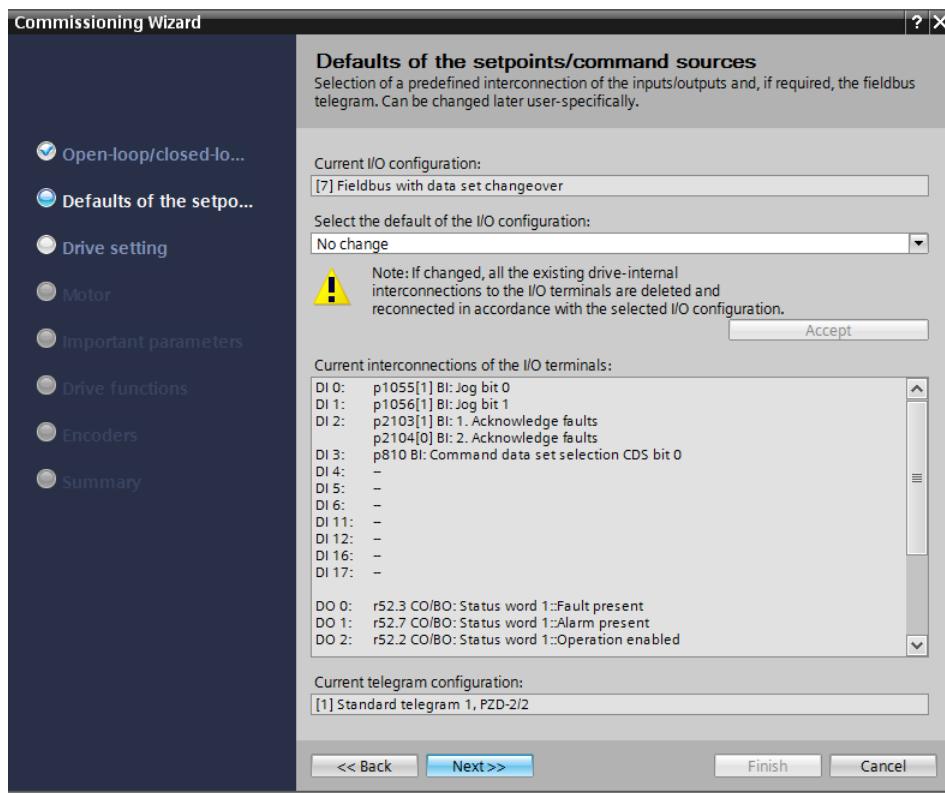
- For selection of the setpoint and command source, select the macro 7 'Fieldbus with data set changeover'. (→[7] Fieldbus with data set changeover)



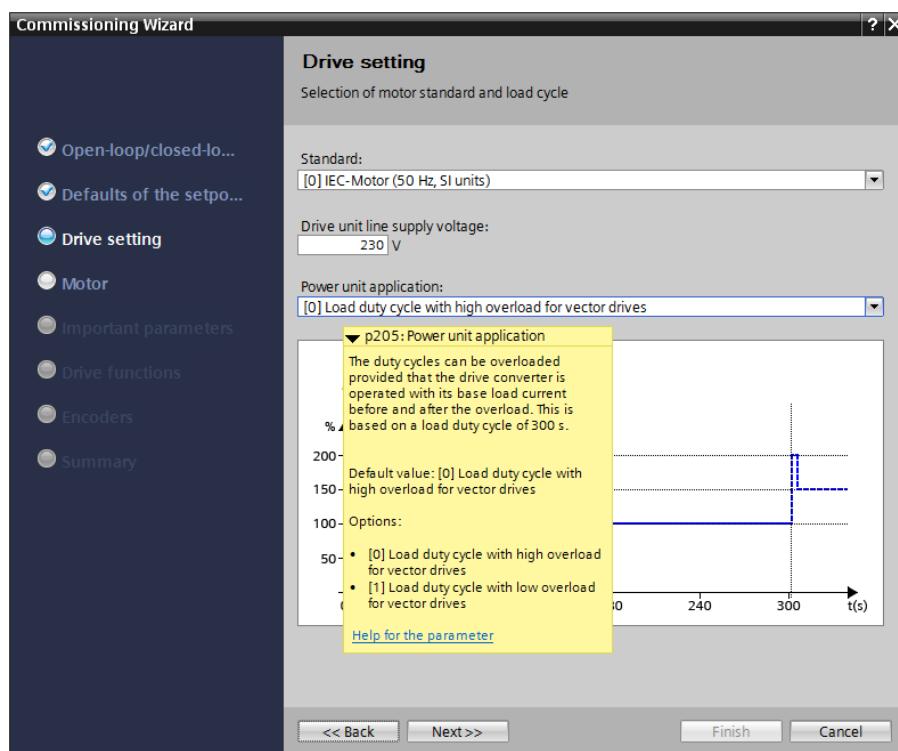
- The selection of the macro '[7] Fieldbus with data set changeover' still has to be confirmed with 'Accept'.
 (→ Accept)



- The current interconnections of the IO terminals for the Macro 7 are now displayed.
 (→ Next)

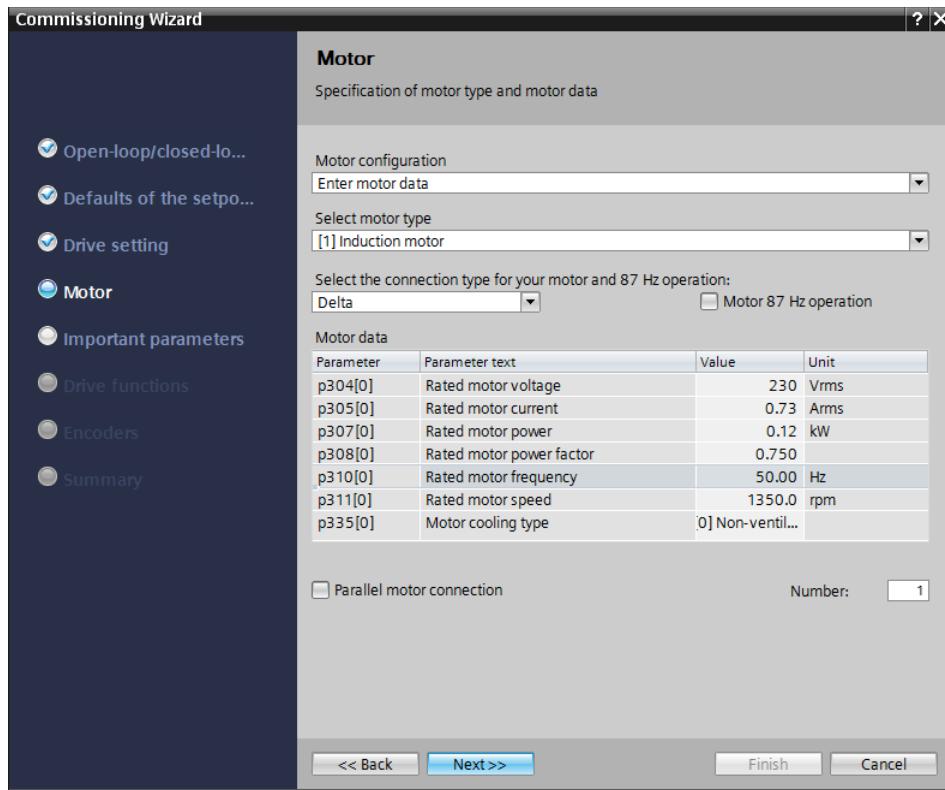


- For the drive settings, select the 'IEC-Motor (50 Hz, SI units)' and 'Load duty cycle with high overload for vector drives'.
 (→ IEC-Motor (50 Hz, SI units) → Load duty cycle with high overload for vector drives → Next)



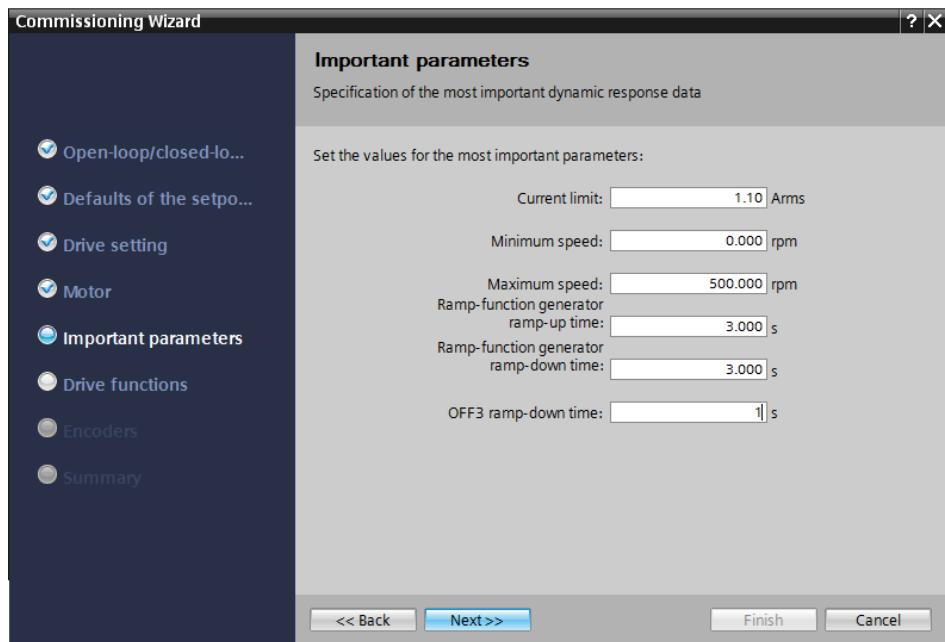
Note: Further information about the settings is available in the tool tip text, the online help or in the list manual.

- In the subsequent dialog, select 'Induction motor' as the motor type and enter the motor data in accordance with the specifications of the rating plate of the motor
 (→ Enter motor data → Induction motor → Connection type: Delta → ... → Next)



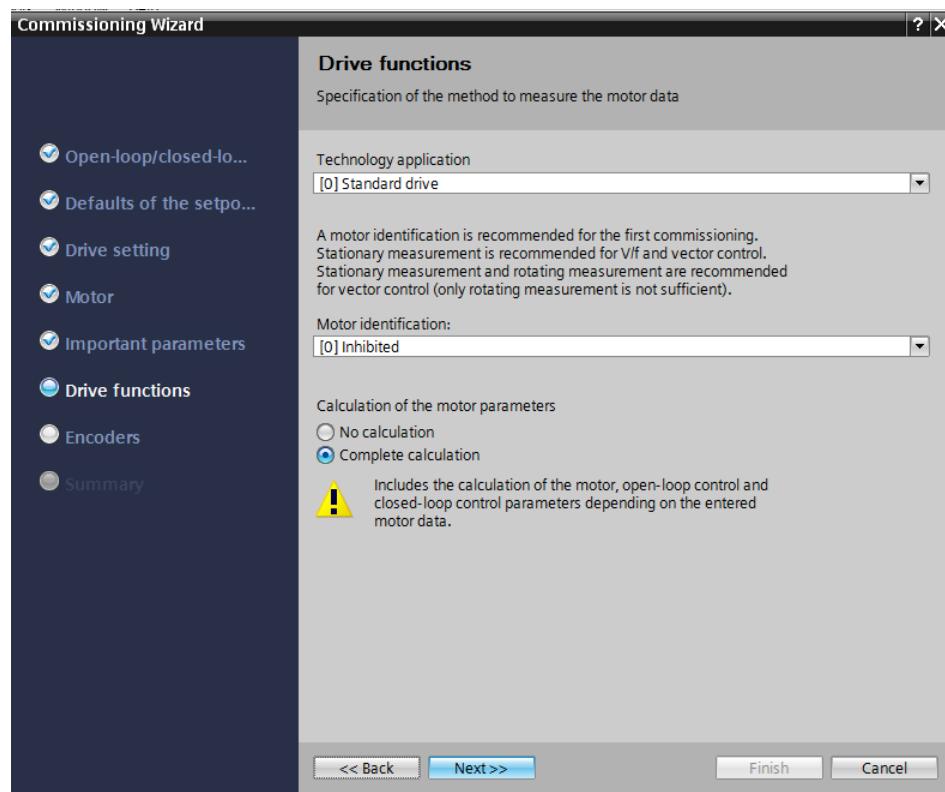
Note: Alternatively SIEMENS motors can also be selected directly via the order numbers.

- The following screenshot shows an example for the parameters for the current/speed limiting and for the ramp-function generator.
 (→ Next)



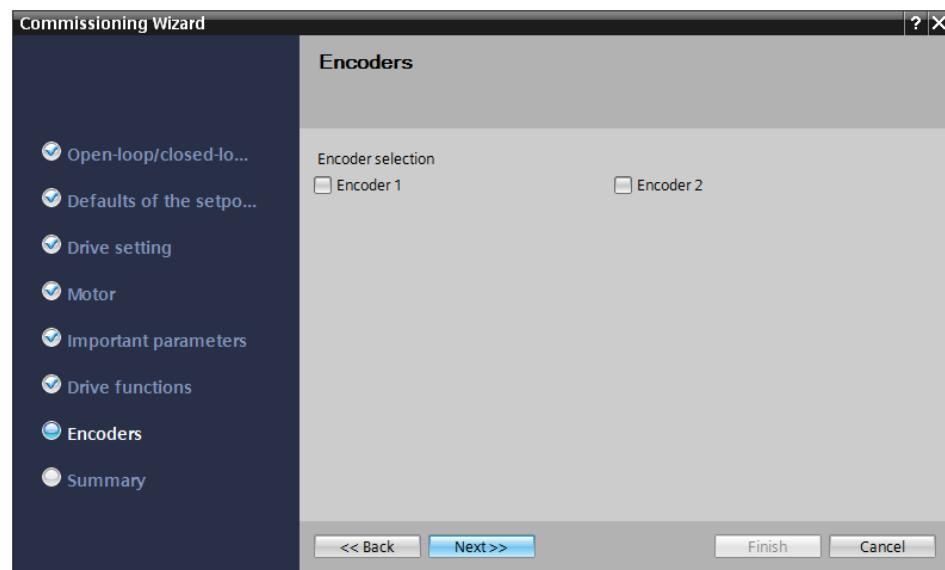
- Select 'Standard drive' for the Technology application. Set the motor identification to 'Inhibited', and select 'Complete calculation' for calculating the motor parameters based on parameter values from before.

(→ Standard drive → Motor identification: Inhibited → Complete calculation → Next)



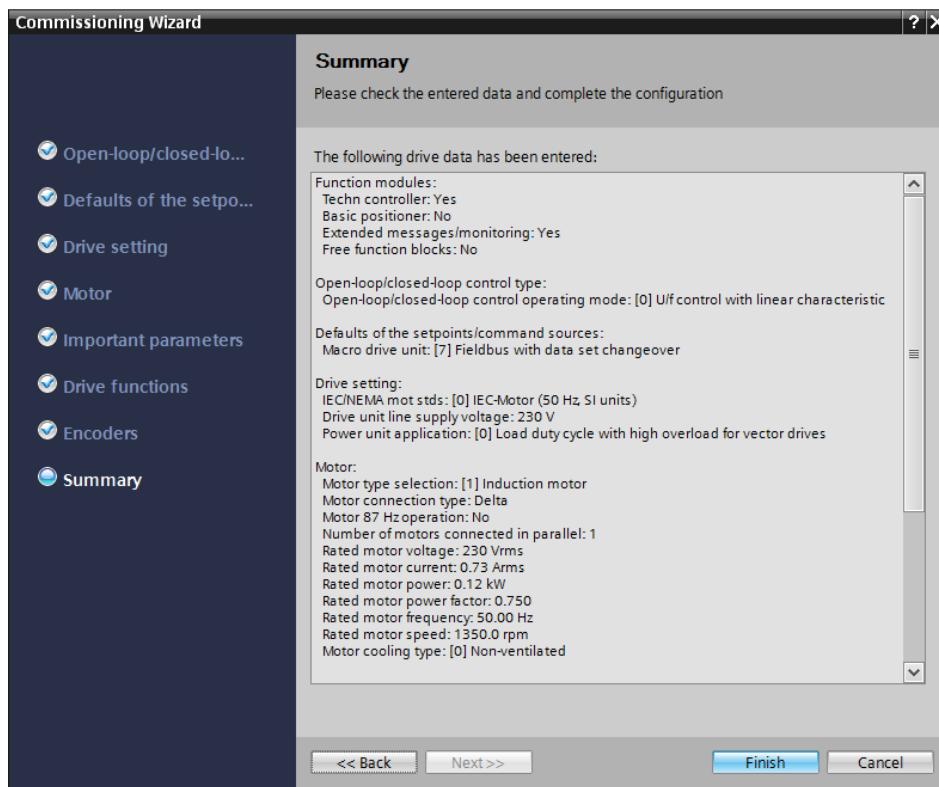
- Do not select an encoder at this point.

(→ Next)



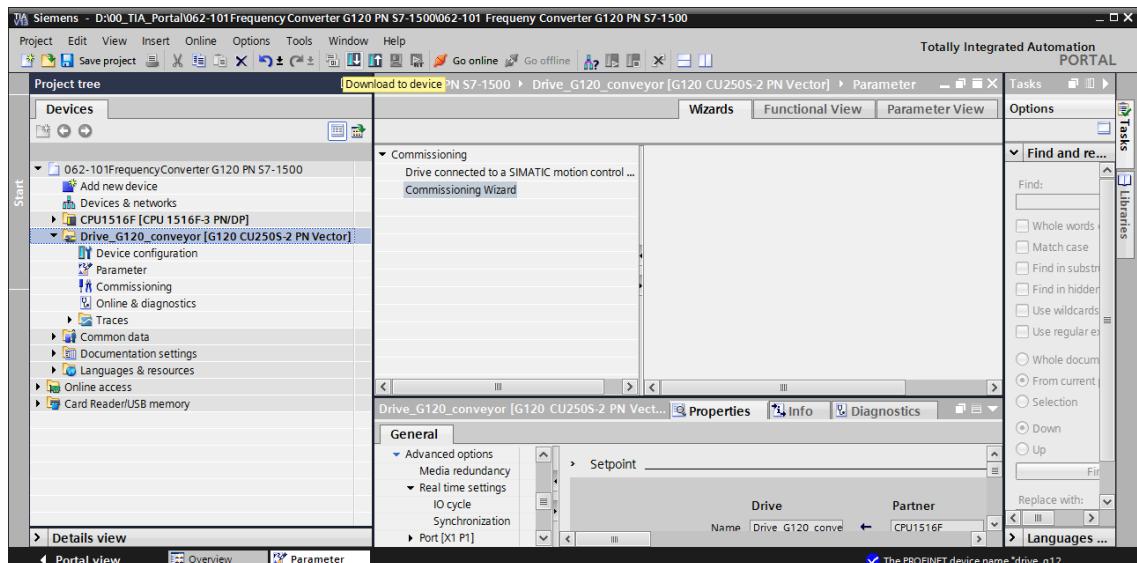
- In the subsequent summary, all the settings are shown once more for checking. These are applied by using the 'Finish' button.

(→ Finish)



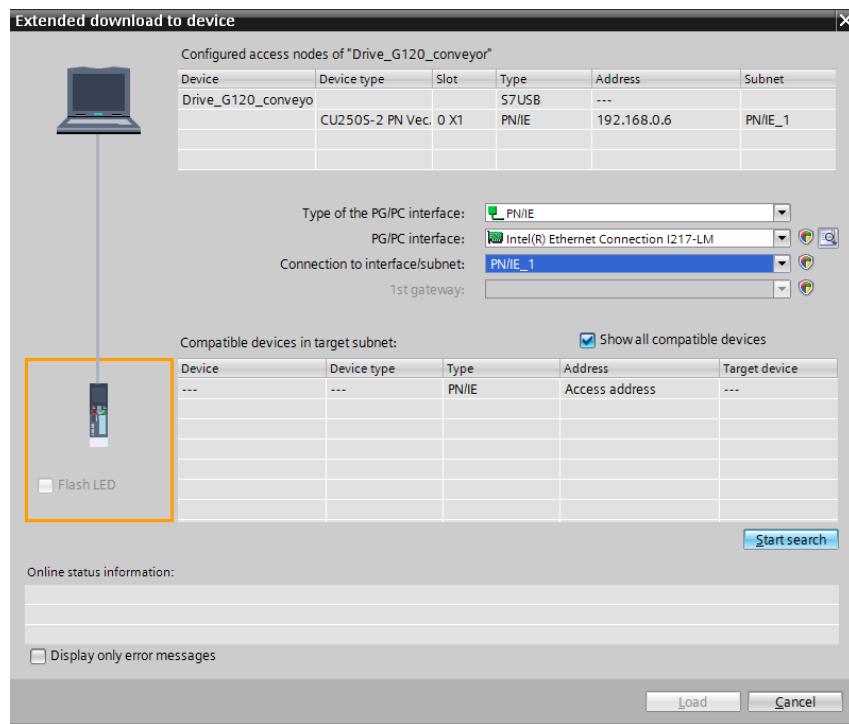
→ Save the project once more before downloading the parameters into the

'Drive_G120_conveyor', . (→ → Drive_G120_conveyor →)

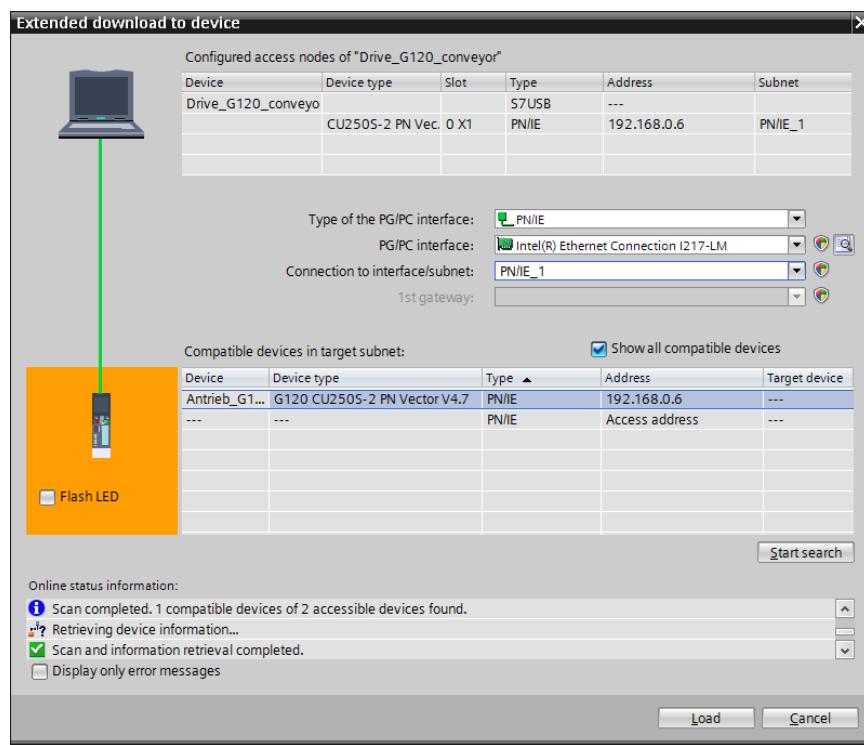


→ In the subsequent dialog, select 'PN/IE' as the PG/PC interface type, select the previously set network adapters as the PG/PC interface and select 'PN/IE_1' as the connection of the CPU to the subnet. Click 'Start search'.

(→ Type of the PG/PC interface: PN/IE → PG/PC interface: → Connection to interface/subnet: PN/IE_1 → Start search)

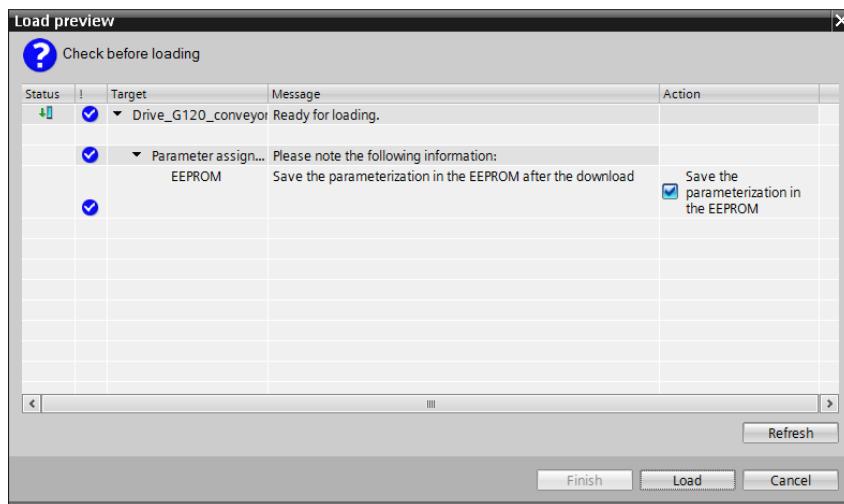


- You should now see your 'SINAMICS drive' and be able to select it as the target device.
Click 'Load'. (→ SINAMICS drive → Load)



The configuration is compiled automatically and is displayed once more in an overview so that you can check the steps to be carried out before loading. Now select ' Save the parameterization in the EEPROM' and click 'Load'.

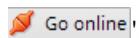
(→ Save the parameterization in the EEPROM → Load)



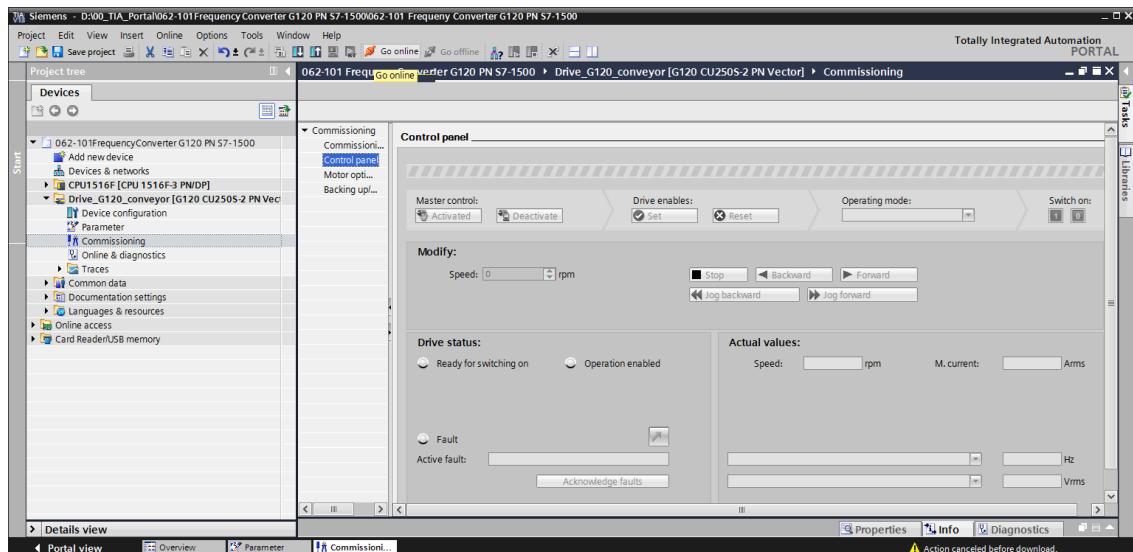
Note: It is advisable to back up the parameters in the EEPROM as well, so that these are retained in the case of a voltage drop

7.4 Testing and commissioning with control panel

- In order to test the current parameter assignment without PLC program, open the 'Control panel' from the 'Commissioning' menu of the 'Drive_G120_conveyor'. Finally, click '

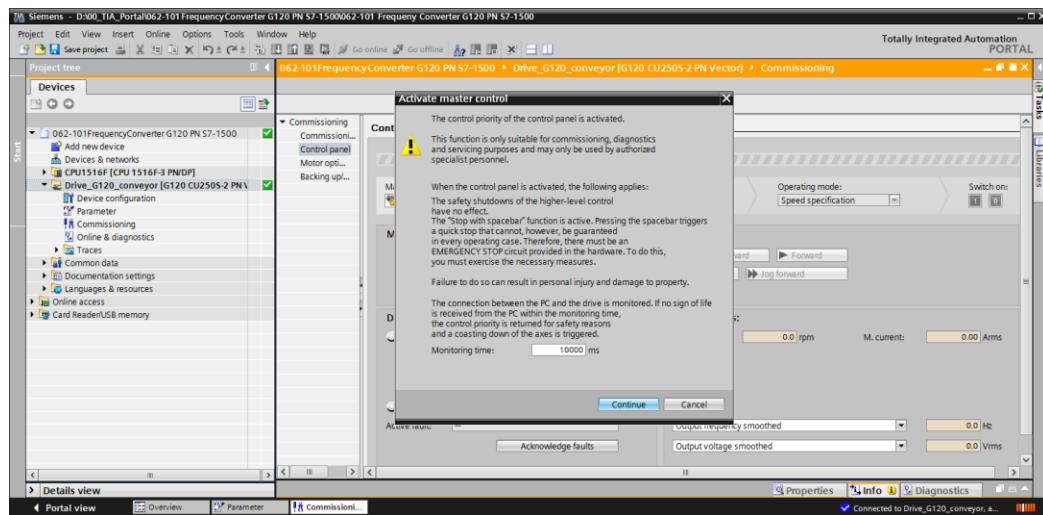


(→ Drive_G120_conveyor → Commissioning → Control panel) (→ )

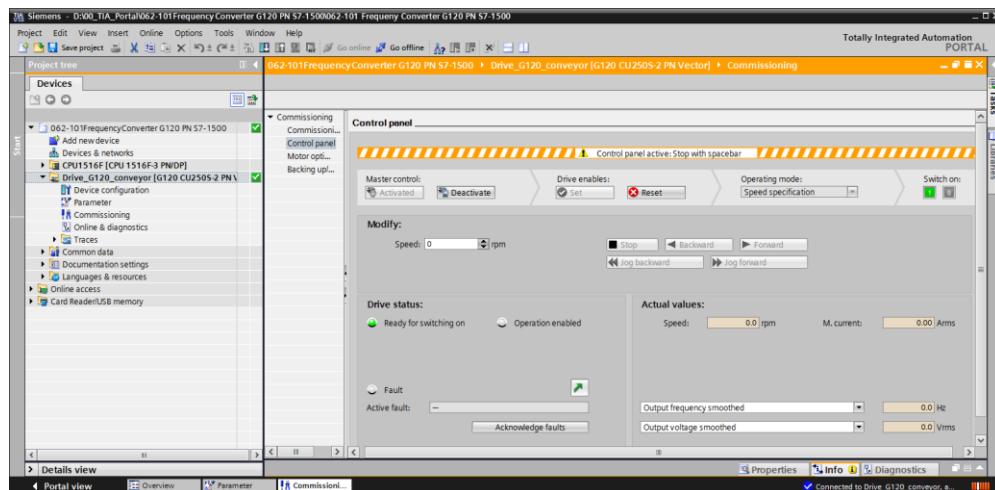


- The first step is to 'Activate master control' in the control panel. The communication between the PC and the converter will then be monitored. It is necessary that successful communication takes place at least every 10000 ms. Otherwise the motor stops and the enables are reset.

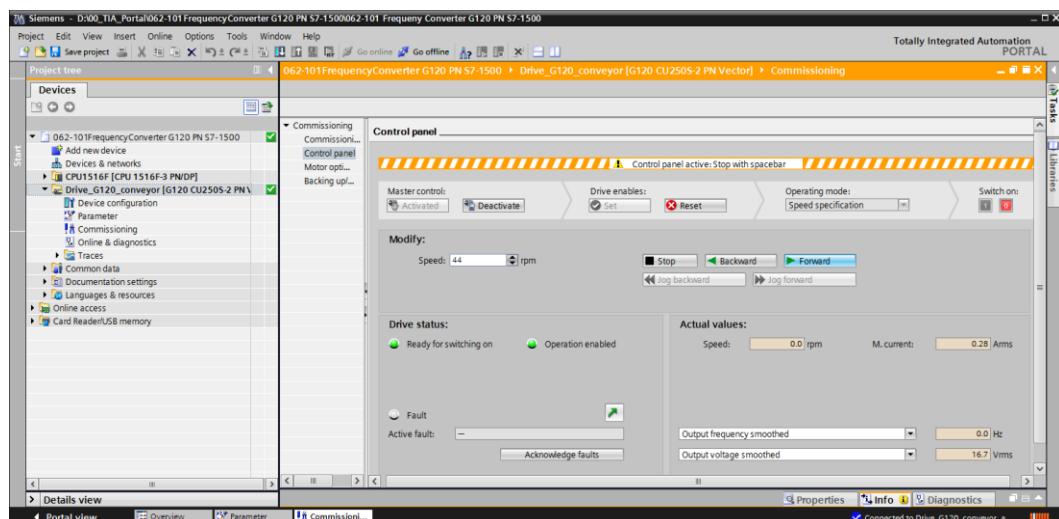
(→ Master control:  → 10000 ms → )



- The drive enables first have to be set in order to start the motor . As a rule this happens automatically. The drive can then be switched on .
(→ Switch on)

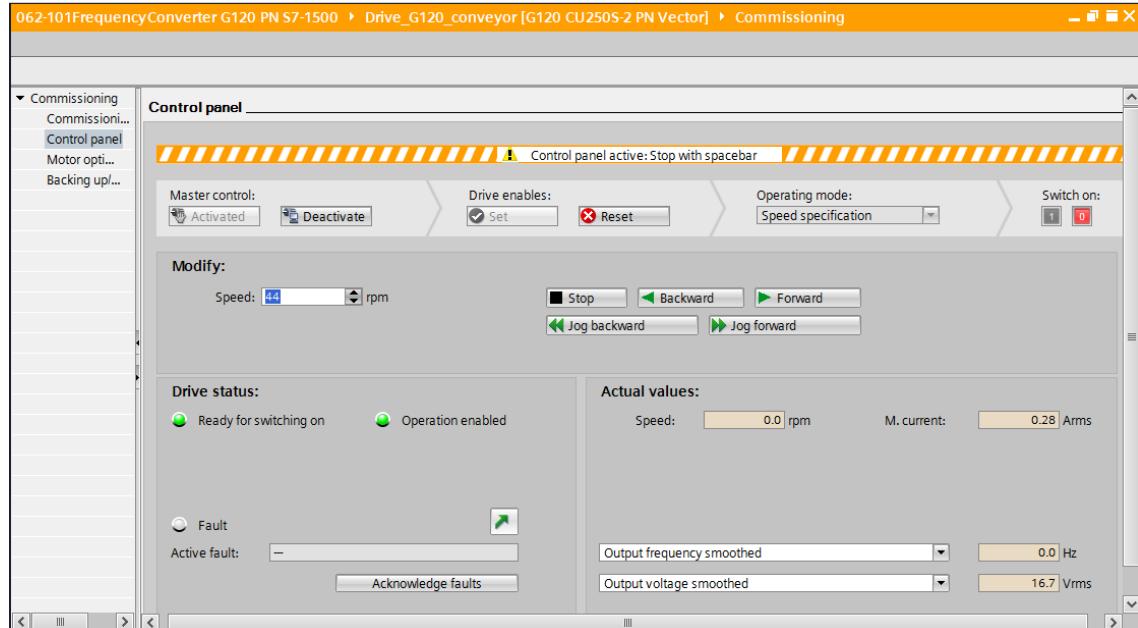


- Now the motor can be run at the selected speed or .
(→ Speed: 44 →)



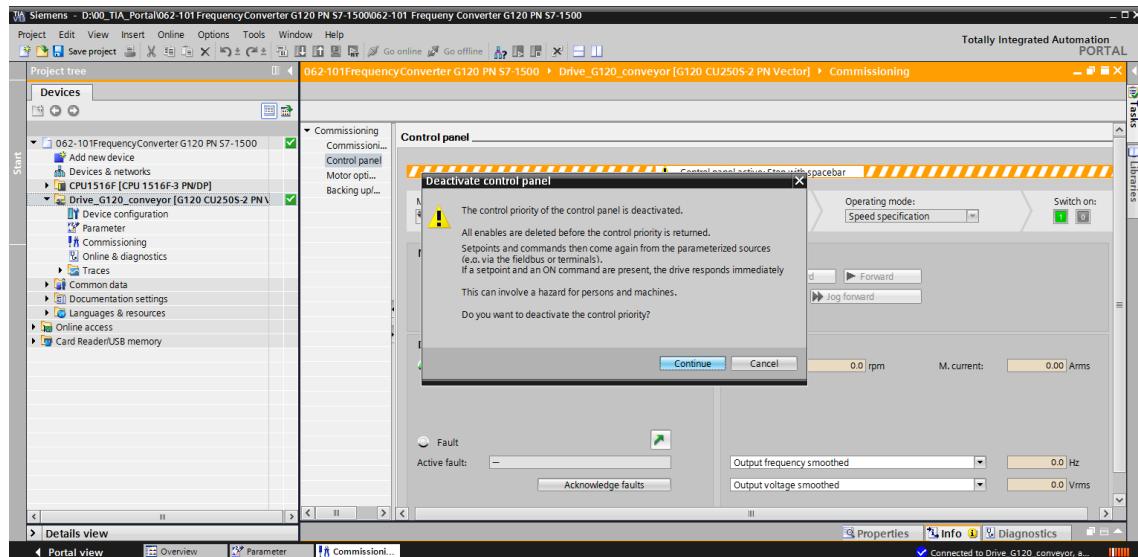
- The drive can be switched off by clicking '0'. After completion of the test, it is necessary to the master control.

(→ →



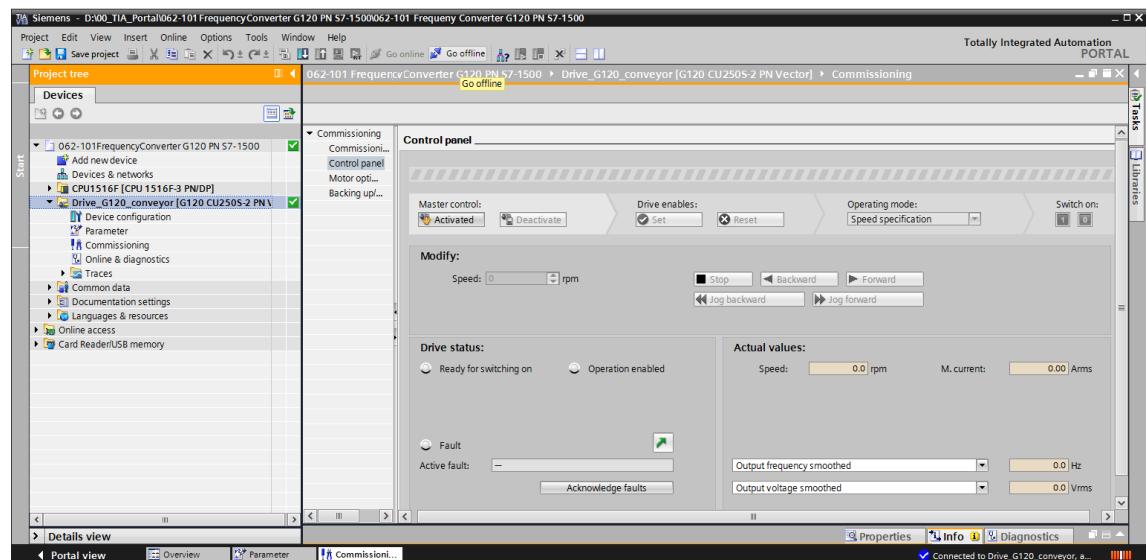
- Confirm the prompt for deactivation with .

(→



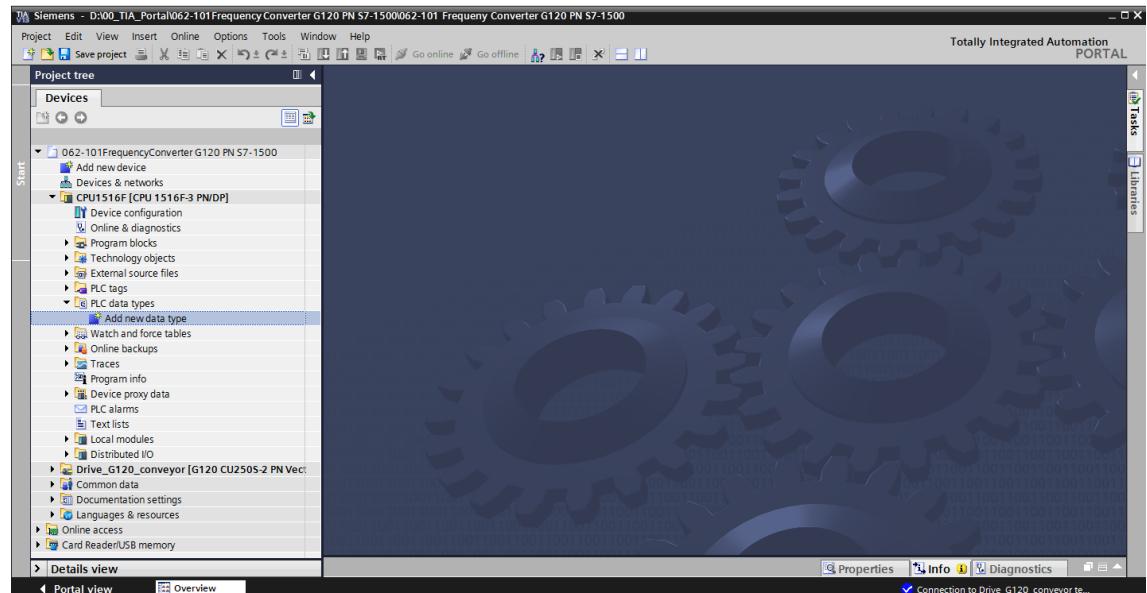
- Finally, and save the project again .

(→ →

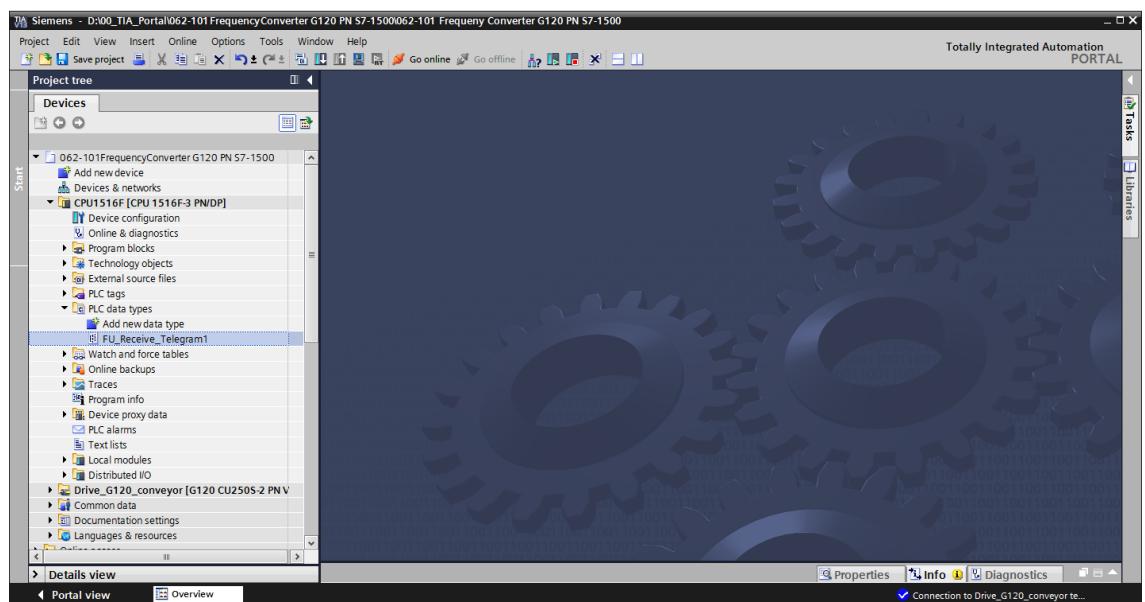


7.5 Creating a program for controlling the frequency converter

- Before you adapt the program so that it can control the frequency converter, two 'PLC data types' have to be created that correspond to the structure of the send and receive Telegram 1.
- (→ PLC data types → Add new data type)



- Change the name of the PLC data type to 'FU_Receive_Telegram1' and open it by double-clicking it.
- (→ FU_Receive_Telegram1)

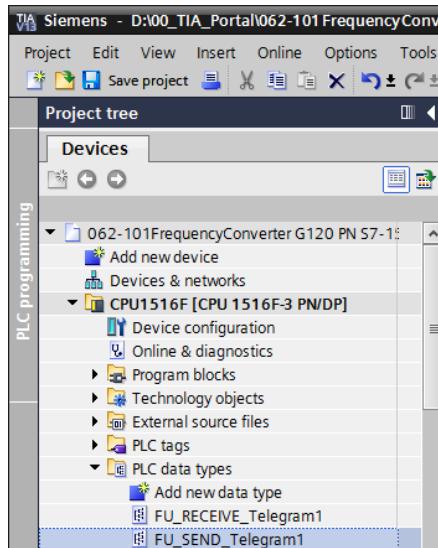


- Create the tags shown below the same as in a data block. (→ FU_Receive_Telegram1)

062-101 Frequency Converter G120 PN S7-1500 > CPU1516F [CPU 1516F-3 PN/DP] > PLC data types > FU_RECEIVE_Telegram1							
	Name	Data type	Default value	Accessible f...	Visible in ...	Setpoint	Comment
1	Speed_OK	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Setpoint / actual speed deviation within the tolerance range (1)
2	Control_requested	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	The automation system is requested to accept the inverter control(1)
3	Max_speed_reached	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Speed is greater than or equal to the maximum speed (1)
4	Warn_torque_limit	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Comparison value for current torque has been reached or exceeded (1)
5	Holding_brake	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Holding brake open(1)
6	Motor_temperature	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Alarm motor overtemperature(0)
7	Direction	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Motor rotates clockwise(1) / counterclockwise(0)
8	PM_overload	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Alarm inverter PM thermal overload (0)
9	Ready_to_Start	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Power supply switched on; electronics initialized; pulses locked(1)
10	Ready	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Motor is switched on (ON/OFF1 = 1), no fault is active(1)
11	Operation_EN	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Operation enabled Motor follows setpoint(1)
12	Fault	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Fault active(1)
13	No_OFF2	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Coast down to standstill is not active(1)
14	No_OFF3	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Quick stop is not active(1)
15	Lockout	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Closing lockout active(1)
16	Alarm	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Alarm active(1)
17	XIST_A	Int	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Actual speed value process data (PZD) word2

- Create an additional PLC data type called 'FU_Send_Telegram1' and the tags shown below.

(→ FU_Send_Telegram1)

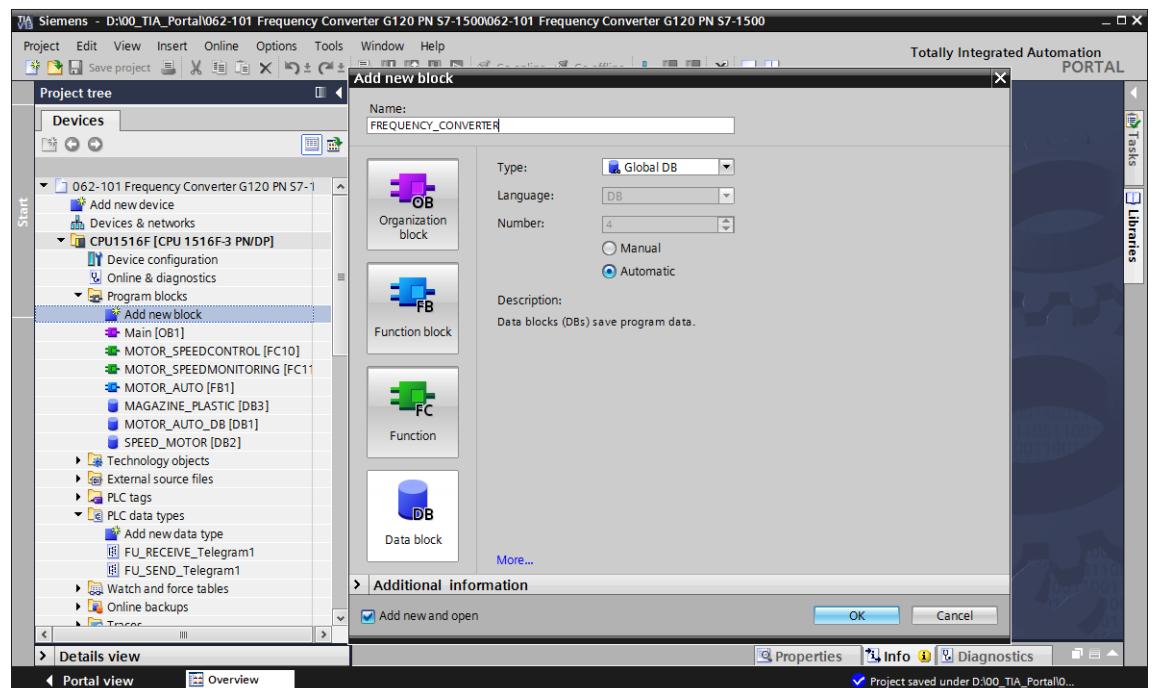


062-101 Frequency Converter G120 PN S7-1500 > CPU1516F [CPU 1516F-3 PN/DP] > PLC data types > FU_SEND_Telegram1

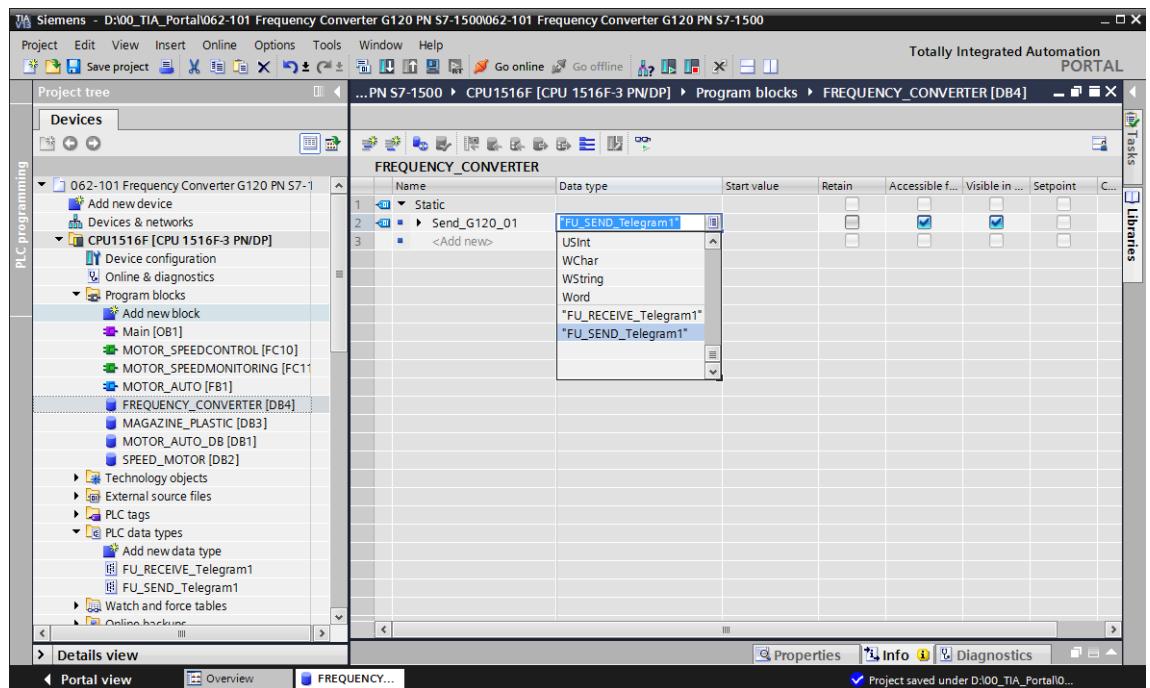
Name	Data type	Default value	Accessible ...	Visible in ...	Setpoint	Comment
1 reserved_8	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	not in use
2 reserved_9	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	not in use
3 Control_via_PL�	Bool	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Control via fieldbus, inverter accepts the process data from fieldbus(1)
4 Rev_direction	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Invert setpoint in the inverter(1)
5 reserved_12	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	not in use
6 MOP_up	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Increase the setpoint saved in the motorized potentiometer(1)
7 MOp_down	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Reduce the setpoint saved in the motorized potentiometer(1)
8 reserved_15	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	not in use
9 ON_OFF1	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	ON(1) / OFF(0) with the ramp-function generator
10 ON_OFF2	Bool	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Switch OFF (0); Switch off the motor immediately, the motor coasts down to standstill
11 ON_OFF3	Bool	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Switch OFF (0); Quick stop, the motor brakes with the OFF3 ramp-down time
12 EN_operation	Bool	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Enable operation; Switch-on motor (pulses can be enabled) (1)
13 EN_ramp	Bool	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Enable ramp-function (1) / Reset ramp-function generator output to 0 (0)
14 Continue_freeze_ramp	Bool	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Enable ramp-function (1) / Freeze ramp-function generator (0)
15 Enable_setpoint	Bool	TRUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Enable setpoint(1) / Inhibit setpoint(0)
16 Acknowledge	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Acknowledge faults (1)
17 NSOLL_A	Int	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Setpoint speed process data (PZD) word2

Note: For some enable bits, the start value is already set to TRUE so that these do not have to be set additionally in the program.

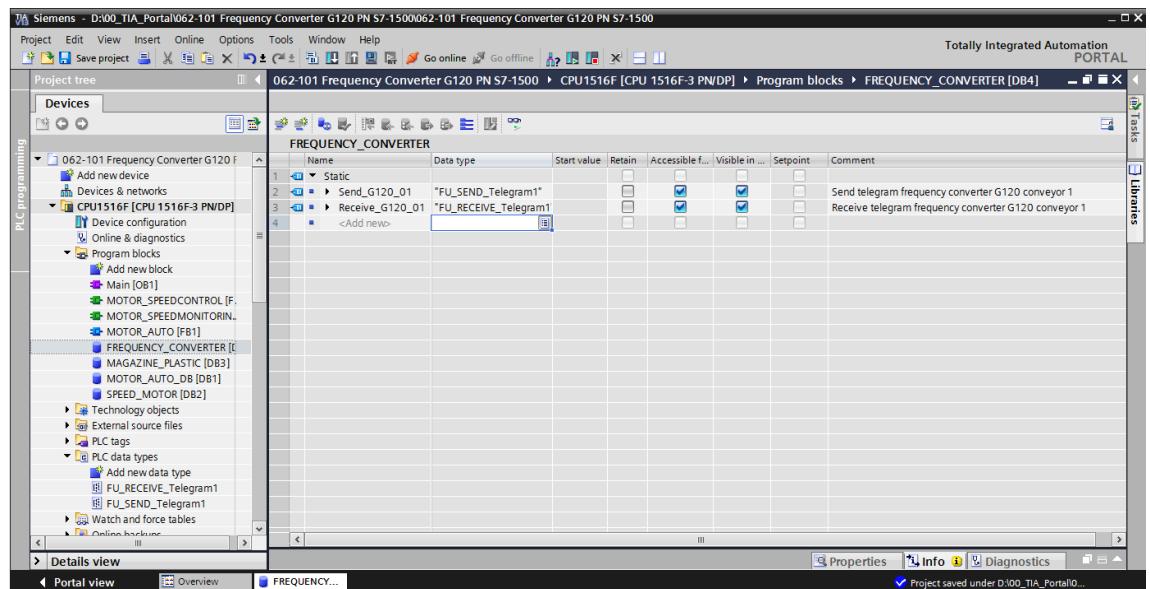
- Create the global data block 'FREQUENCY_CONVERTER' for the request and response telegram.
(→Add new block → DB → Global DB → FREQUENCY_CONVERTER → OK)



- Create the tag 'Send_G120_01' and select 'FU_SEND_Telegram1' as the data type.
(→ Send_G120_01 → "FU_SEND_Telegram1")



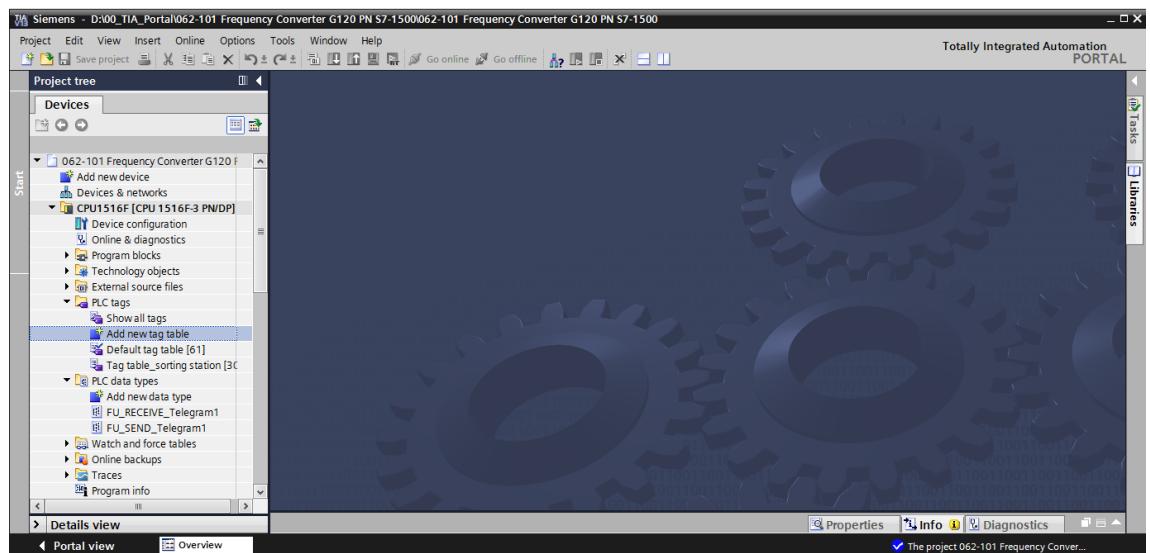
- Create an additional tag 'Receive_G120_01' and select 'FU_RECEIVE_Telegram1' as the data type. Provide comments for the two tags.
(→ Receive_G120_01 → 'FU_RECEIVE_Telegram1')



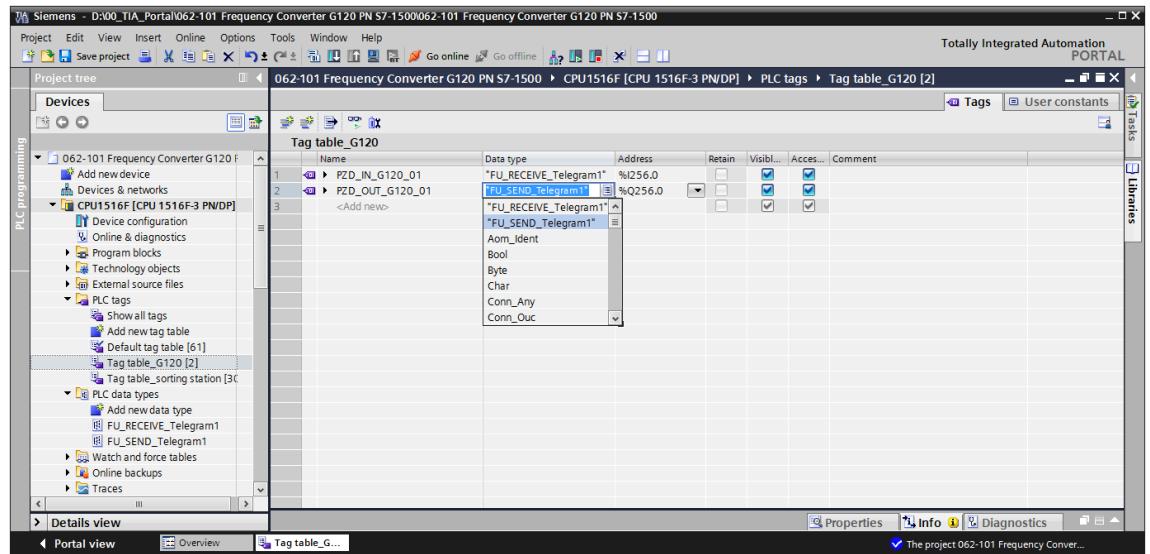
- The data block can be created quickly and efficiently by using the PLC data types 'FU_SEND_Telegram1' and 'FU_RECEIVE_Telegram1', see representation.

FREQUENCY_CONVERTER								
	Name	Data type	Start value	Retain	Accessible f...	Visible in ...	Setpoint	Comment
1	Static							
2	Send_G120_01	"FU_SEND_Telegram1"			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Send telegram frequency converter G120 conveyor 1
3	reserved_8	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		not in use
4	reserved_9	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		not in use
5	Control_via_PLC	TRUE			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Control via fieldbus, inverter accepts the process data from fieldbus(1)
6	Rev_direction	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Invert setpoint in the inverter(1)
7	reserved_12	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		not in use
8	MOP_up	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Increase the setpoint saved in the motorized potentiometer(1)
9	Mop_down	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Reduce the setpoint saved in the motorized potentiometer(1)
10	reserved_15	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		not in use
11	ON_OFF1	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		ON(1) / OFF(0) with the ramp-function generator
12	ON_OFF2	Bool	TRUE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Switch OFF (0); Switch off the motor immediately, the motor coasts down to stand.
13	ON_OFF3	Bool	TRUE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Switch OFF (0); Quick stop, the motor brakes with the OFF3 ramp-down time
14	EN_operation	Bool	TRUE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Enable operation; Switch-on motor (pulses can be enabled)(1)
15	EN_ramp	Bool	TRUE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Enable ramp-function (1) / Reset ramp-function generator output to 0 (0)
16	Continue_freeze...	Bool	TRUE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Enable ramp-function (1) / Freeze ramp-function generator (0)
17	Enable_setpoint	Bool	TRUE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Enable setpoint(1) / Inhibit setpoint(0)
18	Acknowledge	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Acknowledge faults (1)
19	NSOLL_A	Int	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Setpoint speed process data (PZD) word2
20	Receive_G120_01	"FU_RECEIVE_Telegram1"			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Receive telegram frequency converter G120 conveyor 1
21	Speed_OK	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Setpoint / actual speed deviation within the tolerance range (1)
22	Control_reques...	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		The automation system is requested to accept the inverter control(1)
23	Max_speed_rea...	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Speed is greater than or equal to the maximum speed (1)
24	Warn_torque_li...	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Comparison value for current torque has been reached or exceeded (1)
25	Holding_brake	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Holding brake open(1)
26	Motor_tempера...	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Alarm motor overtemperature(0)
27	Direction	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Motor rotates clockwise(1) / counterclockwise(0)
28	PM_overload	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Alarm inverter PM thermal overload (0)
29	Ready_to_Start	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Power supply switched on; electronics initialized; pulses locked(1)
30	Ready	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Motor is switched on (ON/OFF1 = 1), no fault is active(1)
31	Operation_EN	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Operation enabled Motor follows setpoint(1)
32	Fault	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Fault active(1)
33	No_OFF2	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Coast down to standstill is not active(1)
34	No_OFF3	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Quick stop is not active(1)
35	Lockout	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Closing lockout active(1)
36	Alarm	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Alarm active(1)
37	XIST_A	Int	0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Actual speed value process data (PZD) word2

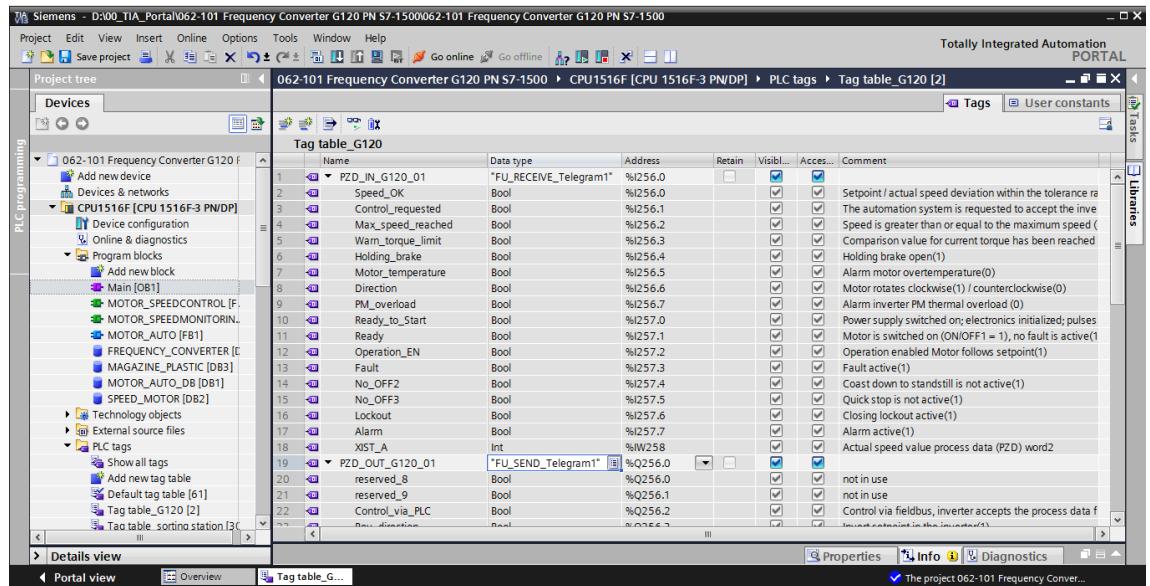
- The global PLC tags are created in a new tag table for the communication with the frequency converter.
 (→ Add new tag table)



- Change the name of the tag table to 'Tag_table_G120' and specify, as shown, two structure tags 'PZD_IN_G120_01' and 'PZD_OUT_G120_01' using the PLC data types 'FU_RECEIVE_Telegram1' and 'FU_SEND_Telegram1'.
 (→ PZD_IN_G120_01 → 'FU_RECEIVE_Telegram1' → PZD_OUT_G120_01 → 'FU_SEND_Telegram1')

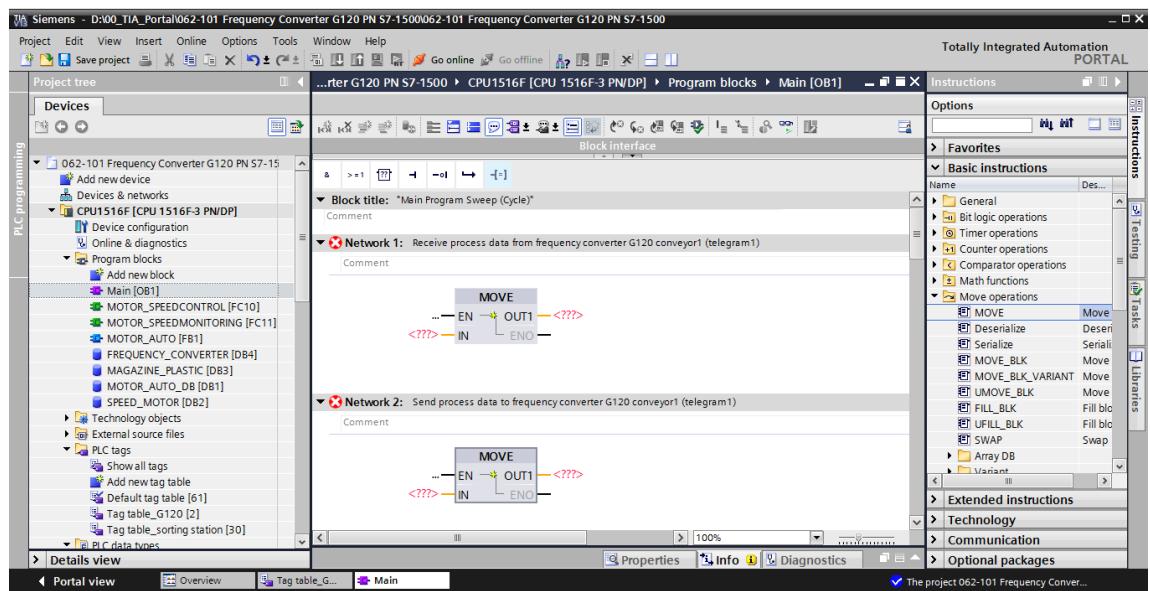


- The tags have been created in accordance with their structures by the use of the PLC data types 'FU_SEND_Telegram1' and 'FU_RECEIVE_Telegram1'. Open the 'Main' block [OB1].
(→ Main [OB1])

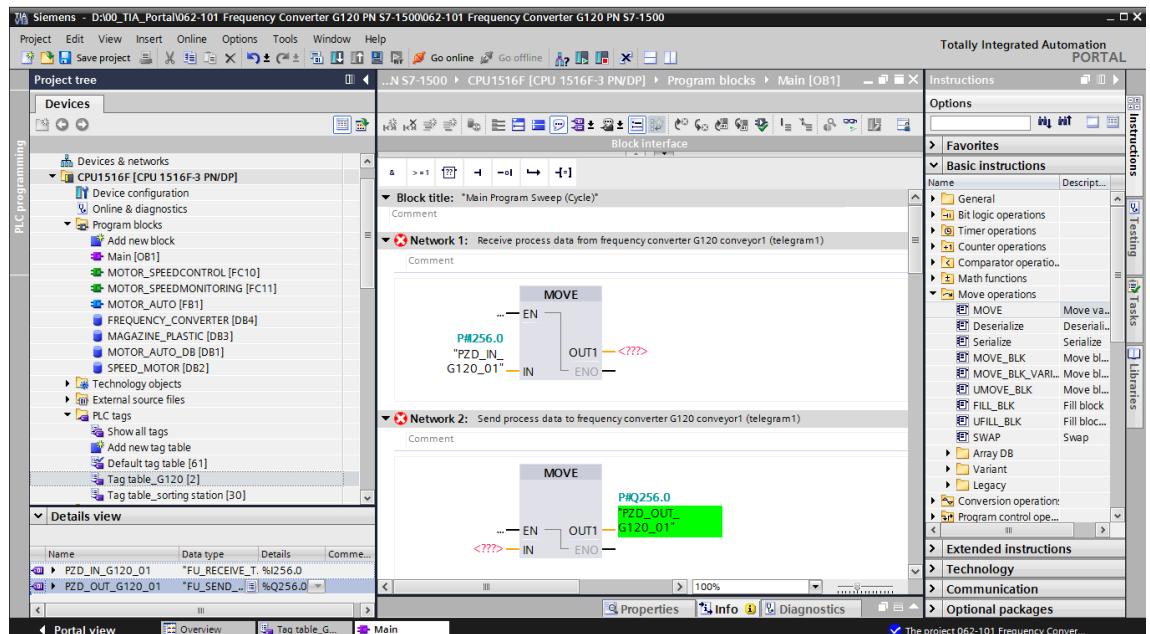


- Insert two new networks at the beginning of the Main [OB1]. Drag-&-drop the 'Move' command from the "Instructions" under the 'Move operations' item into these networks.

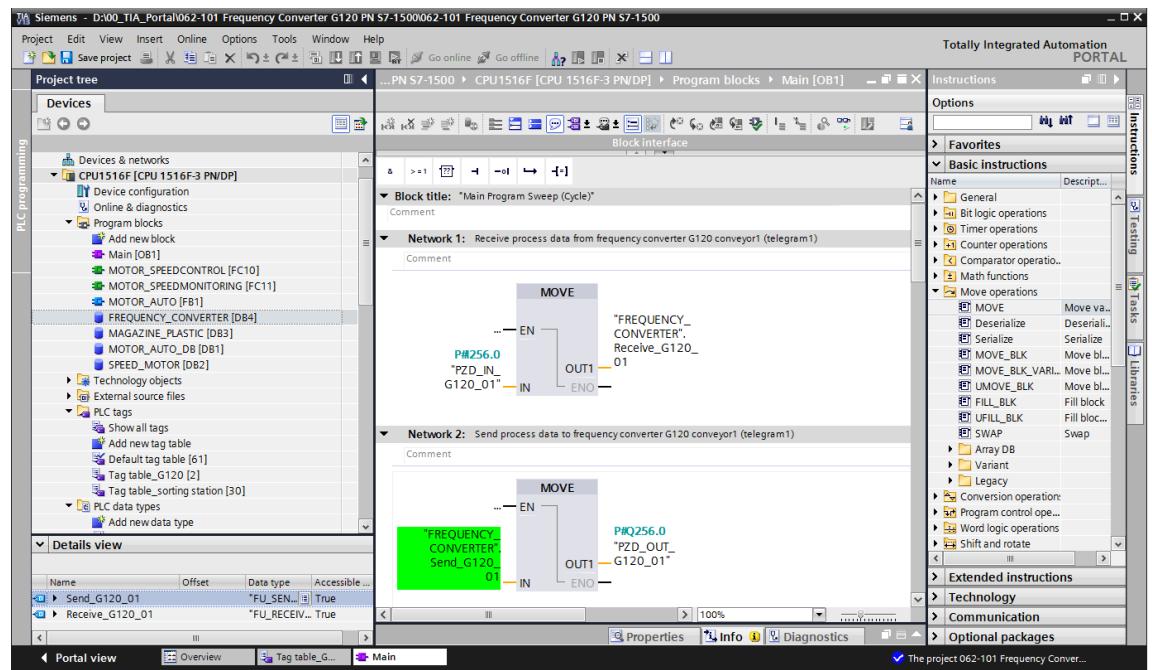
(→ → → Instructions → Move operations → Move → Move)



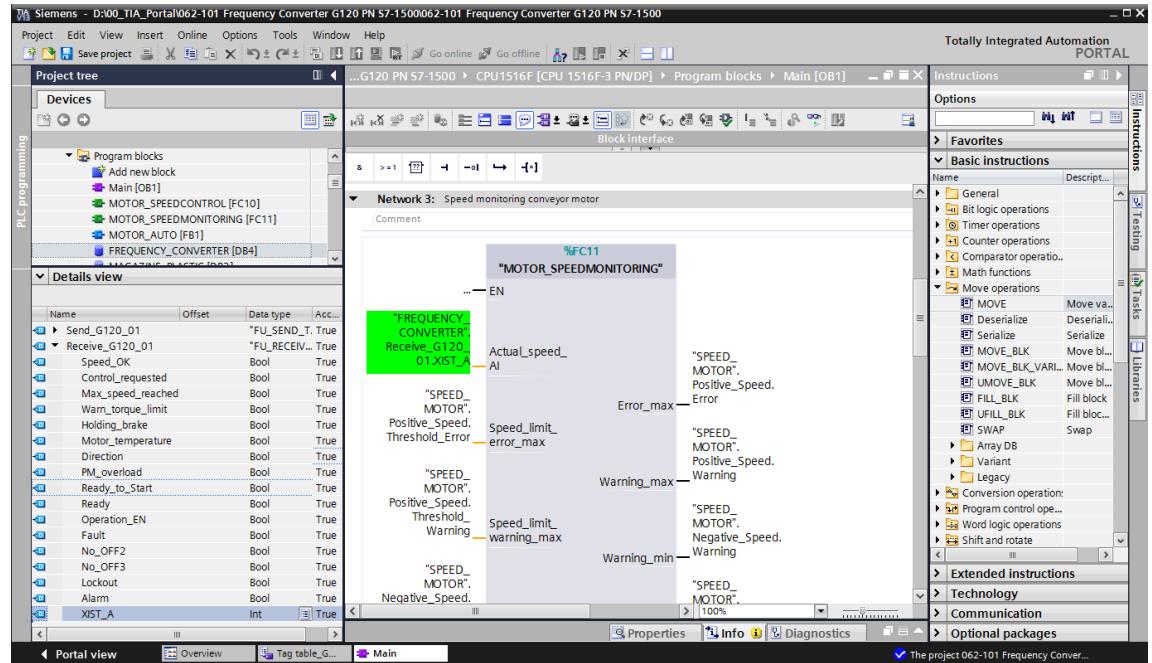
- Select the 'Tag_table_G120' in the project navigation. Now you can drag-&-drop the two tags 'PZD_IN_G120_01' and 'PZD_OUT_G120_01' directly from the details view onto the connections of the Move instructions.
 (→ Tag_table_G120 → PZD_IN_G120_01 → PZD_OUT_G120_01)



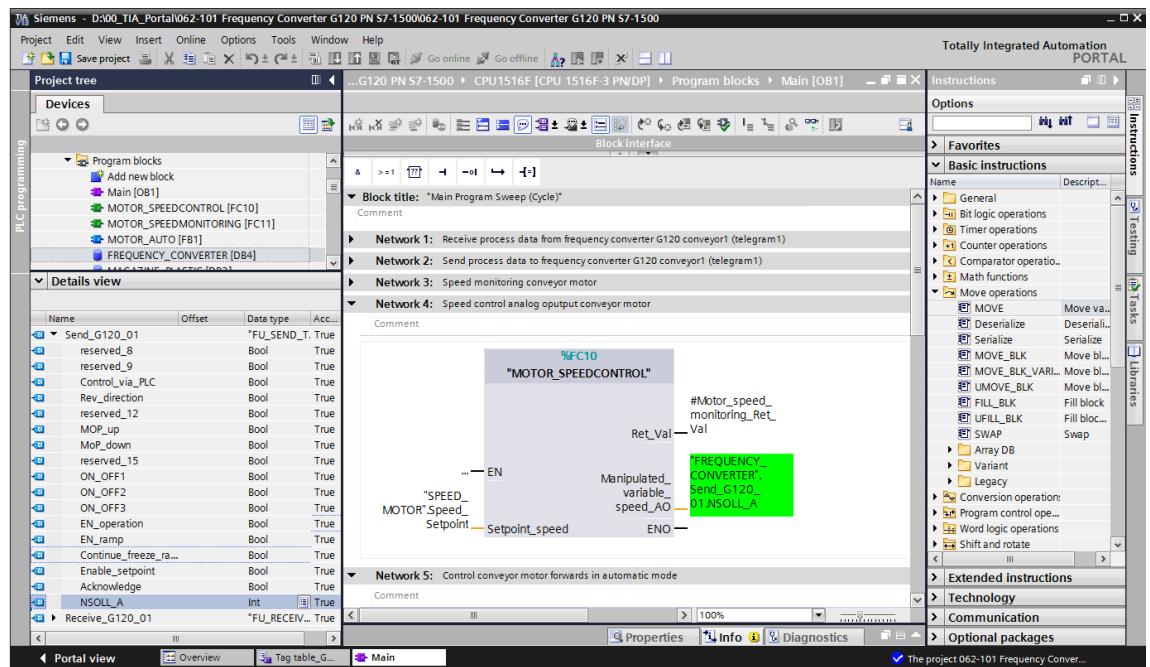
- Select the "FREQUENCY_CONVERTER [DB4]" data block in the project tree. You can again drag-&-drop the two structure tags 'Send_G120_01' and 'Receive_G120_01' directly from the details view onto the connections of the Move instructions.
 (→ Send_IN_G120_01 → Receive_OUT_G120_01)



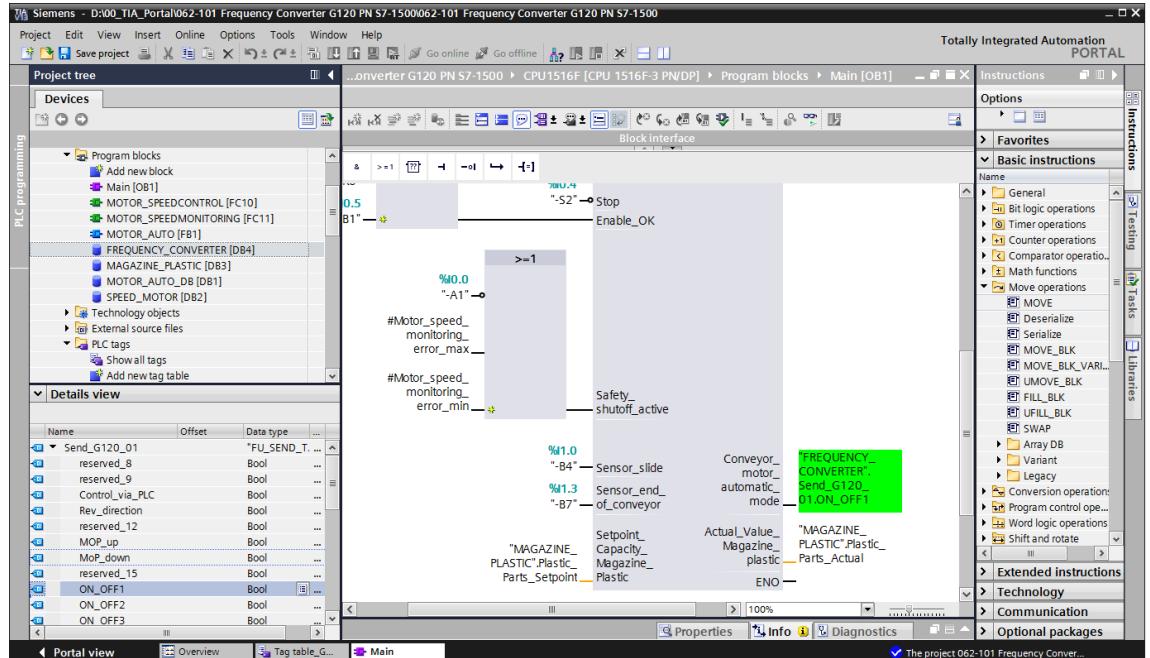
- Open the structure of the tag 'Receive_G120_01' in the details view and from there drag-&-drop the tag 'Receive_G120_01.XIST_A' to the connection 'Actual_speed_AI' of the block 'MOTOR_SPEED_MONITORING'. (→ Receive_G120_01.XIST_A)



- Drag the tag 'Send_G120_01.NSOLL_A' to the connection 'Setpoint_speed' of the block 'MOTOR_SPEEDCONTROL'.
(→ Send_G120_01.NSOLL_A)

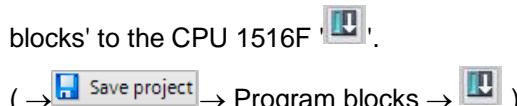


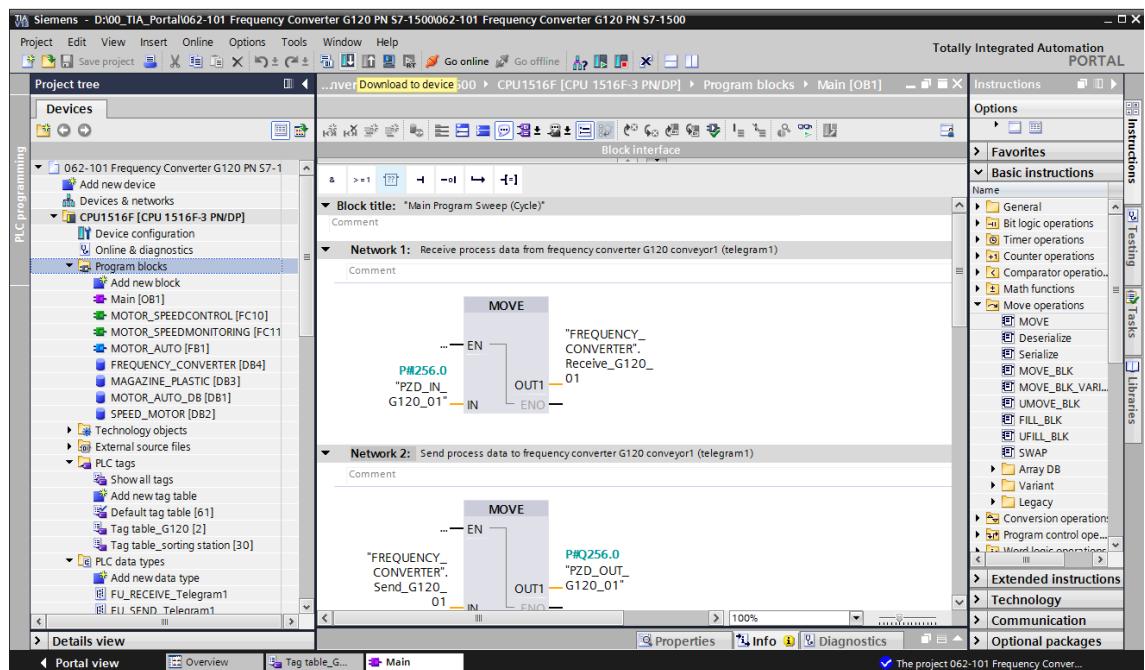
- As the activation command, drag the tag 'Send_G120_01.ON_OFF1' to the connection 'Conveyor_motor_automatic_mode' of the block 'MOTOR_AUTO'.
(→ Send_G120_01.ON_OFF1)



7.6 Loading the program in SIMATIC S7 CPU 1516F-3 PN/DP

- Save the project once more before downloading the modified and created 'Program blocks' to the CPU 1516F [].

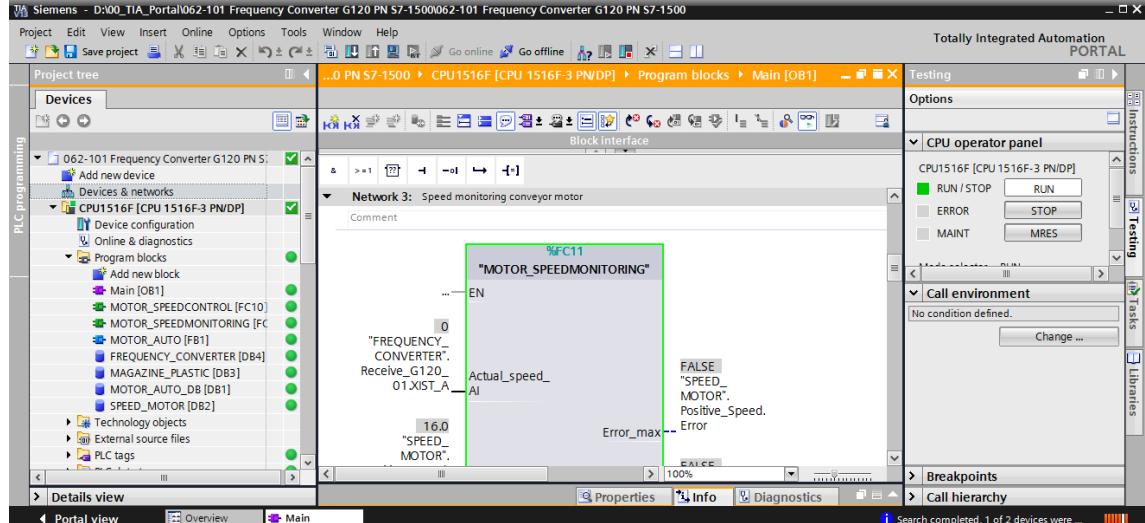




7.7 Diagnostics of SIMATIC S7 CPU 1516F-3 PN/DP

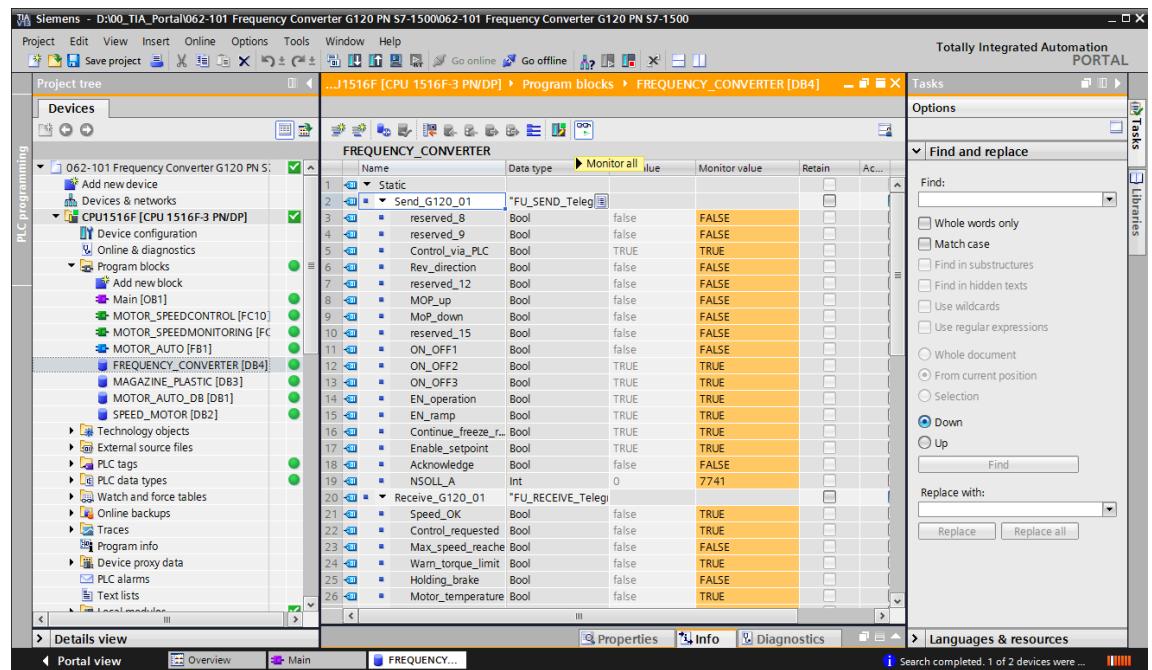
- The block Main [OB1] can be monitored to diagnose the control of the converter from the program. Monitoring is activated and deactivated by clicking the icon.

(→ Main [OB1] →)



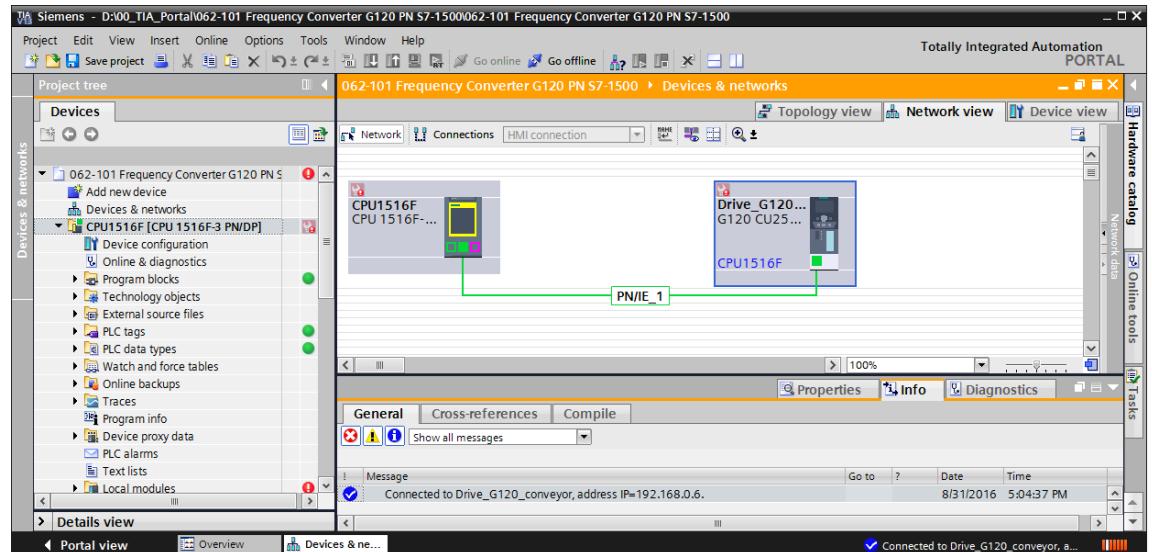
- The complete send and receive data of the communication with the converter (control words/status words/setpoint/actual value) are visible in the 'FREQUENCY_CONVERTER [DB4]' data block. Monitoring can be activated and deactivated at this point as well by clicking the .

(→ FREQUENCY_CONVERTER [DB4] →)



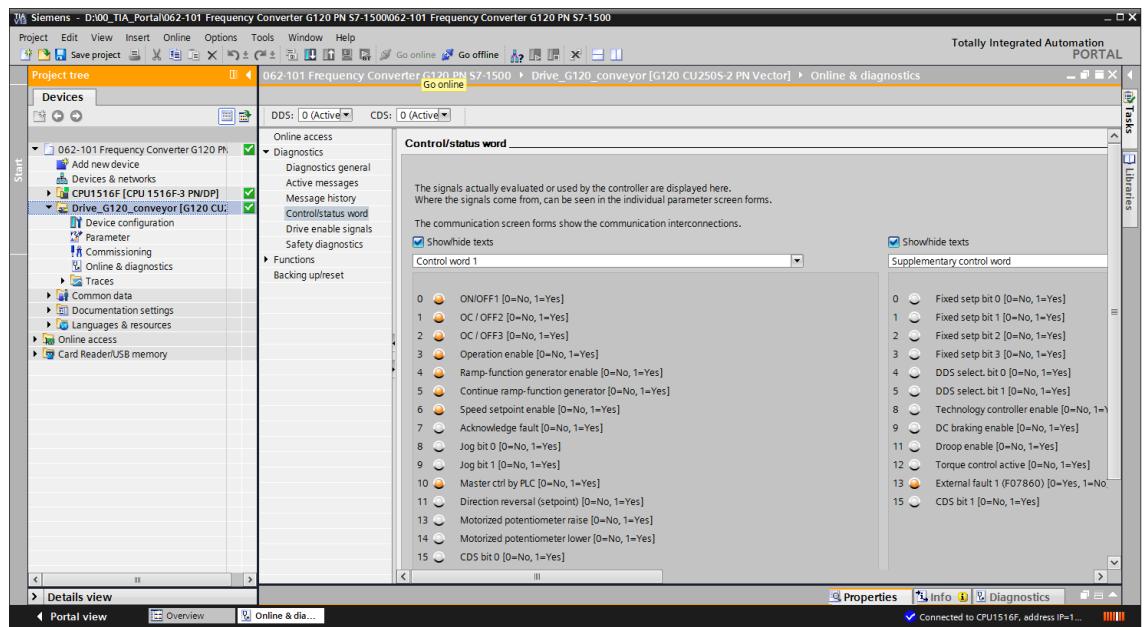
- The online network view lends itself to diagnostics of the PROFINET connection between the CPU 1516F controller and the frequency converter.

(→ Devices & networks → Network view →)



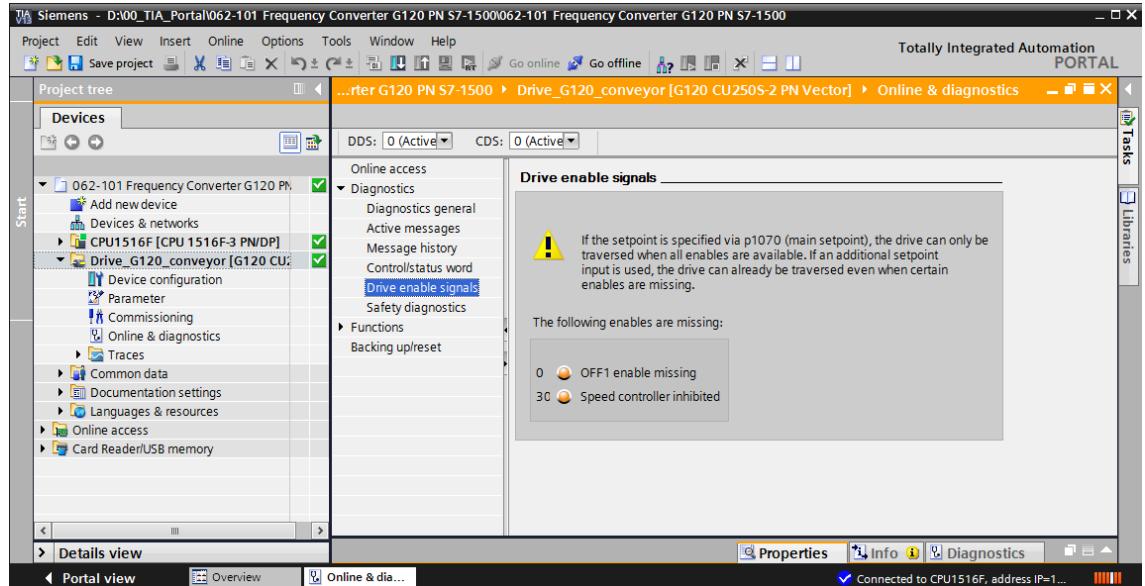
7.8 Diagnostics with SINAMICS Startdrive

- The "Control/status words" can also be monitored in the frequency converter. This is available under 'Online & Diagnostics'
- (→ Drive_G120_conveyor → Online & diagnostics → Diagnostics → Control/status word →)



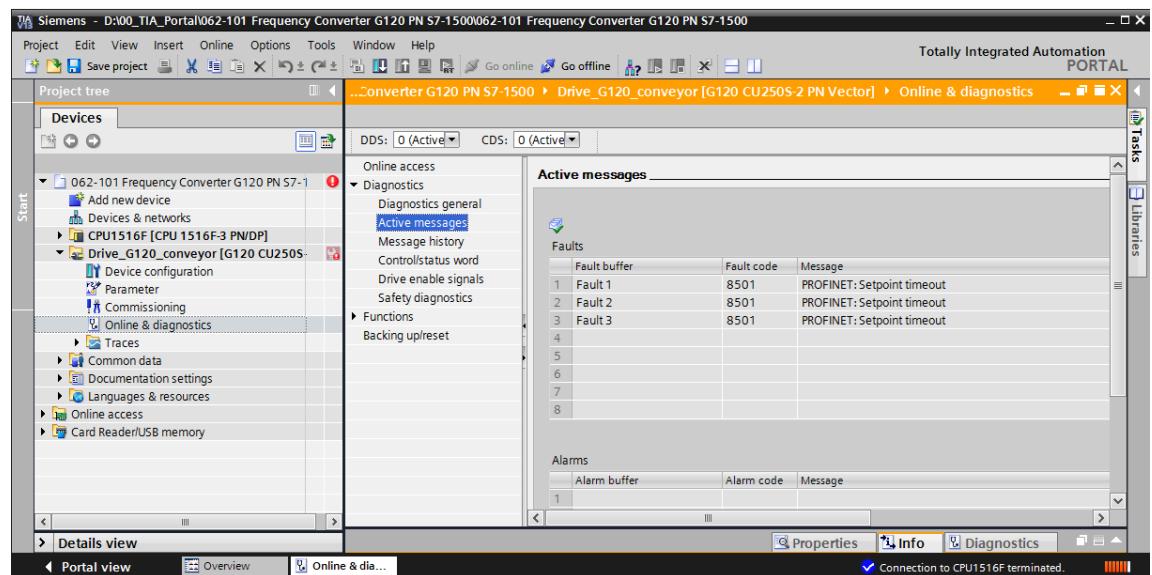
- Under 'Drive enable signals' you also see the missing enables in order to be able to start the motor.

(→ Drive enable signals)

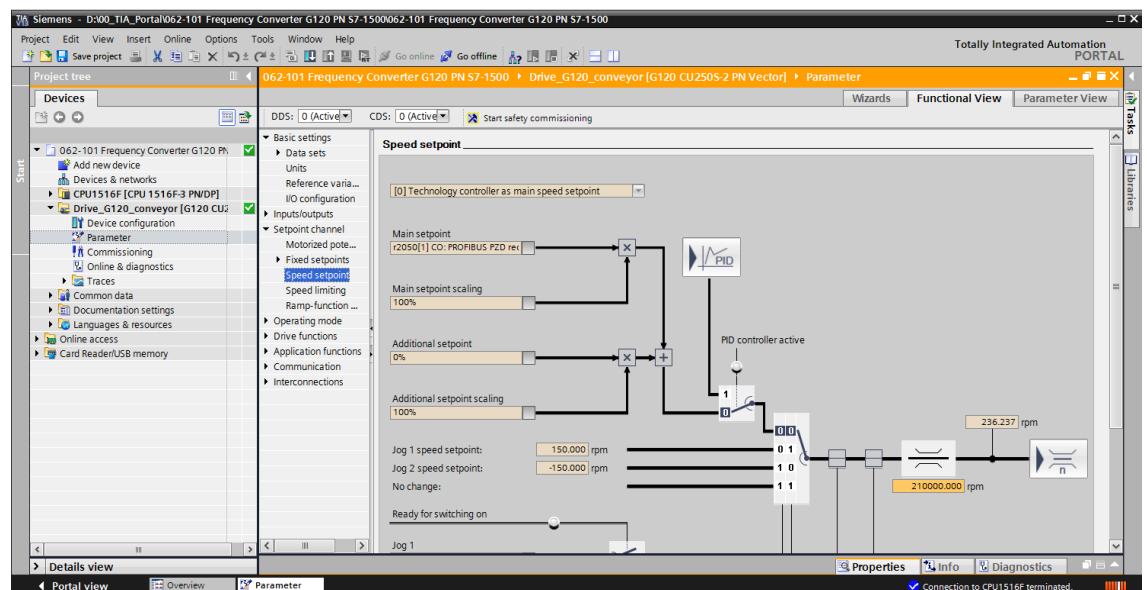


- Under 'Active messages' you see pending faults and warnings. You can click the '!' icon to acknowledge these.

(→ Active messages → !)



- The values can also be monitored online in the 'Functional View' of 'Parameter'.
 (→ Parameter → Functional View)



7.9 Checklist

No.	Description	Checked
1	Frequency converter SINAMICS G120 with Control Unit CU250S-PN Vector created as distributed IO of the CPU1516F- 3 PN/DP.	
2	Device configuration with the frequency converter G120 as device loaded successfully into the CPU1516F-3 PN/DP.	
3	Device name of the Control Unit CU250S-PN Vector assigned.	
4	SINAMICS G120 frequency converter with induction motor parameterized in SINAMICS Startdrive.	
5	Parameter assignment successfully loaded from SINAMICS Startdrive into the SINAMICS G120 frequency converter.	
6	Induction motor tested successfully in operation with SINAMICS G120 frequency converter via control panel.	
7	Data block 'FREQUENCY_CONVERTER' [DB4] created.	
8	Program changes carried out in Main [OB1].	
9	Compiling and downloading of the program blocks is successful and without error message.	
10	Switch on system (-K0 = 1) Cylinder retracted/feedback activated (-B1 = 1) EMERGENCY STOP (-A1 = 1) not activated AUTOMATIC mode (-S0 = 1) Automatic stop pushbutton not actuated (-S2 = 1) Briefly actuate automatic start pushbutton (-S1 = 1) Sensor part at slide activated (-B4 = 1) Then the induction motor is switched on via the frequency converter and remains active → Motor ON	
11	Sensor at conveyor end activated (-B7 = 1) → Motor OFF (after 2 seconds)	
12	Briefly actuate automatic stop pushbutton (-S2 = 0) → Motor OFF	
13	Activate EMERGENCY STOP (-A1 = 0) → Motor OFF	
14	Operating mode manual (-S0 = 0) → Motor OFF	
15	Switch off system (-K0 = 0) → Motor OFF	
16	Cylinder not retracted (-B1 = 0) → Motor OFF	
17	Project archived successfully.	