SIEMENS

SCE Training Curriculum
For Integrated Automation Solutions
Totally Integrated Automation (TIA)

Siemens Automation Cooperates with Education

TIA Portal Module 070-010

PROFINET with IO Controller CPU 315F-2 PNDP
and IO Device ET 200S

Cooperates with Education

Automation

SIEMENS

Suitable SCE trainer packages for these training curriculums

SIMATIC controllers

- **SIMATIC S7-300 with CPU 314C-2PN/DP**
  Order no.: 6ES7314-6EH04-4AB3
- **SIMATIC S7-300 with CPU 314C-2PN/DP (upgrade)**
  Order no.: 6ES7314-6EH04-4AB4
- **SIMATIC S7-300 with CPU 315F-2PN/DP**
  Order no.: ES7315-2FH14-4AB1
- **SIMATIC ET 200S with CPU IM151-8 F PN/DP**
  Order no.: 6ES7151-8FB00-4AB1
- **SIMATIC ET 200S with IM151-3 F PN**
  Order no.: 6ES7151-3BA20-4AB1

SIMATIC STEP 7 Software for Training

- **SIMATIC STEP 7 Professional V11 - Single license**
  Order no.: 6ES7822-1CC01-4YA5
- **SIMATIC STEP 7 Professional V11 - Classroom license (up to 12 users)**
  Order no.: 6ES7822-1AA01-4YA5
- **SIMATIC STEP 7 Professional V11 - Upgrade license (up to 12 users)**
  Order no.: 6ES7822-1AA01-4YE5
- **SIMATIC STEP 7 Professional V11 - Student license (up to 20 users)**
  Order no.: 6ES7822-1AC01-4YA5

Please note that these training packages are replaced with successor packages when necessary.
An overview of the currently available SCE packages is provided under: siemens.com/sce/tp

Advanced Training

For regional Siemens SCE advanced training courses, please get in touch with your regional SCE contact person siemens.com/sce/contact

Additional information regarding SCE

siemens.com/sce

Information regarding usage

This SCE training curriculum for the end-to-end automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational facilities and R&D facilities. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems; i.e., it can be copied entirely or partially and given to those being trained for usage within the scope of their training. Distributing or copying this training curriculum and sharing its contents is permitted within public training and advanced training facilities for training purposes.

Exceptions require written permission from the Siemens AG contact person: Roland Scheuerer roland.scheuerer@siemens.com.

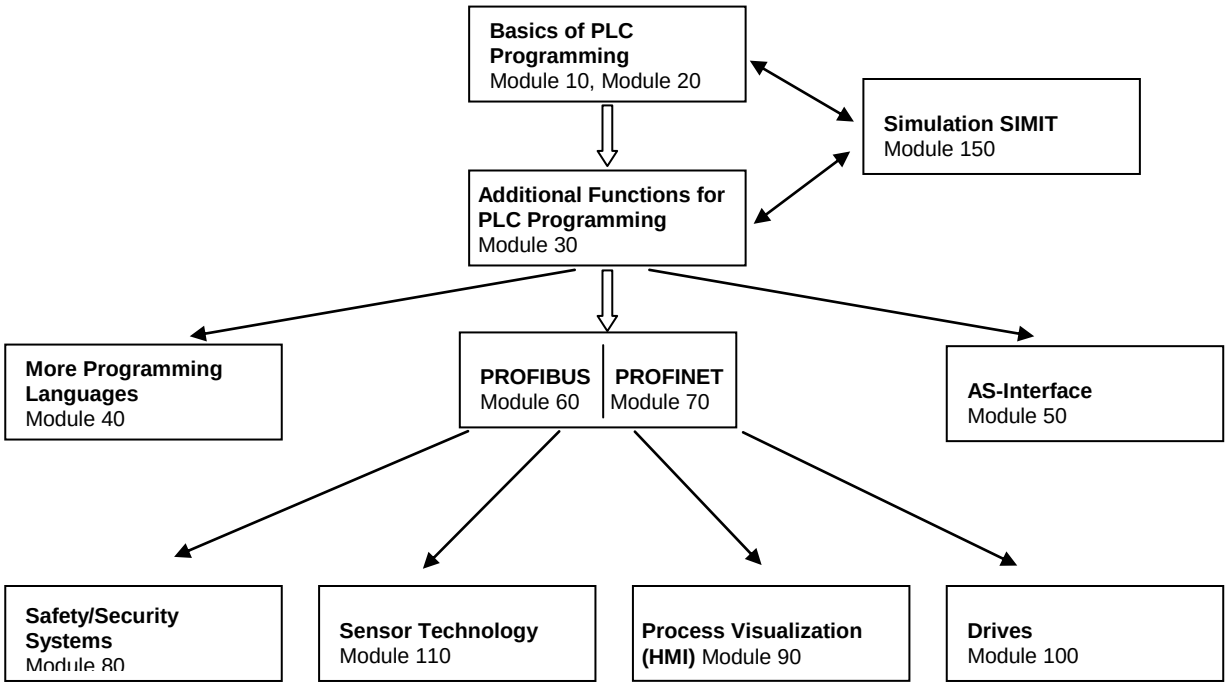
Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.
Usage for industrial customer courses is expressly prohibited. We do not consent to the training curriculum being used commercially.

We wish to thank the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this training curriculum.

PAGE:

1. PREFACE

Regarding its content, module SCE_EN_070-010 is part of the training unit **'PROFINET'**.



Training Objective:

This module provides the reader with information on how to commission the CPU 315F-2PN/DP as IO controller and the ET 200S as IO device on PROFINET. The module demonstrates the general procedure based on a brief example.

Requirements:

To successfully work through this module, the following knowledge is assumed:

- Proficiency in working with Windows.
- Advanced knowledge of PLC programming with the TIA Portal (e.g. module SCE_EN_020-010_R1201_Startup Programming with SIMATIC S7-300 and module SCE_EN_030...).

Hardware and software needed

1 PC Pentium 4, 1.7 GHz 1 (XP) – 2 (Vista) GB RAM, free disk storage approx. 2 GB operating system Windows XP Professional SP3/Windows 7 Professional/Windows 7 Enterprise/Windows 7 Ultimate/Windows 2003 Server R2/Windows Server 2008 Premium SP1, Business SP1, Ultimate SP1

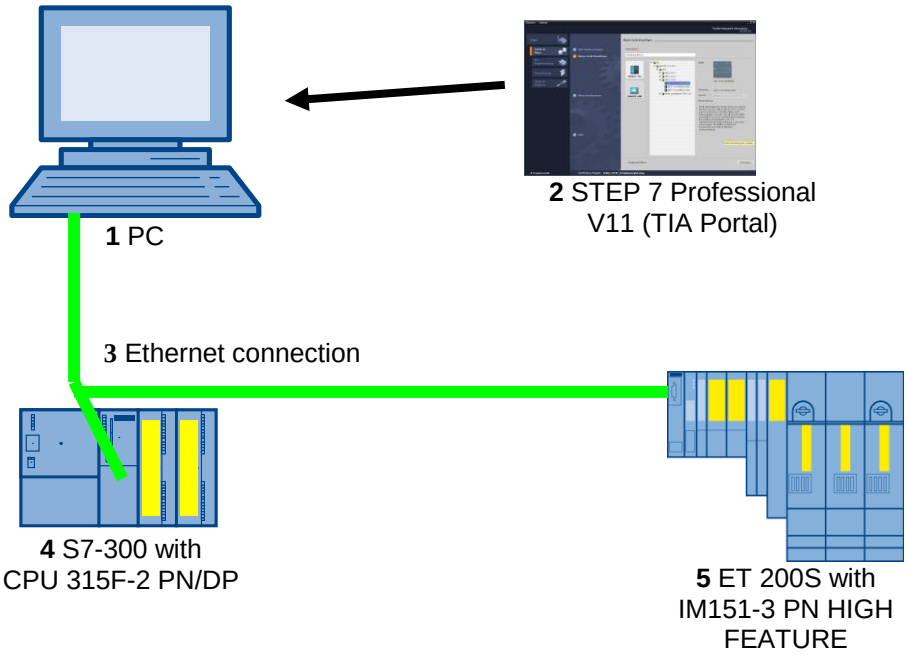
2 Software: STEP 7 Professional V11 SP1 (Totally Integrated Automation (TIA) Portal V11)

3 Ethernet connection between the PC, CPU 315F-2 PN/DP and ET 200S

4 PLC SIMATIC S7-300; for example, CPU 315F-2PN/DP with 16DI/16DO signal module. The inputs have to be brought out to a panel.

5 Distributed IO ET 200S for PROFINET with 8 digital inputs and 8 digital outputs.

- Interface module: IM 151-3 PN HIGH FEATURE
- Power module: PM-E RO DV24V / 8A
- Electronic module 4DI HF DC24V (the inputs must be brought out to a panel)
- Electronic module 4DI HF DC24V (the inputs must be brought out to a panel)
- Power module: PM-E RO DV24V / 8A
- Electronic module 4DO HF DC24V/0.5A
- Electronic module 4DO HF DC24V/0.5A



2. NOTES ON USING THE CPU 315F-2 PN/DP

The CPU 315F-2 PN/DP is a CPU which is supplied with 2 integrated interfaces.

- The first interface is a combined MPI/PROFIBUS DP interface which can be used as master or slave on PROFIBUS DP to connect distributed IO/field devices with very fast response times. The CPU can be programmed via MPI or via PROFIBUS DP.
- The second interface is an integrated PROFINET interface with 2 ports. This allows you to use the CPU as PROFINET IO controller to operate distributed IO devices on PROFINET. You can also use this interface to program the CPU.
- Failsafe I/O devices can also be used with both interfaces.
- In addition, CPU 315F-2 PN/DP features a web server that allows remote diagnostics to be conducted using any browser, even without STEP 7 software

Notes:

- In this module, the CPU 315F-2 PN/DP is used as IO controller on PROFINET.
- A Micro Memory Card is required in order to operate this CPU.
- Parameters for the addresses of the input and output modules can be assigned for this CPU.

3. NOTES ON USING THE ET 200S WITH IM 151-3 PN HF

SIMATIC ET 200S is a highly modular, distributed I/O device. It can be operated with various interface modules. Here is a selection of these:

IM 151-1 BASIC, IM 151-1 STANDARD and IM 151-1 FO STANDARD for the connection of max. 63 I/O modules (all types, except PROFIsafe) to PROFIBUS DP; alternatively, bus connection with RS 485 D-Sub connector or using integrated fiber-optic connection.

IM 151-1 HIGH-FEATURE for connection of max. 63 IO modules (all types, including isochronous mode for PROFIsafe) to PROFIBUS DP; bus connection with RS485 D-Sub connector.

IM 151-3 PN for connection of max. 63 IO modules (all types, including isochronous mode for PROFIsafe) to PROFINET IO controller; bus connection with RJ45 connector.

IM 151-3 PN HF (HIGH FEATURE) for connection of max. 63 I/O modules (all types, including isochronous mode for PROFIsafe) to PROFINET IO controller; bus connection with 2x RJ45 connectors.

IM151-8 F-CPU PN/DP, IM 151-7/F-CPU, IM 151-7/CPU and IM 151-7/CPU FO for connection of max. 63 I/O modules (all types, PROFIsafe only with IM151-8 F-CPU PN/DP and IM151-7/F-CPU) to PROFINET or PROFIBUS DP (alternatively, bus connection with RS 485 D-Sub connector or using integrated fiber-optic connection). With integrated CPU to preprocess the process data.

The following I/O modules, for example, can be used here:

Power modules to group load and encoder supply voltages separately and to monitor these voltages

Digital electronic modules to connect digital sensors and actuators

Analog electronic modules to connect analog sensors and actuators

Sensor module to connect IQ-Sense sensors

Technology modules Electronic modules with integrated technological functions, such as counting, positioning, data exchange, etc.

Frequency converter and motor starter modules

Notes:

- The interface module IM151-3 PN HF is used as PROFINET IO device in this module.
- A Micro Memory Card is recommended for the operation of the IM151-3 PN HF to allow a device replacement without programming device.

4. COMMISSIONING THE PROFINET (IO CONTROLLER CPU 315F-2 PN/DP / IO DEVICE ET 200S)

Below, a description is provided on how to commission a PROFINET network with the CPU 315F-2 PN/DP as IO controller and the ET 200S as IO device.
To test the configuration, a program is written in which an indicator light P1 is activated when the S1 button is pressed. Another indicator light P2 is activated if two buttons, S1 and S2, are pressed at the same time.

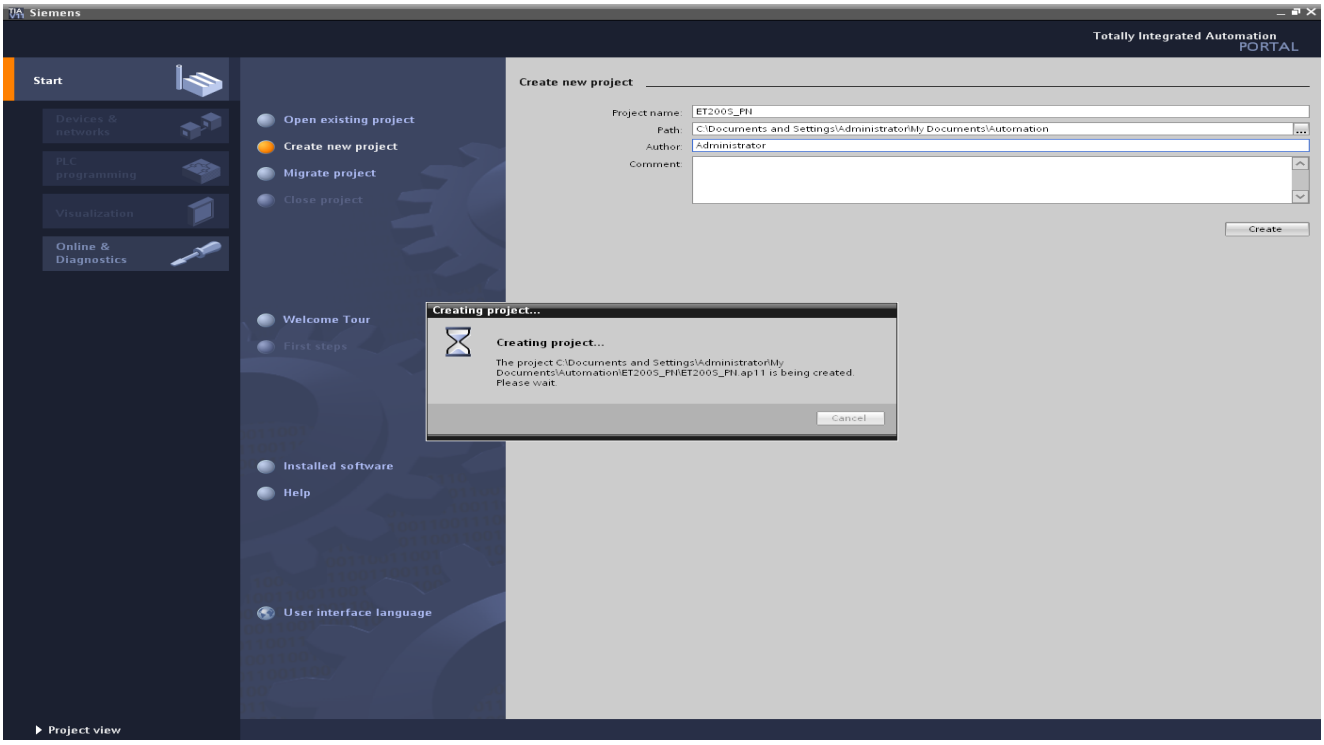
Assignment list:

%I0.0	S1	Selector switch S1 (NO)
%I0.4	S2	Selector switch S2 (NO)
%Q0.0	P1	Indicator light P1
%Q0.4	P2	Indicator light P2

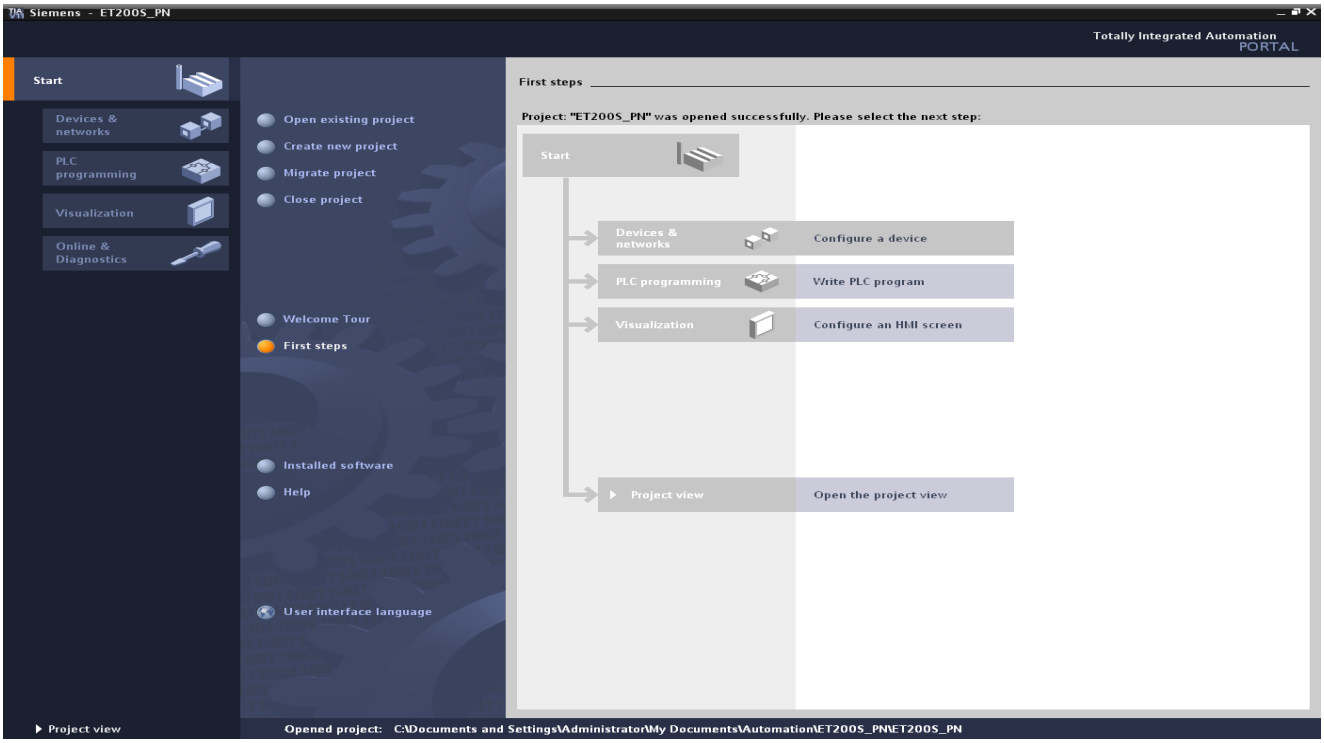
4.1. Creating a project and configuring hardware

1. The central tool is the **'Totally Integrated Automation Portal'**, which is opened with a double-click. (→ TIA Portal V11)

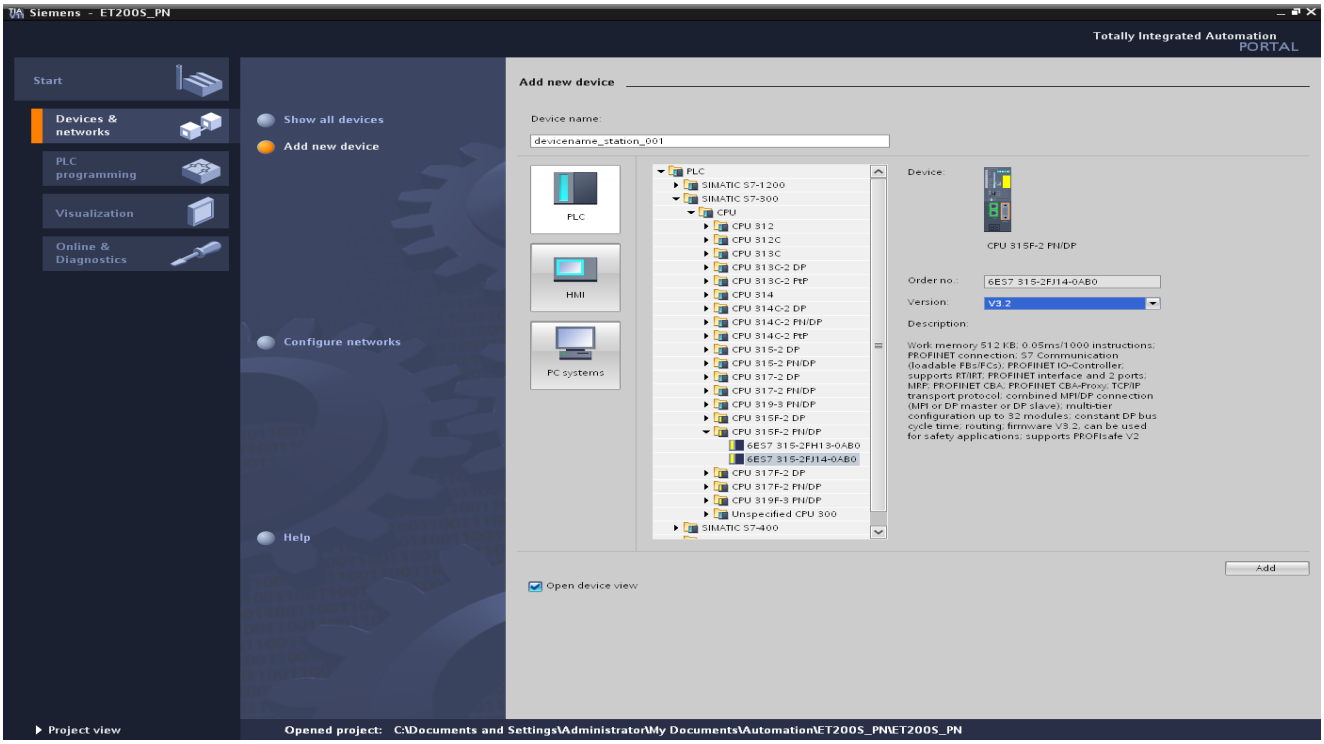
2. Programs for SIMATIC S7-300 are managed in projects. Such a project is now created in the portal view. (→ Create new project → ET 200S_PN → Create)



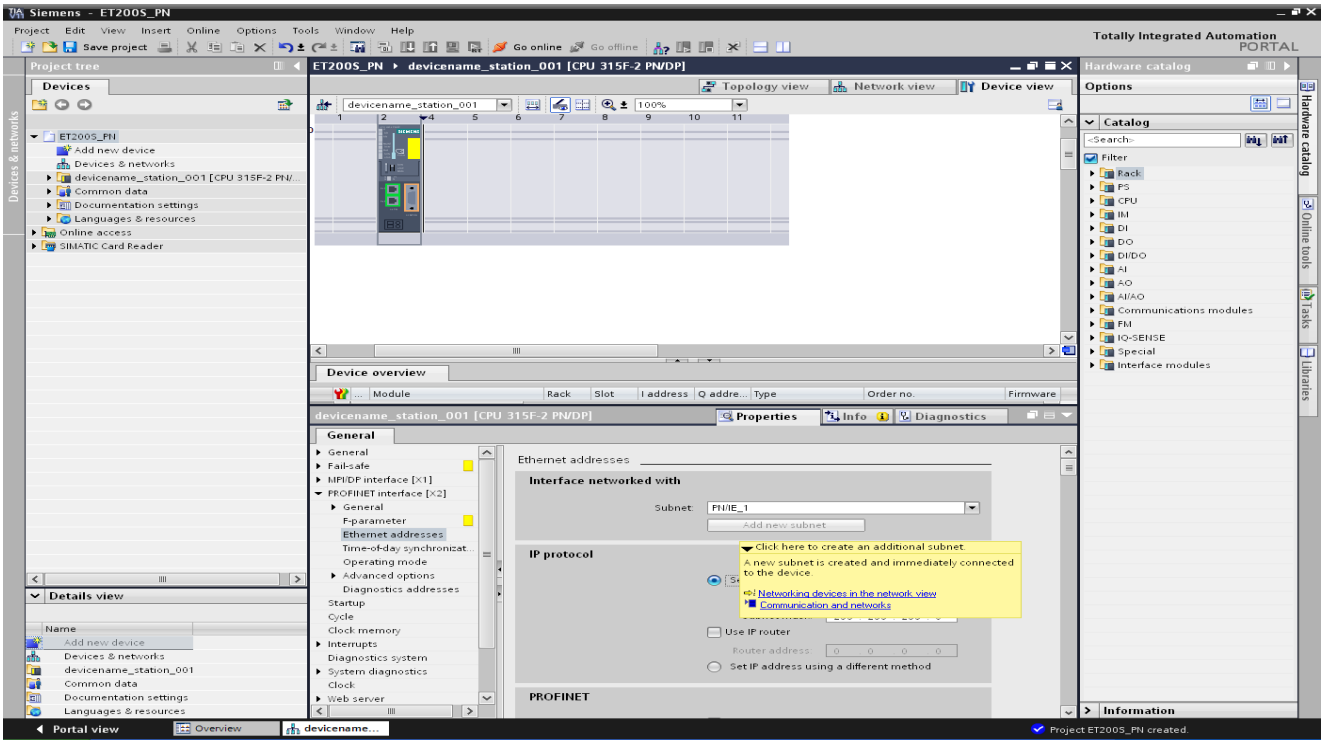
3. **'First steps'** for configuring are now suggested. We first want to **'Configure a device'**. (→ First steps → Configure a device)



4. The next step is **'Add new device'** with the **'Device name control 001'**.
   Choose the **'CPU 315F-2 PN/DP'** with the appropriate order number from the catalog. (→ Add new device → Control 001 → PLC → SIMATIC S7-300 → CPU → CPU 315F-2 PN/DP → 6ES7 315-2FJ14-0AB0 → V3.2 → Add)



5. The software now switches automatically to the project view containing the opened hardware configuration in the device view. Additional modules can be added from the hardware catalog (to the right!).
   To ensure that the software will access the correct CPU later, the IP address and the subnet mask of the CPU must be set. In addition, this interface is also connected with a subnet.
   (→ Properties → General → PROFINET interface → Ethernet addresses → Add new subnet → PN/IE_1 → Set IP address in the project → IP address: 192.168.0.1 → Subnet mask: 255.255.255.0).
   (See also: Module SCE_EN_020-010_R1110_Startup Programming with SIMATIC S7-300 to set the programming interface).



Notes on networking on Ethernet

MAC address:
The MAC address consists of a fixed and a variable part. The fixed part ("base MAC address") identifies the manufacturer (Siemens, 3COM, ...). The variable part of the MAC address differentiates between the various Ethernet devices and should be unique worldwide. A factory-assigned MAC address is imprinted on each module.

Value range for the IP address:
The IP address consists of 4 decimal numbers of the range from 0 to 255, separated by a period; for example, 141.80.0.16.

Value range for the subnet mask:
This mask is used to detect whether a device or its IP address belongs to the local subnet, or can be accessed only by means of a router.
The subnet mask consists of 4 decimal numbers from the value range 0 to 255, separated by a period. For example, 255.255.0.0.

In their binary representation, the 4 decimal numbers of the subnet mask must include a continuous series of "1" values without any gaps from the left and a series of "0" values without any gaps from the right. The "1" values specify the area of the IP address for the network number. The "0" values specify the area of the IP address for the device address.

Example:
Correct values: 255.255.0.0 decimal = 1111 1111.1111 1111.0000 0000.0000 0000 binary
255.255.128.0 decimal = 1111 1111.1111 1111.1000 0000.0000 0000 binary
255.254.0.0 decimal = 1111 1111.1111 1110.0000 0000.0000.0000 binary
Incorrect value: 255.255.1.0 decimal = 1111 1111.1111 1111.0000 0001.0000 0000 binary

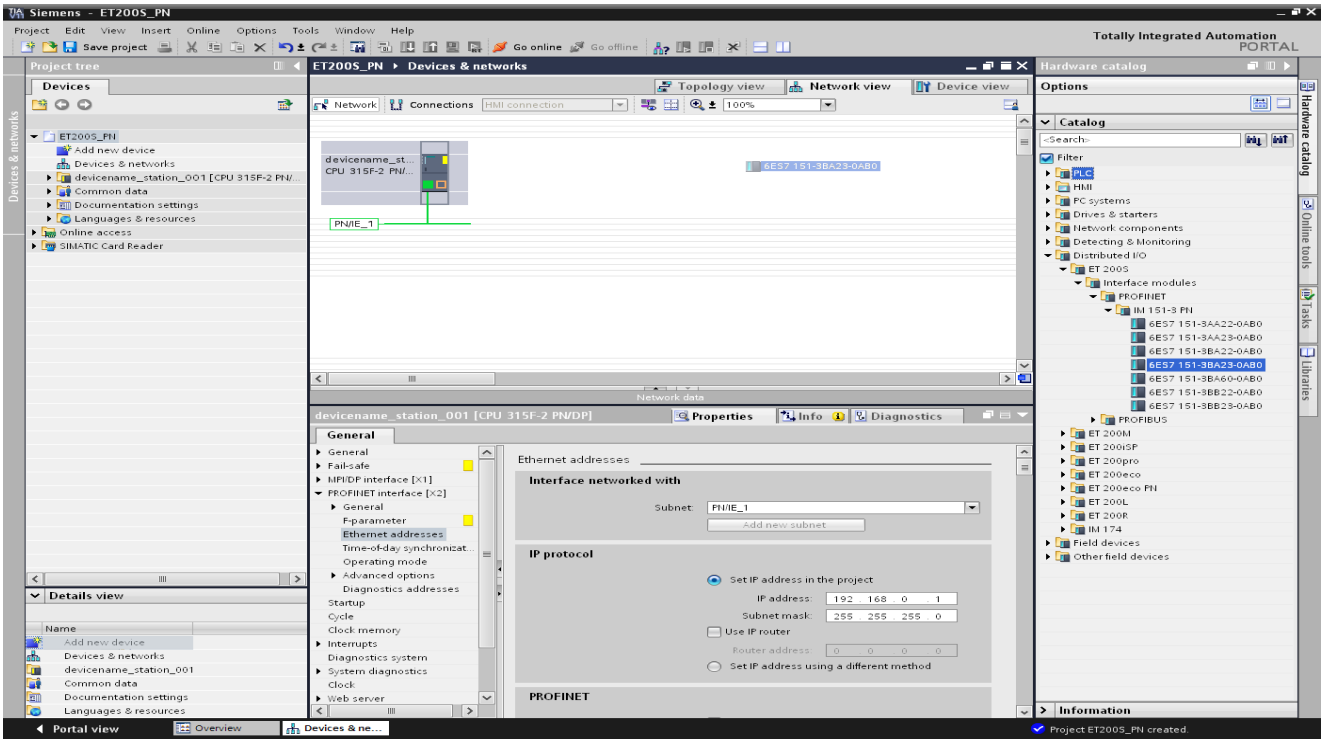
Value range for the address of the gateway (router):
The address consists of 4 decimal numbers from the value range 0 to 255, separated by a period. For example, 141.80.0.1.

Relation of IP addresses, router address, and subnet mask:
The IP address and the address of the gateway may only differ at the positions at which there is a "0" in the subnet mask.

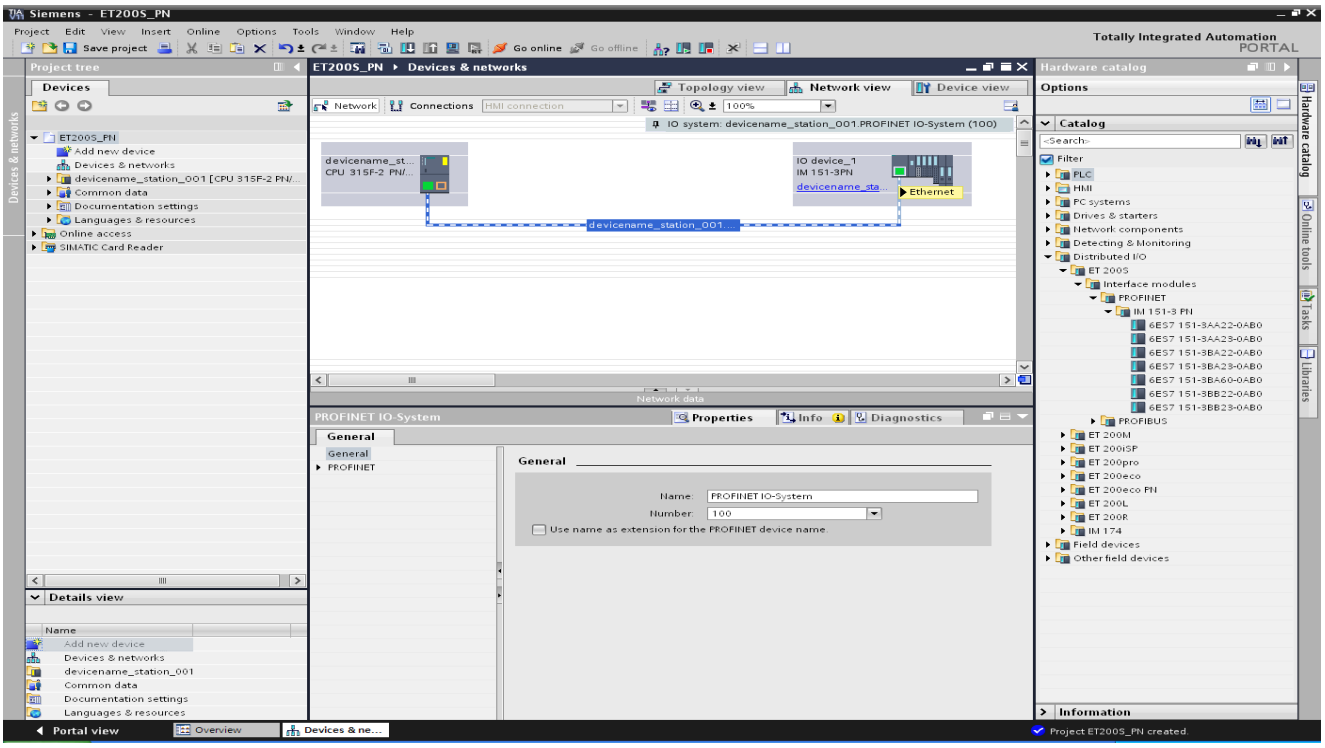
Example: You have entered the following: for the subnet mask 255.255.255.0, for the IP address 141.30.0.5 and for the router address 141.30.128.1.

The IP address and the address of the gateway may only have a different value in the 4th decimal number. In the example, however, the 3rd position is different.
In the example, you will therefore need to make one of the following changes:
- The subnet mask to: 255.255.0.0 or
- The IP address to: 141.30.128.5 or
- The address of the gateway to: 141.30.0.1.

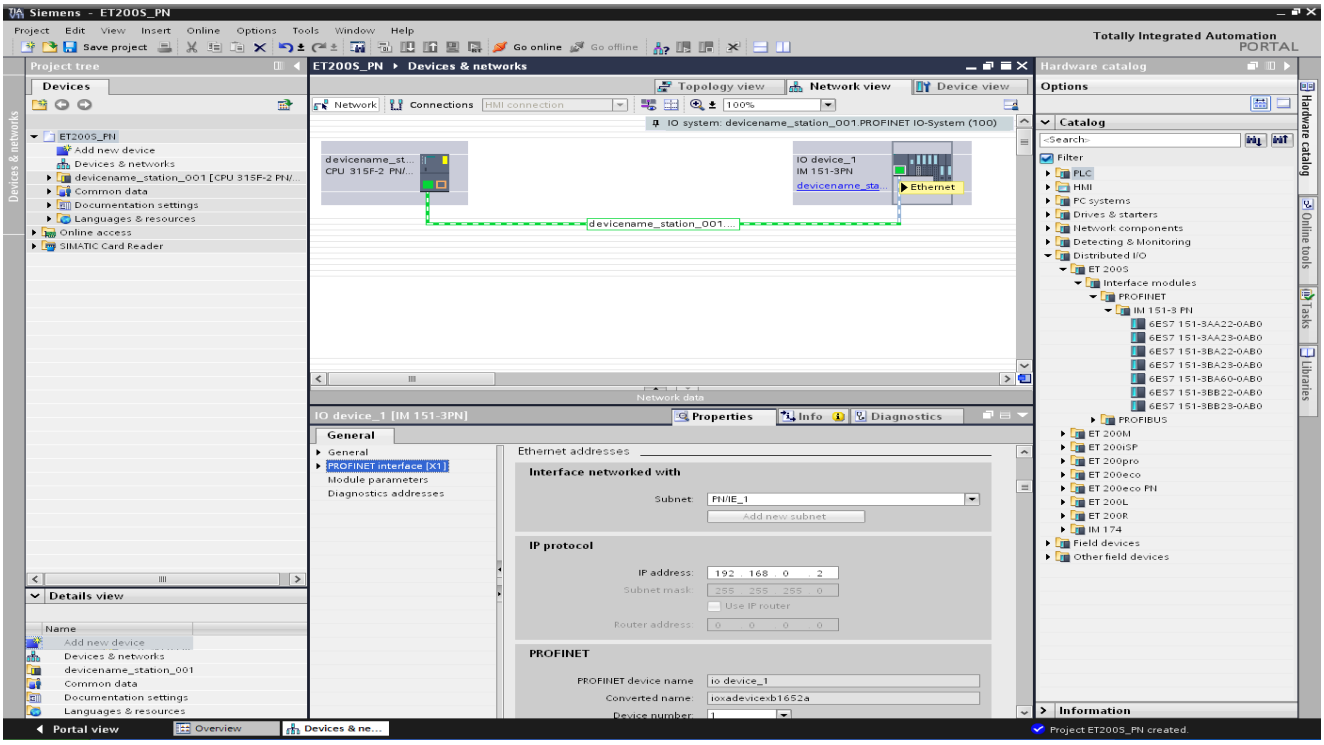
6. To connect the ET 200S with the CPU 315-2 PN/DP, you must go to the **'network view'**. The relevant PROFINET module can be moved to the network view by means of drag-and-drop. (→ Network view → Distributed IO → ET 200S → Interface module → PROFINET → IM151-3 PN → 6ES7 151-3BA23-0AB0)



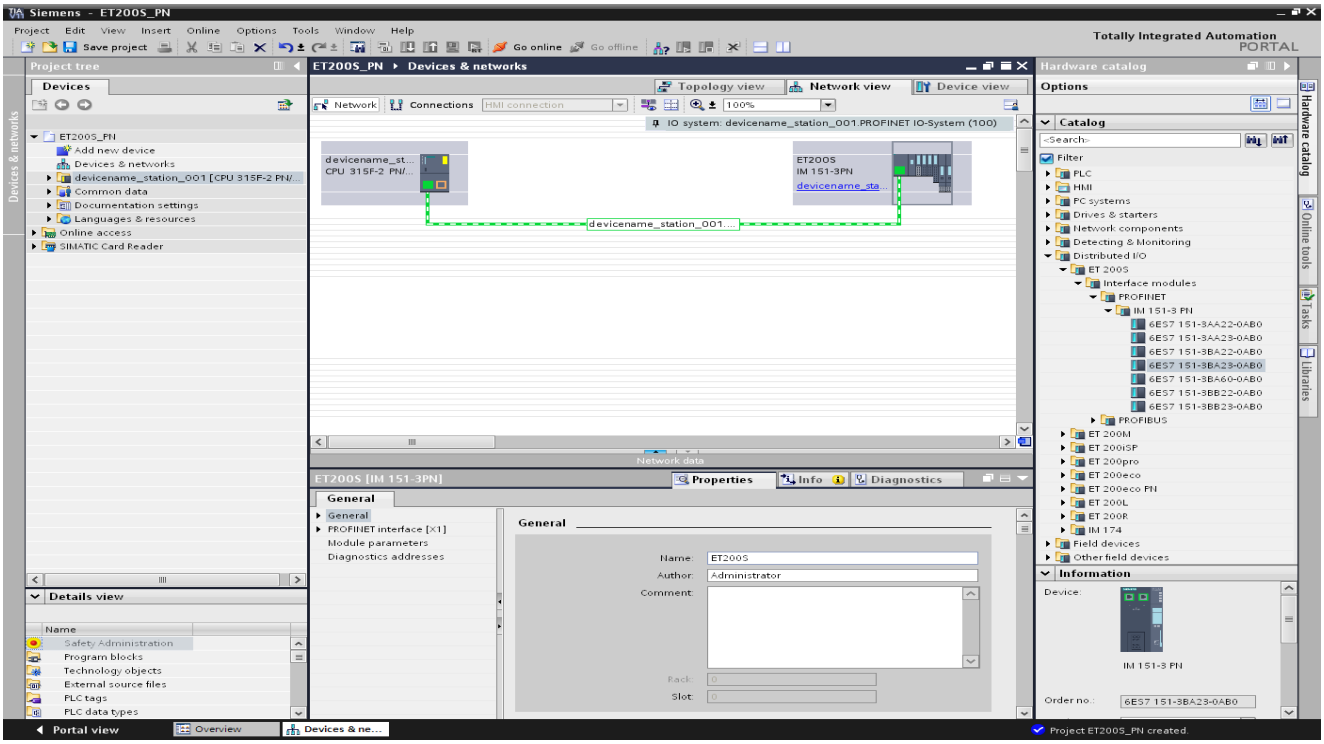
7. You then connect the Ethernet interfaces of the CPU 315-2 PN/DP and those of the IM 151-3PN with the mouse. (→  Ethernet →  Ethernet)



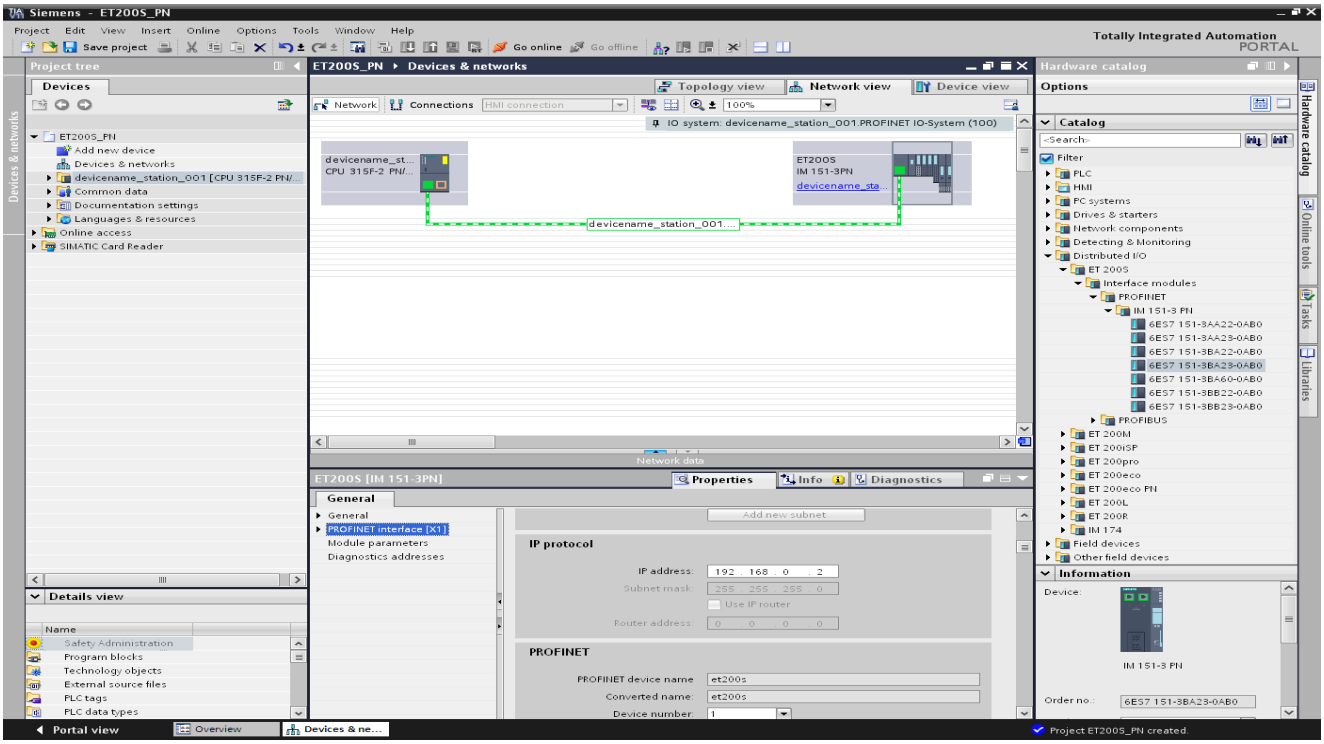
8. Next, set the properties of the **'IM 151-3 PN'** and its **'PROFINET interface'** and assign here an appropriate IP address for the CPU. (→ IM 151-3 PN → Properties → PROFINET interface [X1] → IP protocol → IP address: 192.168.0.2)



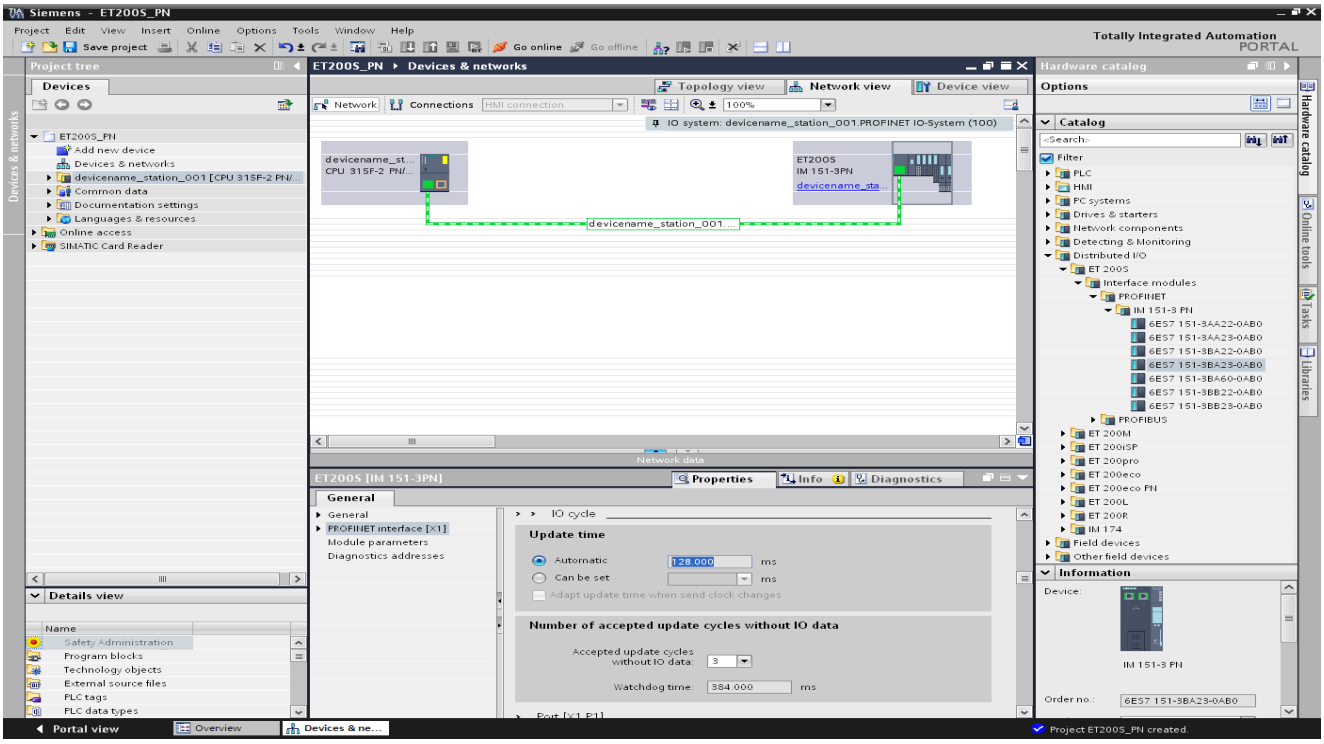
9. After this, the device name has to be assigned under **'General'**. (→ General → Name: ET 200S)



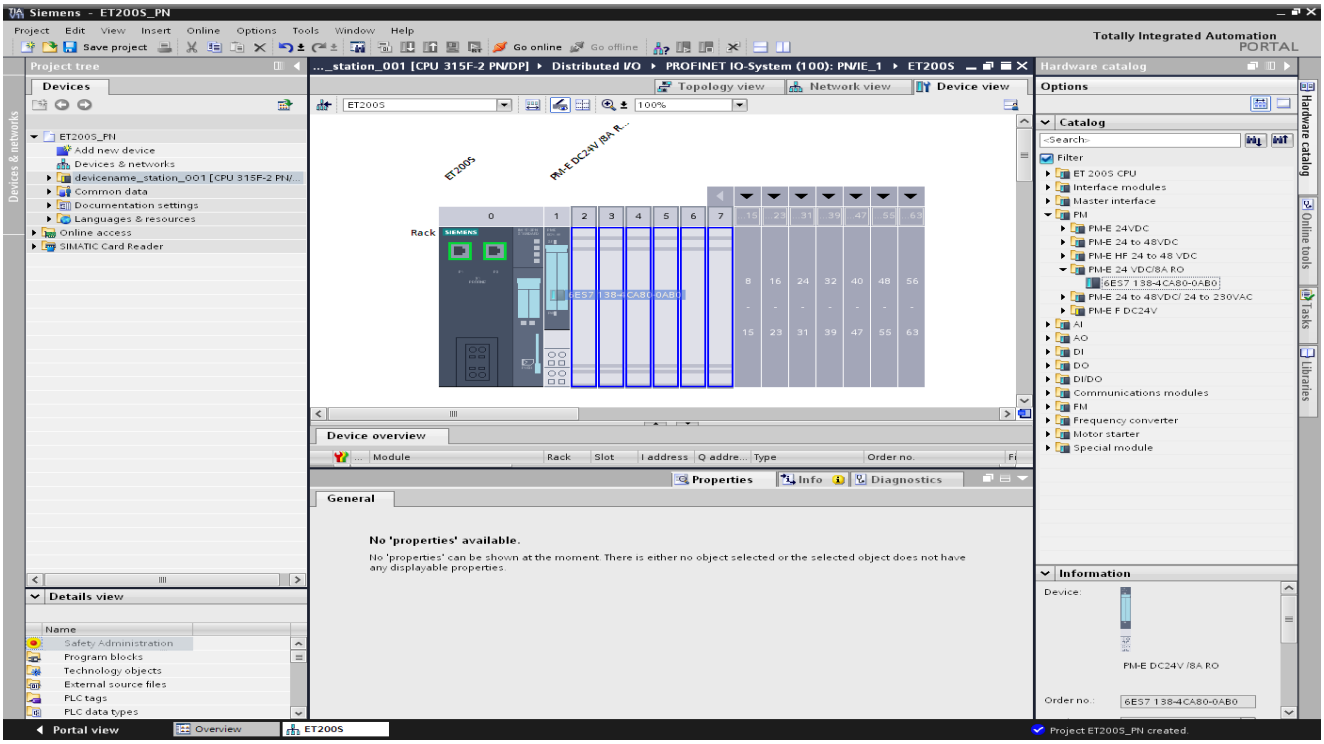
10. With the **'PROFINET interface'** of the **'IM 151-3PN'**, this name is automatically applied as PROFINET device name under the item **'PROFINET'**. (PROFINET interface[X1] → PROFINET)



11. The settings for the **'IO cycle'** such as **'Update time'** and **'Watchdog time'** for each device can be set here. (→ Update time → Watchdog time)



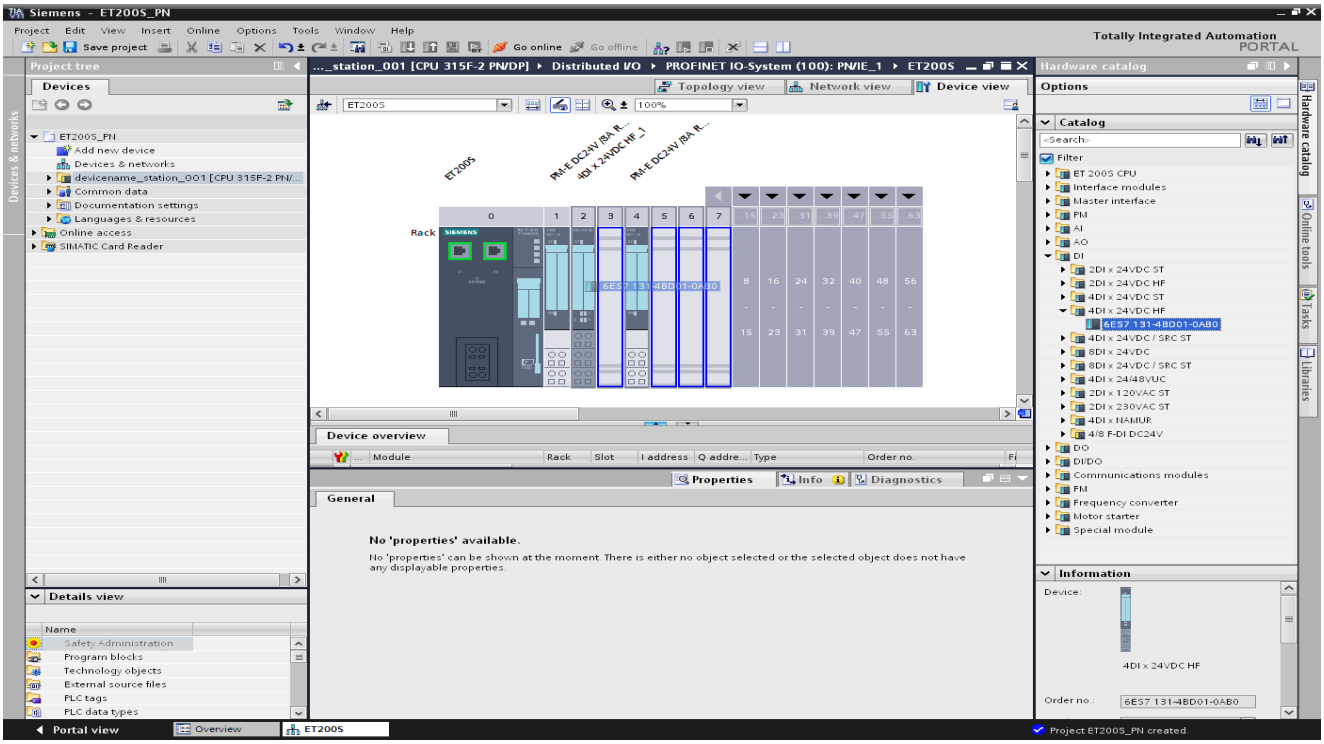
12. In the device view, all other modules which are also inserted in the real ET 200S can now be selected from the hardware catalog and inserted into the configuration table. To do this, click on the name of the corresponding module, keep the mouse button pressed, and drag it to the appropriate column of the configuration table. We start with the 2 power modules **'PM-E DC24V / 8A RO'** which are dragged to slots 1 and 4**.** (→ PM → PM-E DC24V / 8A RO → 6ES7 138-4CA80-0AB0)



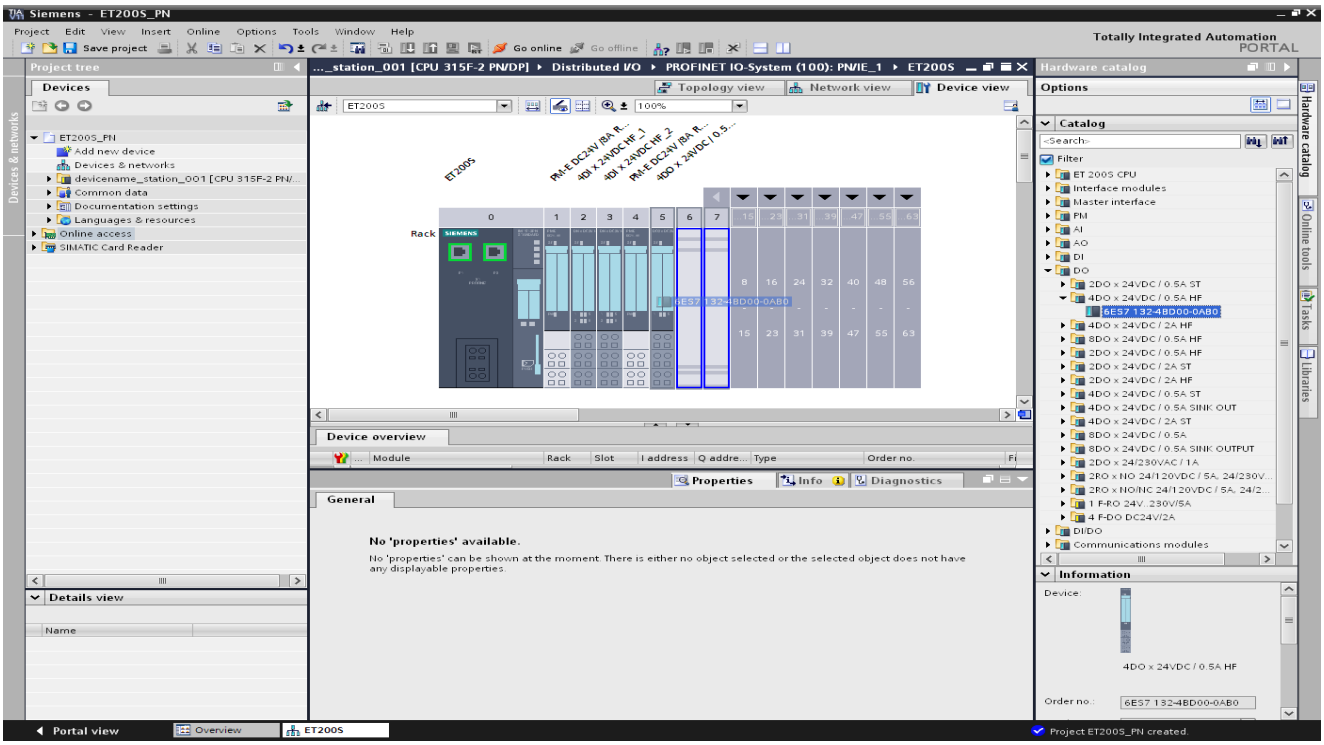
Notes:

When several power modules are used, we have the option of switching the power supply to specific areas on and off. In the event of an emergency stop, for example, the power supply to the output modules can be switched off separately from the power supply to the input modules.

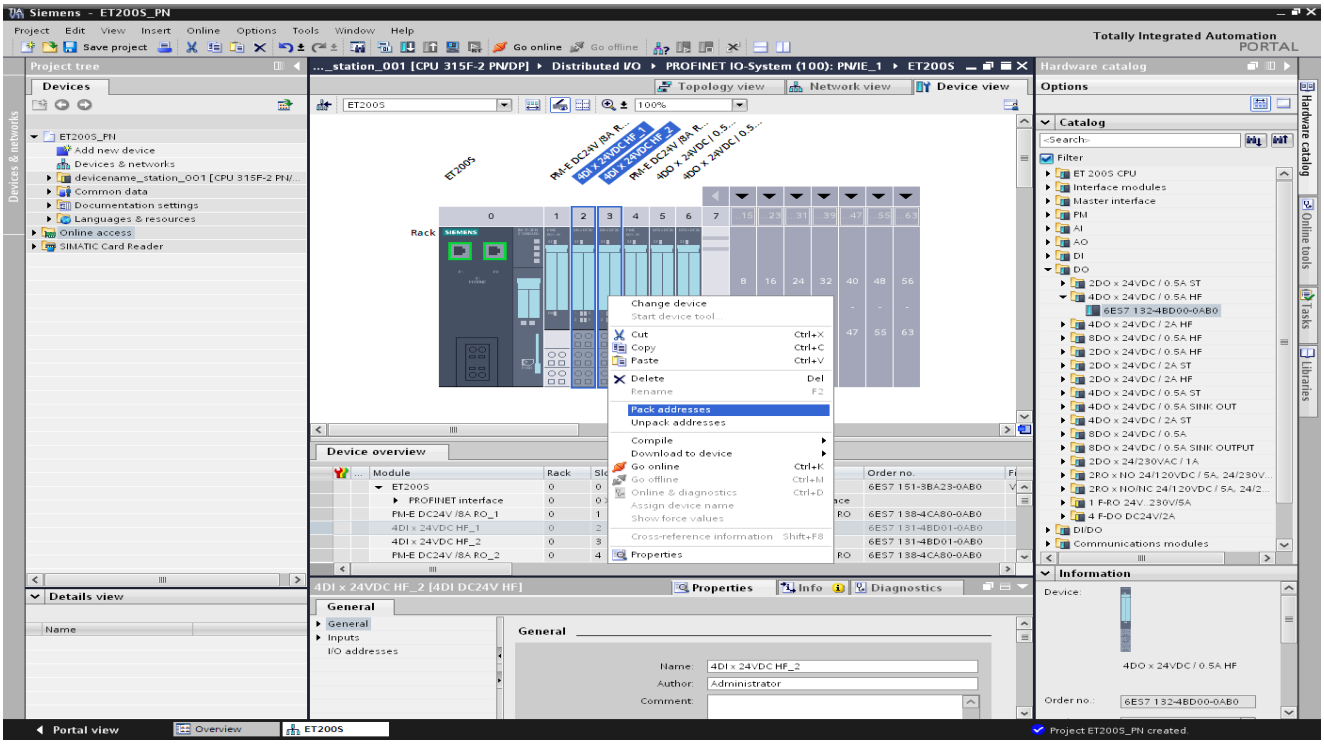
13. We drag the input module **'4DI x DC24V HF'** to slots 2 and 3.
(→ DI → 4DI x DC24V HF → 6ES7 131-4BD01-0AB0)



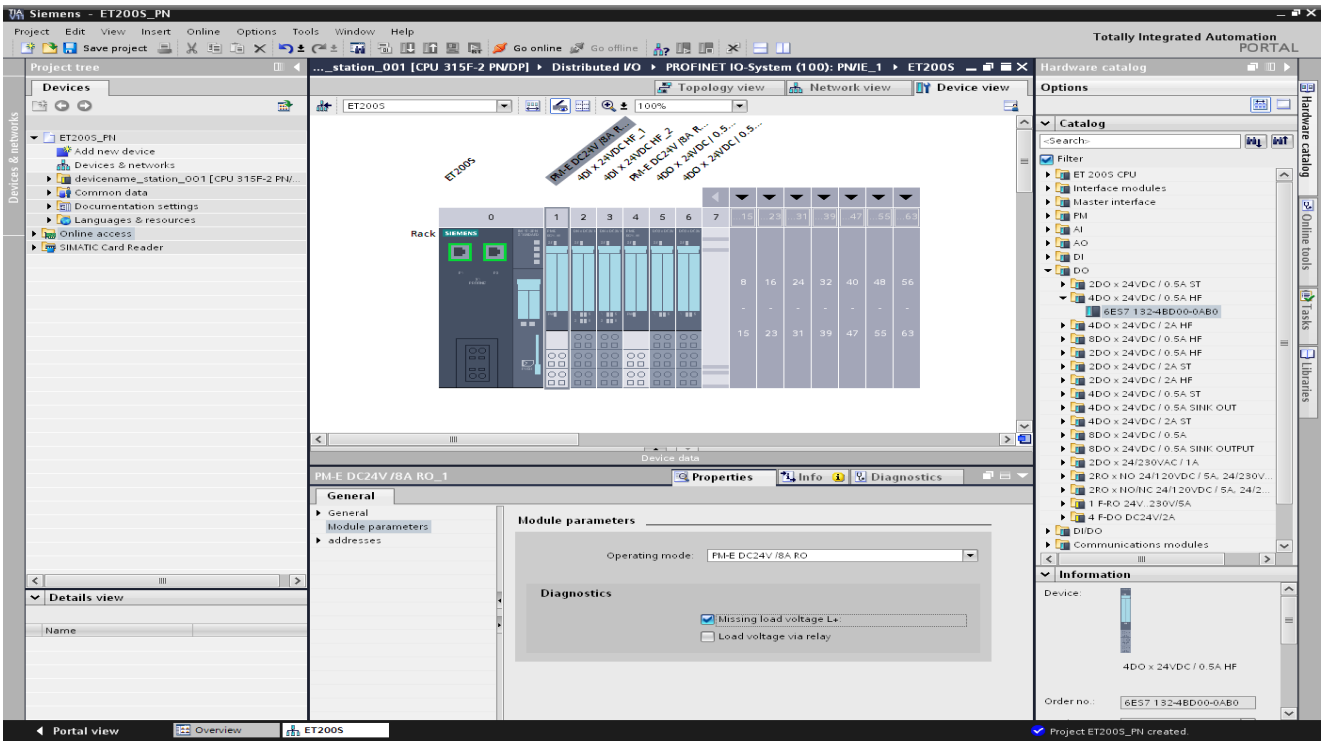
14. We drag the output module **'4DO x DC24V / 0.5A HF'** to slots 5 and 6. (→ DO → 4DO x DC24V / 0.5A HF → 6ES7 132-4BD00-0AB0)



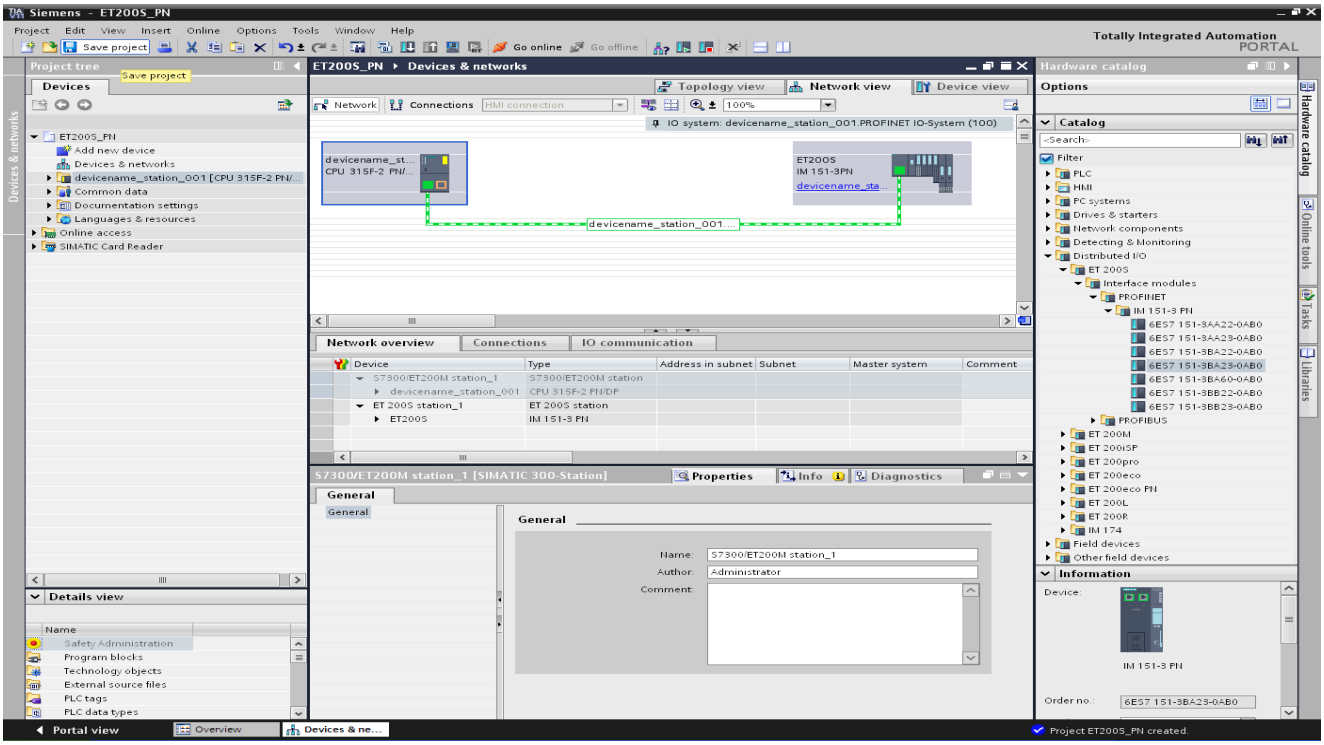
15. Addresses can be packed so that a new byte address does not need to be started with every module. To do this, select the modules in question and right-click on **'Pack addresses'**. (→ Pack addresses)



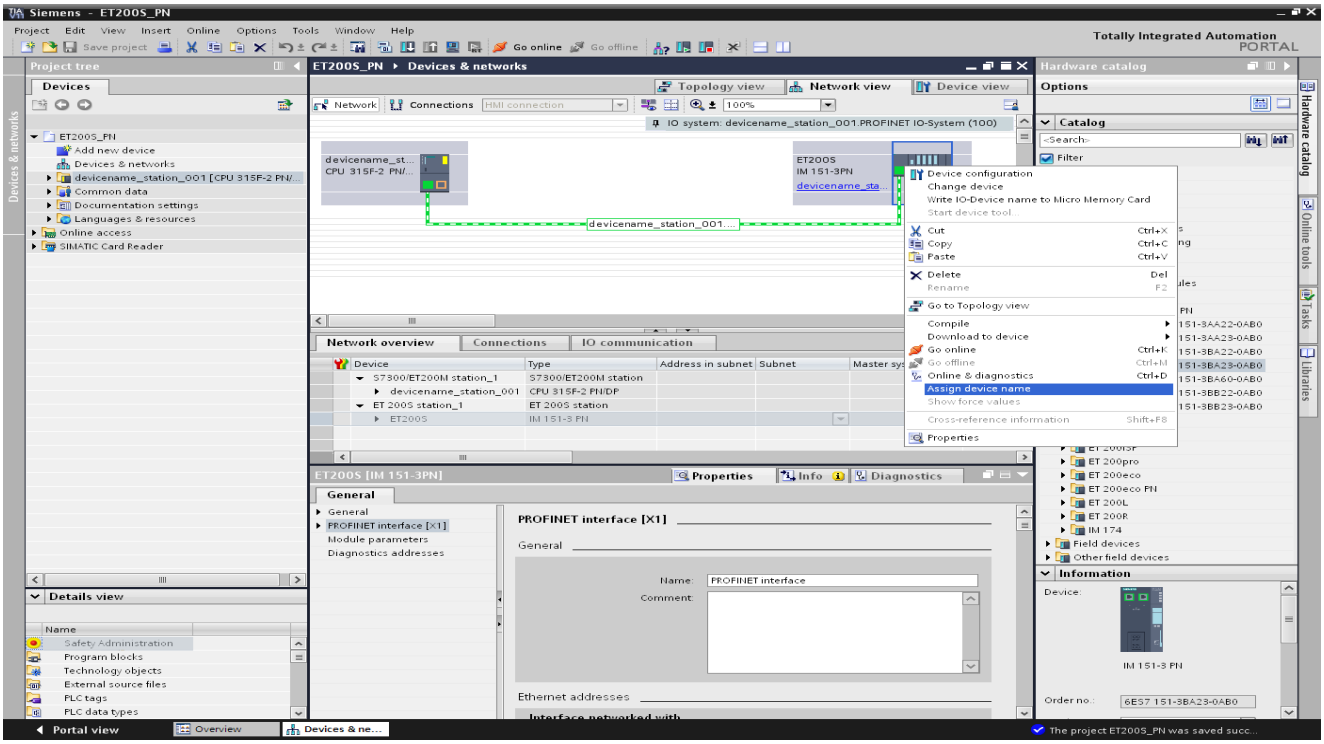
16. **‘Properties’** for the modules can also be set in the **‘Module parameters’**. E.g. **‘Diagnostics, missing load voltage L+’** for the first power module. (→ PM-E DC24V / 8RA RO_1 → General → Module parameters → Diagnostics → Missing load voltage L+)



17. Now, we save the project with the settings made so far. (→ Save project)



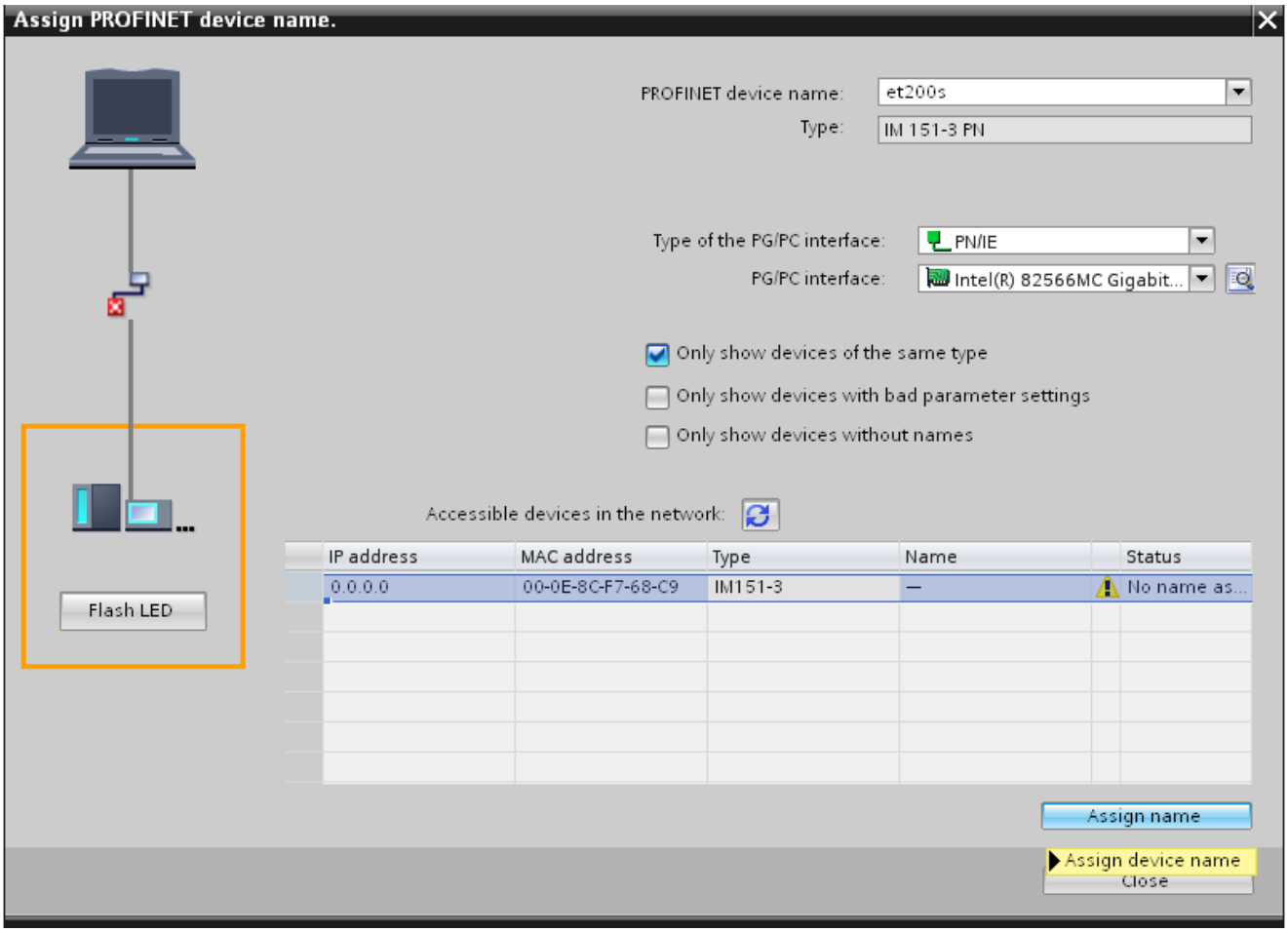
18. After the IO device **'IM151-3PN'** has been selected, the **'Assign device name'** step has to be performed online. (→ IM151-3PN → Assign device name)



Notes:

Before you do this, make sure that the programming device is connected to the IM 151-3PN via Ethernet.

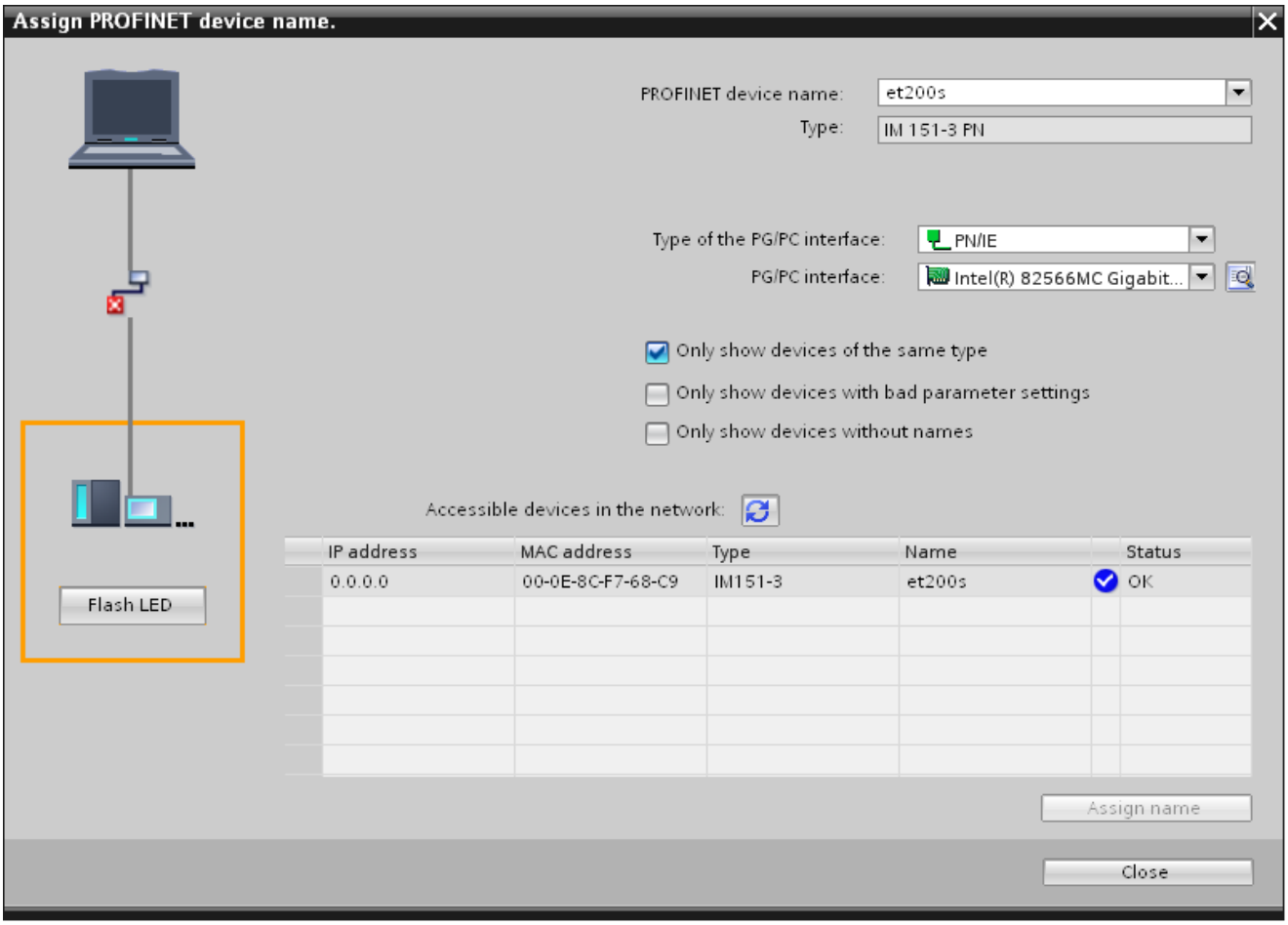
19. The **'Type of the PG/PC interface'** can be selected in the following dialog in order to then select the **'IM151-3PN'** and **'Assign name'**. (→ Assign name)



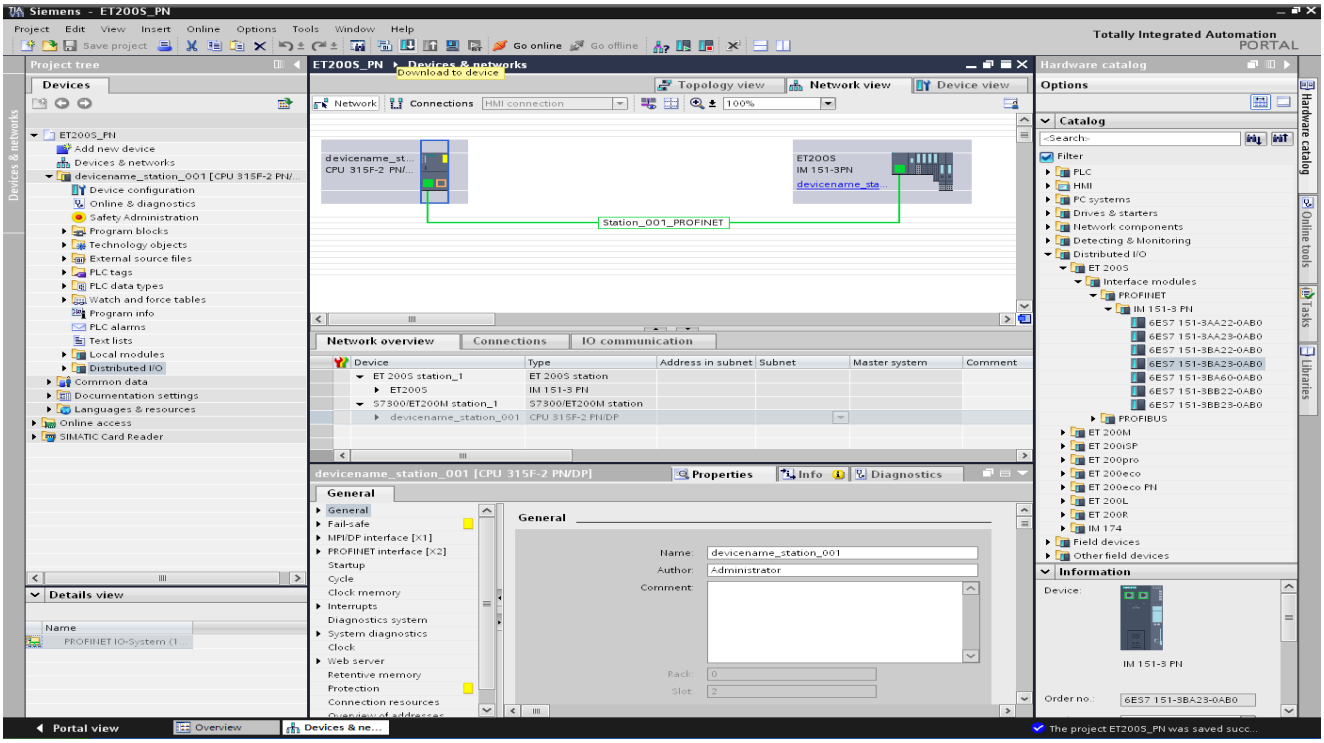
Notes:

If there are several IO devices in the network, the device can be identified on the basis of the imprinted MAC address.

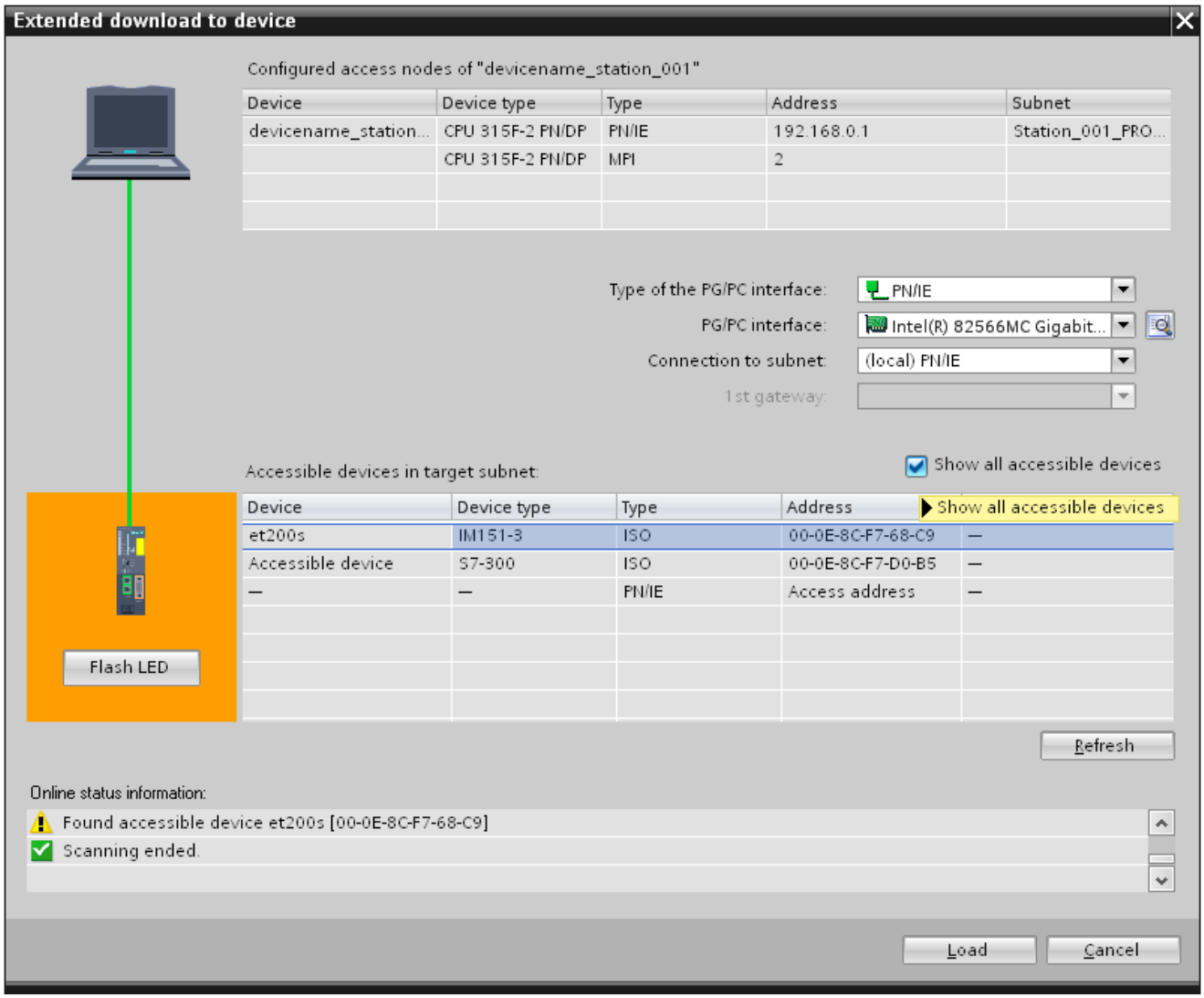
20. If the required module is not displayed, the view can be updated by clicking **'Accessible devices in the network'**  . If the device name was successfully assigned, this is indicated in the status with 'OK'. (→  → Close)



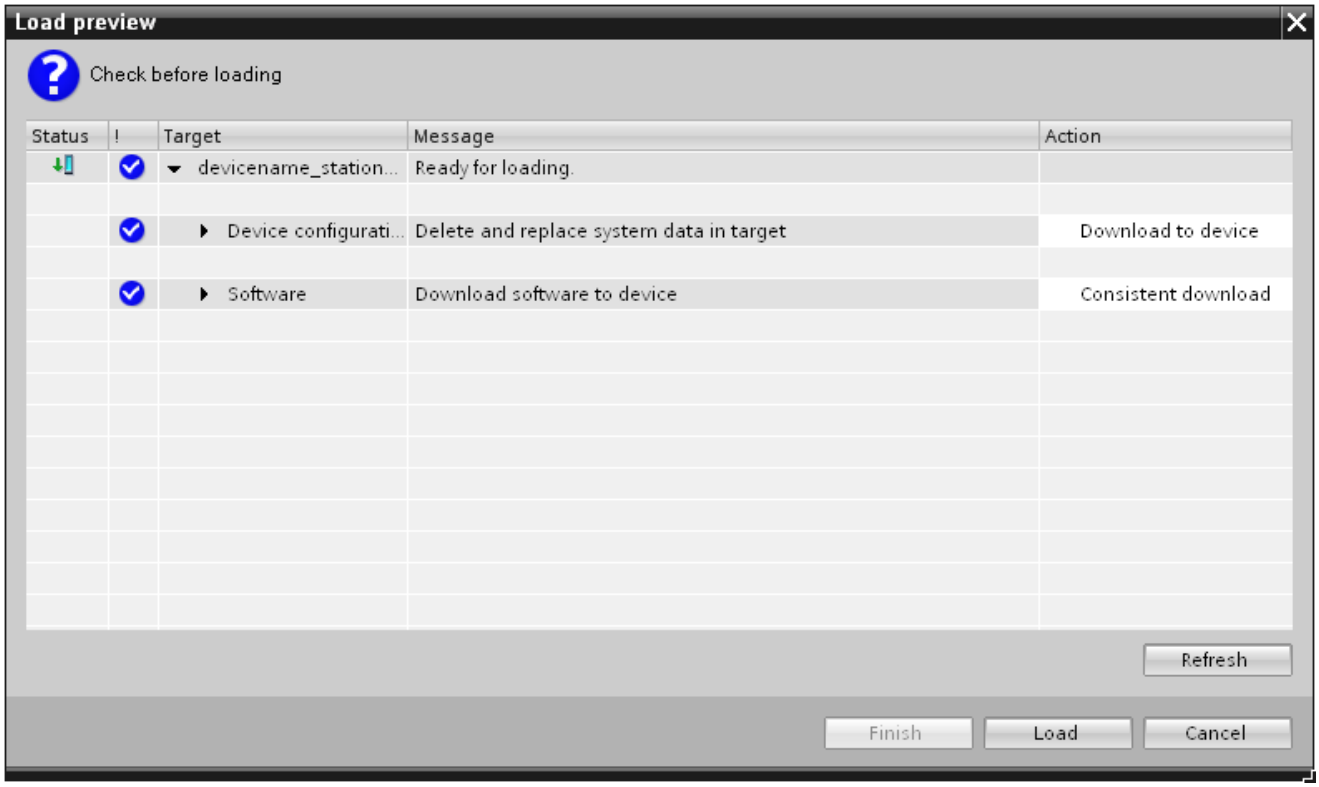
21. To load your entire program to the CPU, first select the **'Control 001'** folder and then click the Download to device icon  . (→ Control 001 → )



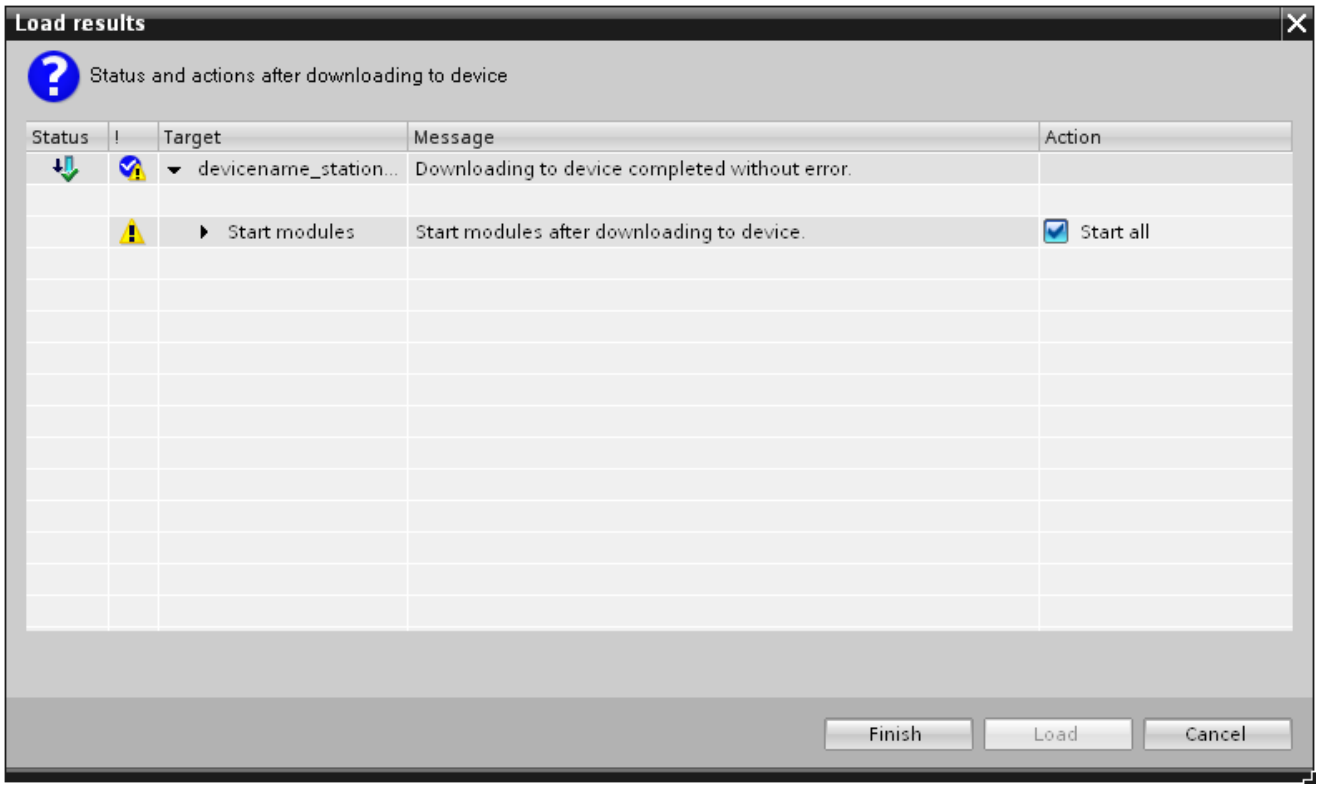
22. In the following dialog, select **'PN/IE'** as the PG/PC interface type and then the network card installed in the PG/PC as PG/PC interface. After you have also activated the **'Show all accessible devices'** option and clicked **'Refresh'**, you should see an **'S7-300'** with the matching MAC address and be able to select it as target device. Next, click **'Load'**. (→ Type of the PG/PC interface: PN/IE → PG/PC interface: …… → Show all accessible devices → Refresh → S7-300 → Load)



23. The configuration is now compiled automatically, and an overview of the steps to be performed is displayed once again for checking before the program is loaded. Click **'Load'** to start loading the program. (→ Load)



24. The successful load result is displayed in a window. Now, click **'Start all'** and then **'Finish'** to set the CPU to Run mode again. (→ Start all → Finish)

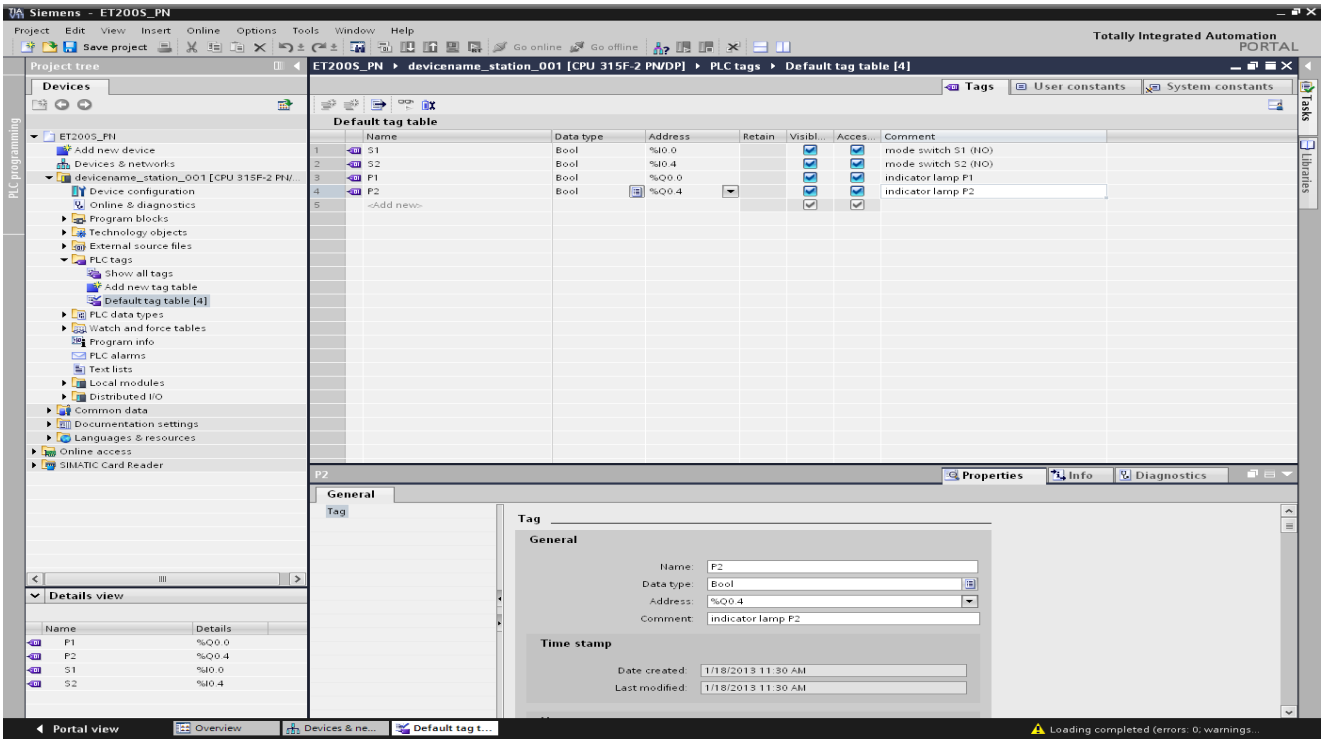


4.2. Generating and testing a program

25. Because modern programming uses tags and not absolute addresses, the **global PLC tags** must be defined here as an initial step.

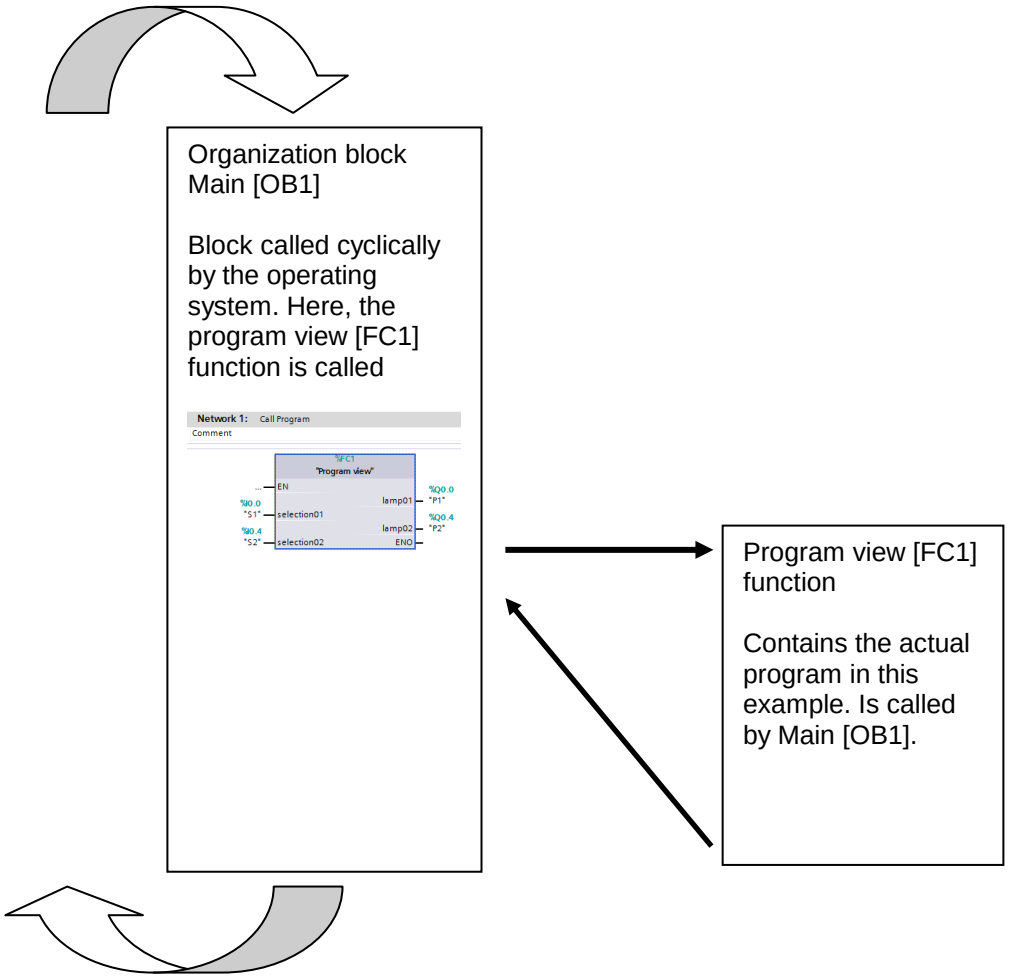
These global PLC tags are descriptive names with a comment for each input and output used in the program. The global PLC tags can then be accessed later during programming via their names. These global tags can be used in all blocks anywhere in the program.

For this purpose, select **'Control 001 [CPU 315F-2 PN/DP]'** and then **'PLC tags'** in the project tree. Double-click the **'Default tag table'** to open it and enter the names for the inputs and outputs as shown below. (→ Control 001 [CPU 315F-2 PN/DP] → PLC tags→ Default tag table)

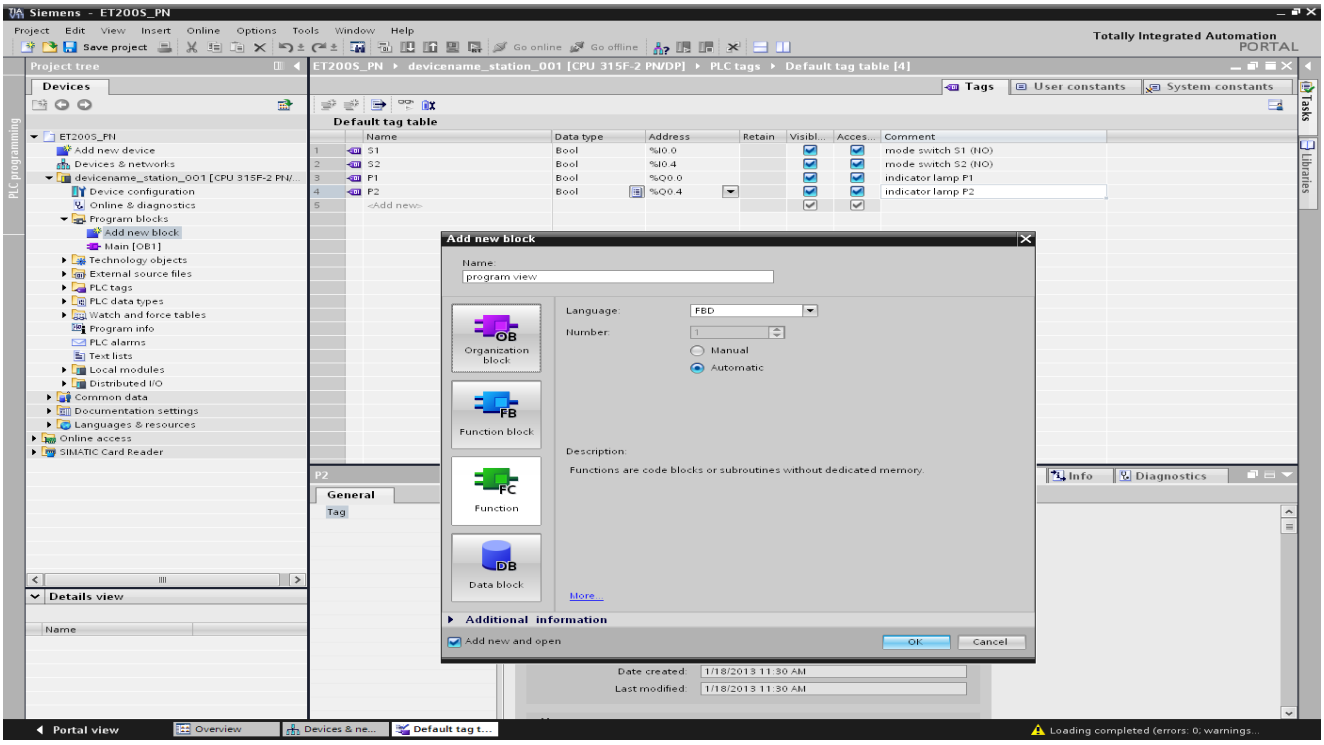


26. The program sequence is written to so-called blocks. The organization block Main [OB1] is already provided as standard. This block represents the interface to the CPU operating system and is automatically called and cyclically processed by this operating system.
From this organization block, additional blocks can be called in turn for structured programming, e.g., the program view [FC1] function.
The purpose is to break down an overall task into partial tasks. This makes it easier to solve these tasks and to test their functionality.

Program structure of the example:



27. To create the program view [FC1] function, select **'Control 001 [CPU 315F-2 PN/DP]'** in the project tree and then **'Program blocks'**. Afterwards, double-click **'Add new block'**. Select **'Function (FC)'** and assign the name **'program view'**. Specify the **'FBD'** function block diagram as programming language. The numbering will be automatic. Since this FC1 will be called using its symbolic name later in any case, the number does not play an important role. Click **'OK'** to accept your entries. (→ Control 001 [CPU 315F-2 PN/DP]' → Program blocks → Add new block → Function (FC) → program view → FBD → OK)



28. The **'program view [FC1]'** block then opens automatically. The interface of the block must be declared before the program can be written. In the interface declaration, the local tags known only in this block are defined.

The tags are divided into two groups:

• Block parameters that form the block interface for the call in the program.

Type	Designation	Function	Available in
Input parameters	Input	Parameters whose values are read by the block.	Functions, function blocks, and some types of organization blocks
Output parameters	Output / Return	Parameters whose values are written by the block.	Functions and function blocks
In/out parameters	InOut	A parameter whose value is read by the block when it is called and is written back by the block to the same parameter after it is processed.	Functions and function blocks

• Local data that is used for saving intermediate results.

Type	Designation	Function	Available in
Temporary local data	Temp	Tags that are used to store temporary intermediate results. Temporary data is retained for only one cycle.	Functions, function blocks, and organization blocks
Static local data	Static	Tags that are used for saving static intermediate results in the instance data block. Static data is retained until it is overwritten, which may be after several cycles.	Function blocks

29. The following tags are required in our example for declaration of the local tags.

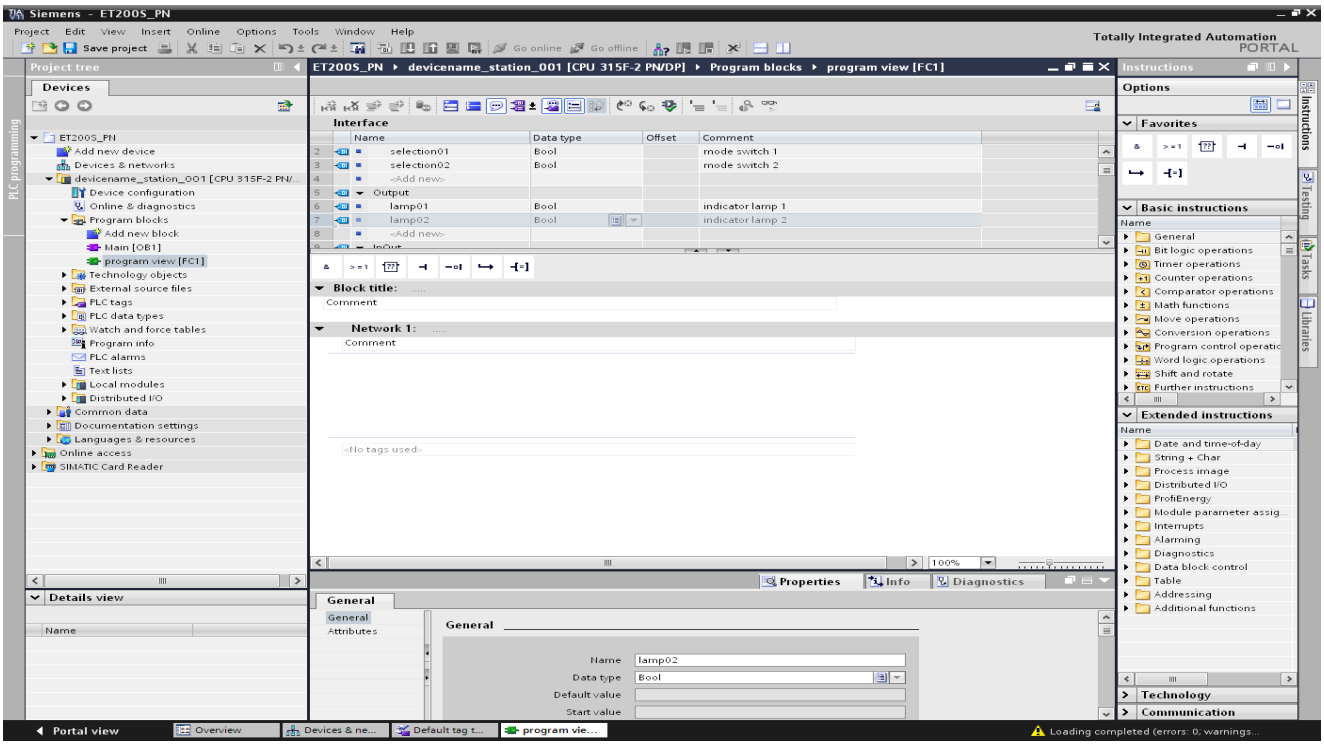
Input:

selection01	Selector switch 1
selection02	Selector switch 2

Output:

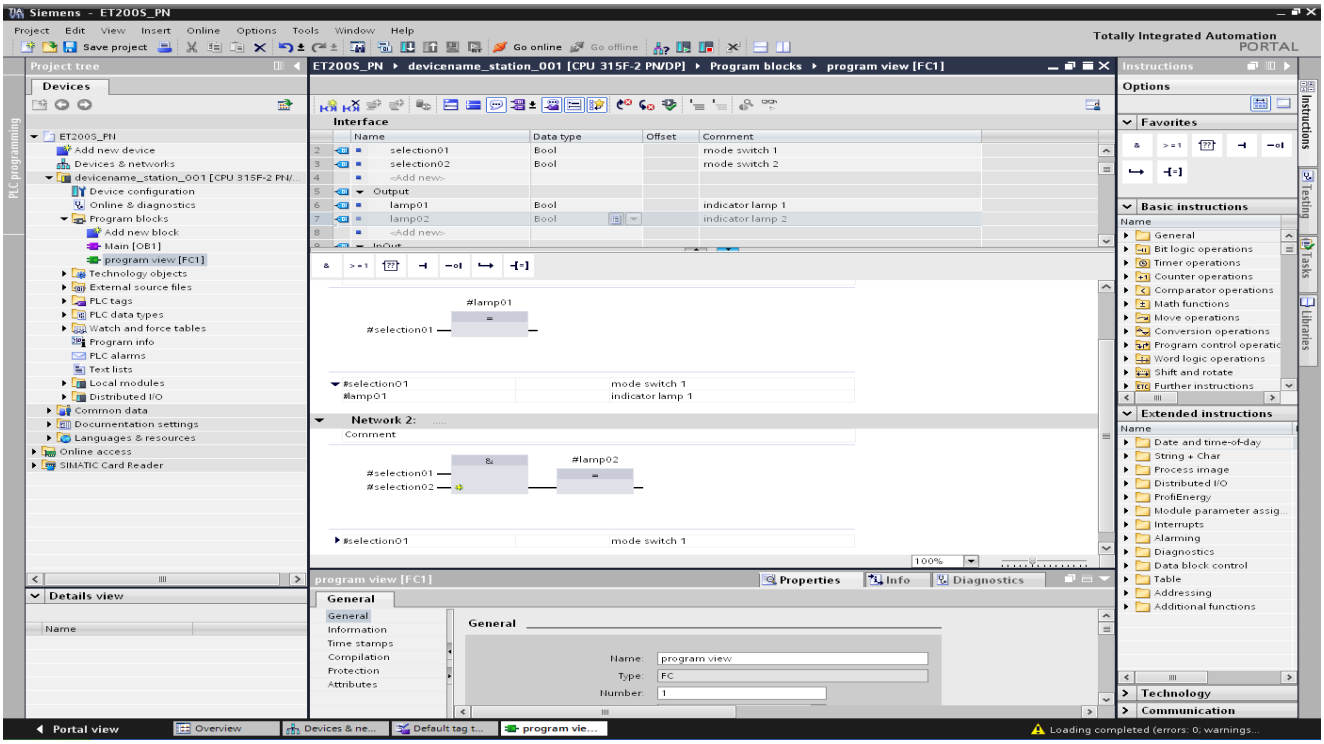
lamp01	Indicator light 1
lamp02	Indicator light 2

In this case, all of the tags are ‘Bool’ type tags. That is, they are binary tags that can only have the state ‘0’ (false) or ‘1’ (true).
All local tags should also be provided with a sufficiently descriptive comment for better understanding.

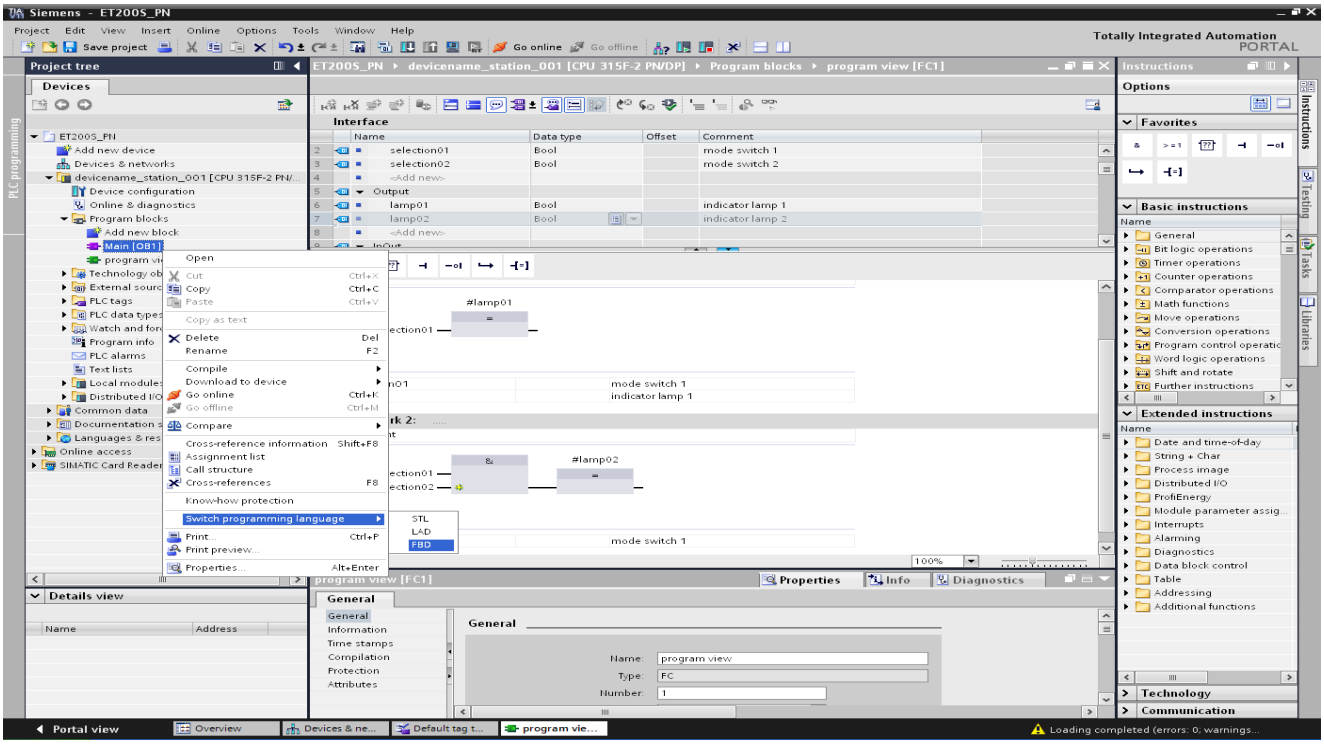


Note:
To avoid confusion with the PLC tags, it is helpful to write the local tags with lowercase letters.

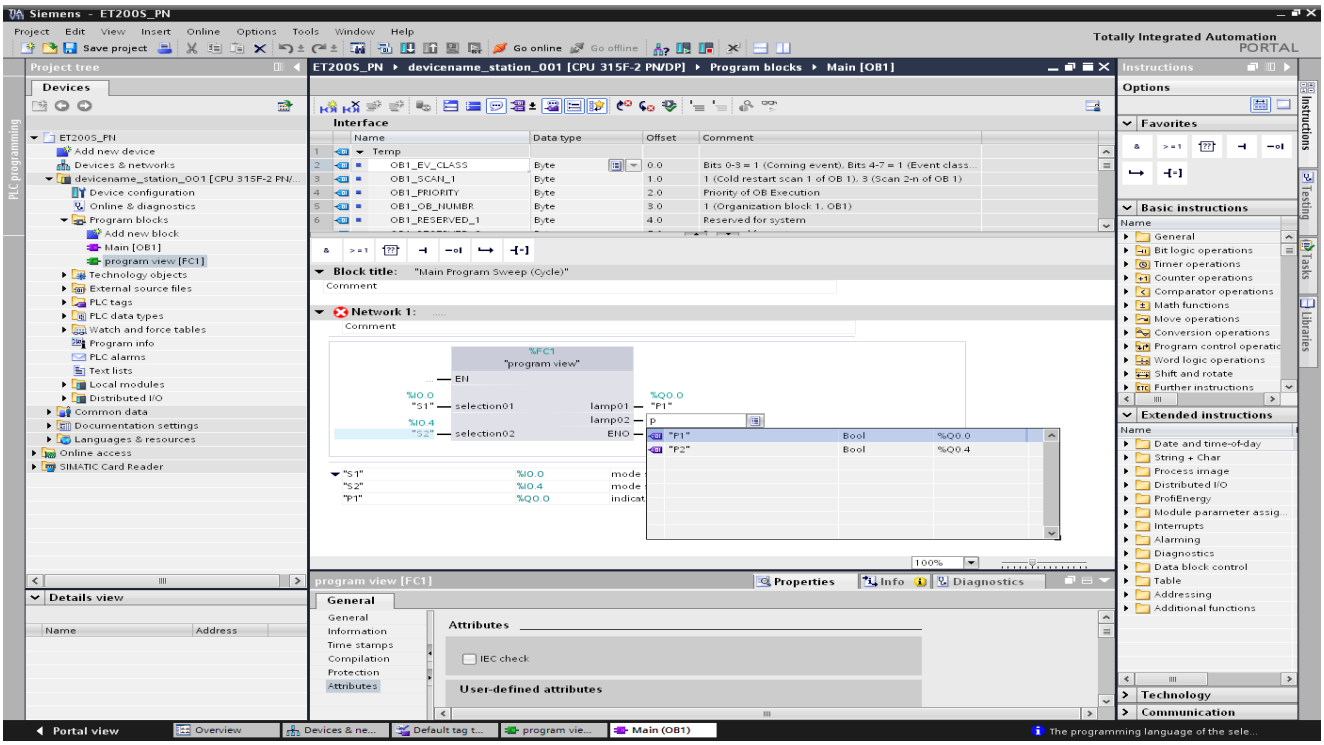
30. Once the local tags have been declared, you can start to create the program shown here. To provide a better overview, we program in networks. A new network can be inserted by clicking on the symbol  **'Insert network'**. Like the block itself, each network should be documented in the title line at least. If a longer text is needed for the description, the **'Comment'** field can be used in addition.



31. Before we double-click to open the block **'Main[OB1]'**, we select **'FBD'** as its programming language.
(→ Main[OB1] → Switch programming language → FBD → Main[OB1])

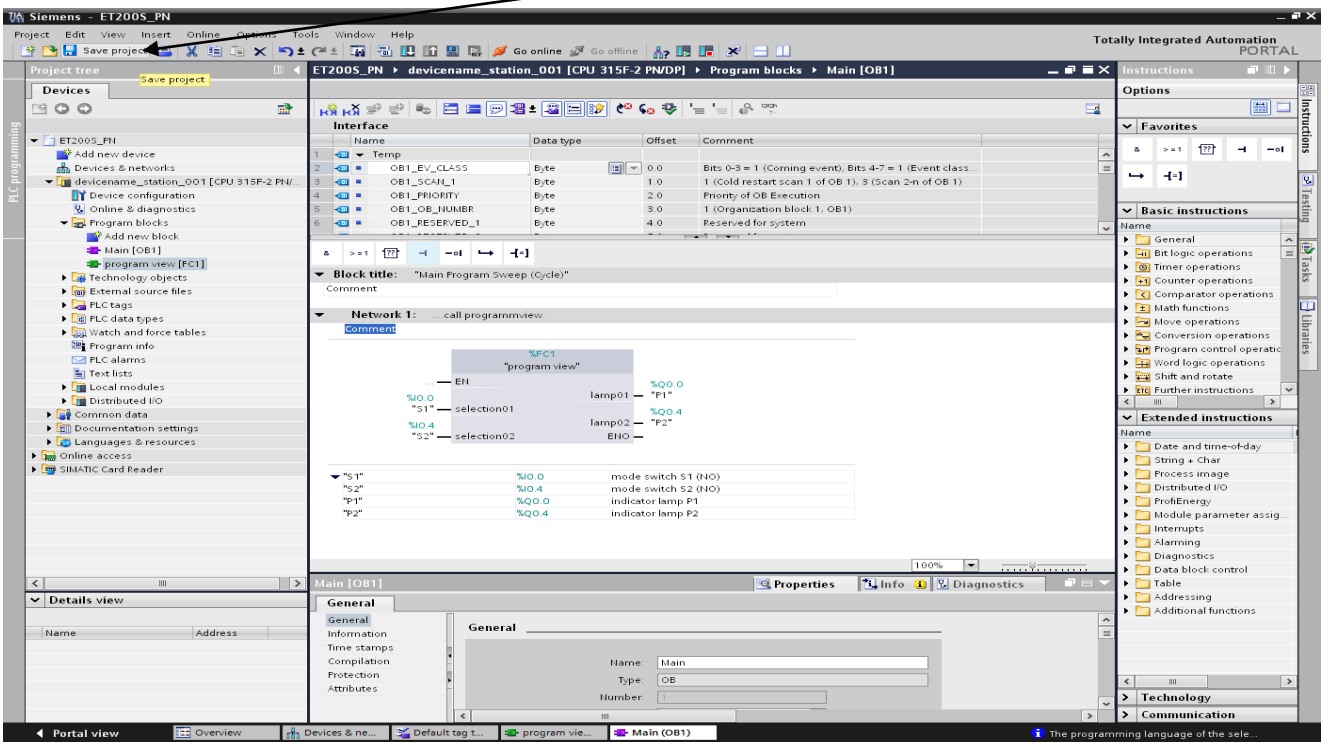


32. The **‘program view’** block can then be moved to Network 1 of the Main[OB1] block by means of drag-and-drop. The interface parameters of the **‘program view’** block must now be connected to the global PLC tags as shown here. Don't forget to document the networks in the Main[OB1] block. (→ Program view [FC1])

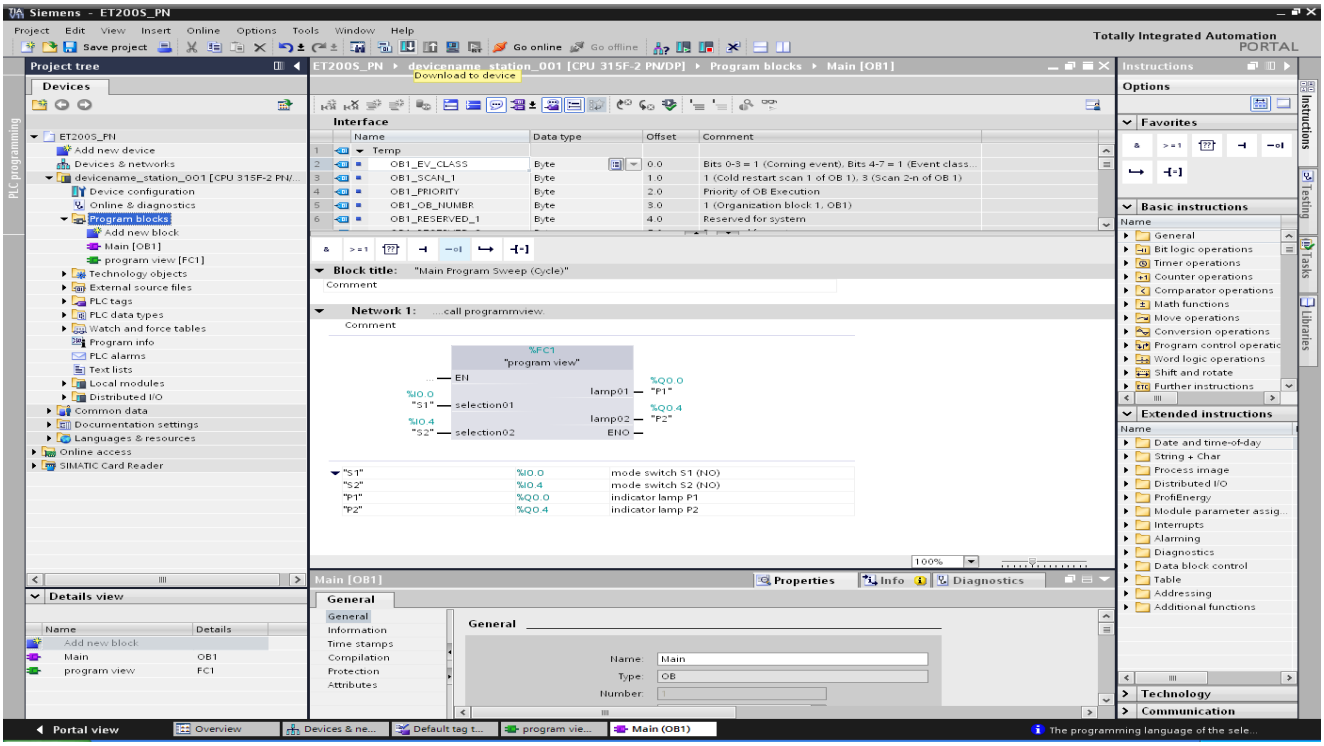


33. The button Save project is then used to save the project once again.

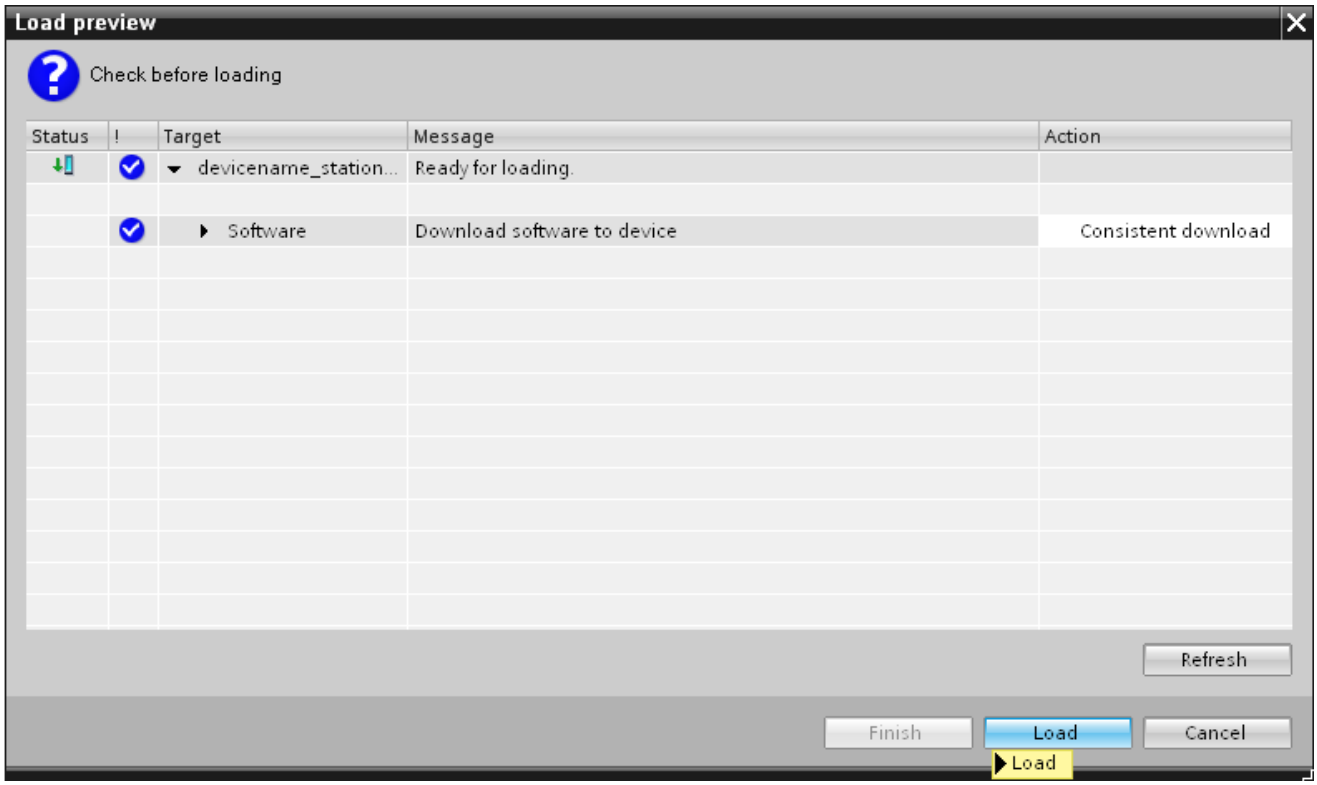
("EMERGENCY STOP" → −o| → Save project)



34. To only load the program blocks to the CPU, first select the **'Program blocks'** folder and click the Download to device icon  . (→ Program blocks → )



35. The program is now compiled automatically, and an overview of the steps to be performed is displayed once again for checking before the program is loaded. Click **'Load'** to start loading the program. (→ Load)



36. Clicking the  icon (Monitoring on/off) allows you to monitor the state of the input and output tags on the **'program view'** block. (→ )