

EUCHNER

Application



Integration of EKS with PROFNET IO interface into TIA Portal from V14 SP1

Contents

1.	About this document.....	3
1.1.	Version	3
1.2.	Scope.....	3
1.3.	Target group	3
1.4.	Supplementary documents	3
1.5.	Notice.....	3
2.	Components/modules used	4
2.1.	EUCHNER	4
2.2.	Others	4
2.3.	Software	4
3.	Functional description.....	4
4.	Overview of the communication data.....	5
4.1.	Input area (read process).....	5
4.2.	Output area (write process)	5
5.	Installing the GSD file.....	6
6.	Configuring the EKS	7
7.	Configuring different submodules.....	9
8.	Assigning PROFINET device names to the EKS.....	10
9.	Using the TIA Portal library (from TIA V14 SP1)	11
9.1.	Retrieving the library	11
9.2.	Library for S7-1200/1500.....	12
9.2.1.	Copying the blocks for the EKS.....	12
9.2.2.	Identifying the hardware identifiers of the submodules.....	13
9.2.3.	Calling and description of the EKS_Communication_FC in OB1	13
9.3.	Library for S7-300/400.....	15
9.3.1.	Copying the blocks for the EKS.....	15
9.3.2.	Identifying input/output address of the submodules	16
9.3.3.	Calling and description of the EKS_Communication_FC in OB1	16
10.	Reading and writing Electronic-Key data	18
10.1.	Transferring program to the PLC.....	18
10.2.	Reading contents of the Electronic-Key memory using a watch and force table	18
10.3.	Writing contents of the Electronic-Key memory using a watch and force table.....	18
11.	Important note – please observe carefully!	20

1. About this document

1.1. Version

Version	Date	Change/addition	Chapter
01-09/18	03.09.2018	Prepared	All
02-01/20	17.01.2020	Update: GSDML file	5
02-01/20	17.01.2020	Update: identification of hardware identifier	9.2.2
02-01/20	17.01.2020	Expansion with S7-300/400 library	9.3
02-01/20	17.01.2020	Update: important note	11

1.2. Scope

The purpose of this document is the integration and configuration of the EKS with PROFINET IO interface into SIEMENS TIA Portal from version 14 SP1.

1.3. Target group

Design engineers and installation planners for safety systems on machines, as well as setup and servicing staff possessing special expertise in handling safety components as well as expertise in the installation, setup, programming and diagnostics of programmable logic controllers (PLCs) and bus systems.

1.4. Supplementary documents

The overall documentation for this application consists of the following documents:

Document title (document number)	Contents	
Manual (2516210)	Electronic-Key-System Manual EKS and EKS FSA with PROFINET IO interface	
Possibly enclosed data sheets	Item-specific information about deviations or additions	

1.5. Notice

This application is based on the manual for the EKS with PROFINET IO interface. Please refer to the manual for the technical details and other information. In the rest of this document the EKS with PROFINET IO interface is referred to as the *EKS* for short.

2. Components/modules used

2.1. EUCHNER

Description	Order number / item
EKS compact with PROFINET IO interface	106305 / EKS-AHX-G01-ST02/03
EKS compact FSA with PROFINET IO interface	106306 / EKS-AHXA-G01-ST02/03/04
EKS modular with PROFINET IO interface	122352 / EKS-A-AIX-G18
EKS modular FSA with PROFINET IO interface	122353 / EKS-A-AIXA-G18



TIP!

More information and downloads about the aforementioned EUCHNER products can be found at www.euchner.com. Simply enter the order number in the search box.

2.2. Others

Description	Order number / item
SIMATIC S7-1215 FC DC/DC/DC	6ES7 215-1AF40-0XB0
SIMATIC S7-315F-2 PN/DP	6ES7315-2FH13-0AB0

2.3. Software

Description	Version
Totally Integrated Automation Portal	Version V14 SP1 update 9
STEP 7 Professional	Version V14 SP1 update 9

3. Functional description

EKS PROFINET devices are read/write systems with electronics for the inductive bidirectional interface to the transponder and interface electronics.

The system is connected via the integrated PROFINET interface, which is designed as an RJ45 socket. A separate switch may be required for the PROFINET connection. The EKS does not have an integrated switch.

The current state of the Electronic-Key adapter is displayed using a 3-color LED.

The Electronic-Key is placed on the Electronic-Key adapter for operation. The power supply for the transponder and the data are transferred between the Electronic-Key adapter and the Electronic-Key without using any contacts.

4. Overview of the communication data

4.1. Input area (read process)

PROFINET	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0 (status byte)	Job in progress	Job finished	-	-	-	-	Electronic-Key detected	Device ready for operation

PROFINET	Description	Function
Byte 1 . . . Byte 127	Receive data	Max. 124 bytes user data from the Electronic-Key plus 3 bytes reserve. If fewer bytes of data were selected during configuration, these are filled with 0hex.

4.2. Output area (write process)

PROFINET	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0 (command byte)	-	-	-	-	-	-	-	Write Electronic-Key

PROFINET	Description	Function
Byte 1	Start address	Defines first byte in the memory in the Electronic-Key that is written on setting bit no. 0 in the command byte. Start address of user data: byte no. 0, 4, 8 ... 112.
Byte 2	Number of bytes	Defines the number of bytes in the memory in the Electronic-Key that are written on setting bit no. 0 in the command byte. Number of bytes of user data: 4, 8, 12 ... 116 bytes.
Byte 3	Not used	
Byte 4 . . . Byte 119	Transmit data	If bit no. 0 in the command byte is set to 1, the content of these bytes is written to the Electronic-Key starting from the start address defined.
Byte 120 . . Byte 127	Not used	



TIP!

You will find further information in the manual.



NOTE!

On the Electronic-Key read/write with 116 bytes freely programmable, the memory is organized in 4-byte blocks. This means the start address must be given for writing in the range byte number 0 to byte number 112, always in 4-byte steps (byte number 0, 4, 8 ... 112). Also a multiple of 4-byte sized blocks must always be written (4, 8, 12 ... 116 bytes).

During reading it is possible to access the memory byte-by-byte without the above-mentioned restriction for writing.

5. Installing the GSD file

You will require the corresponding GSD file in GSDML format to integrate the EKS into TIA Portal:

▸ GSDML-V2.31-Euchner-EKS_2524496-YYYYMMDD

You will find the GSD file in the download area at www.euchner.com. Always use the latest GSD file.

Please proceed as follows to install the GSD file in TIA Portal V14:

1. Click *Options* and select *Manage general station description files (GSD)*.

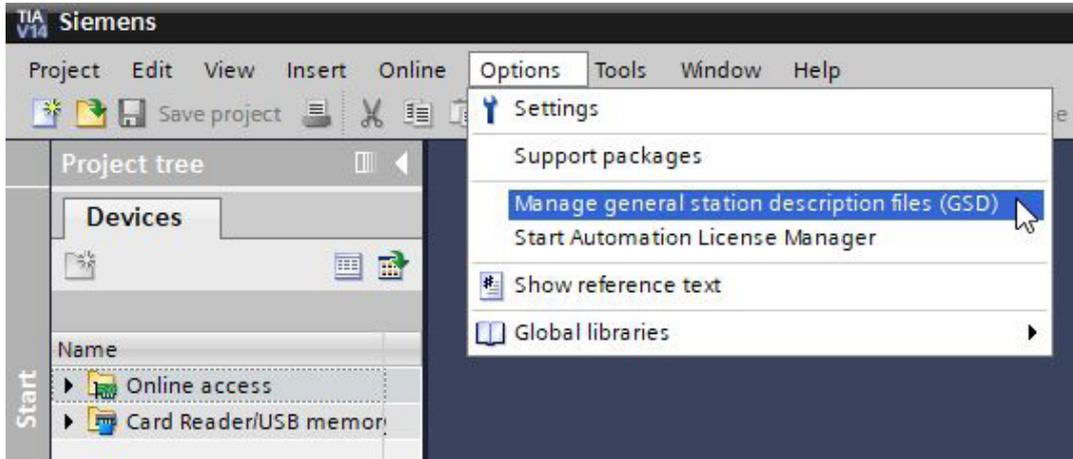


Figure 1: GSD file selection

2. Select the source path for the GSD file and install it.

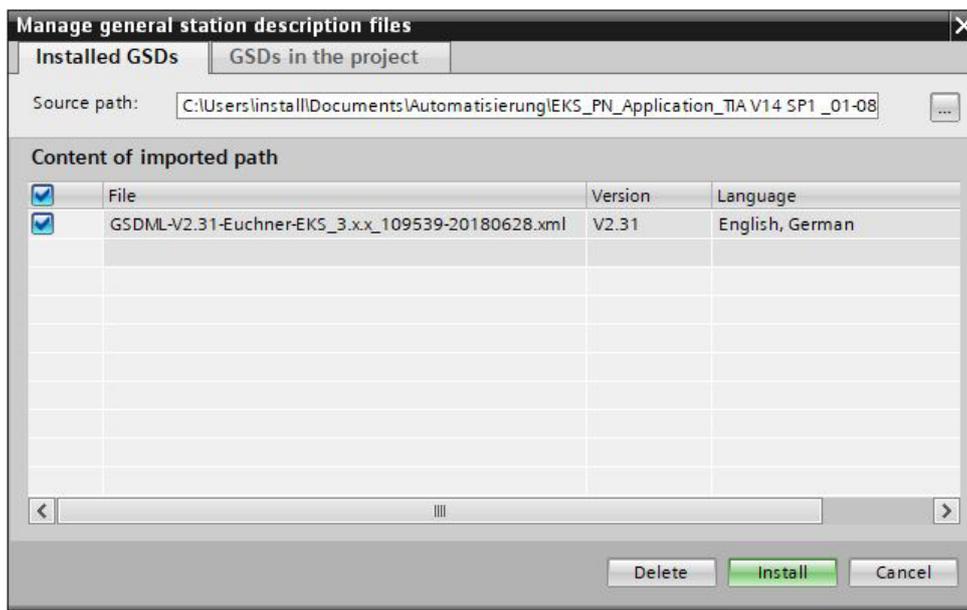


Figure 2: GSD file installation

6. Configuring the EKS

Select the corresponding EKS from the *Hardware catalog* and add it to the *Network view* using drag & drop. Then assign it to the CPU.

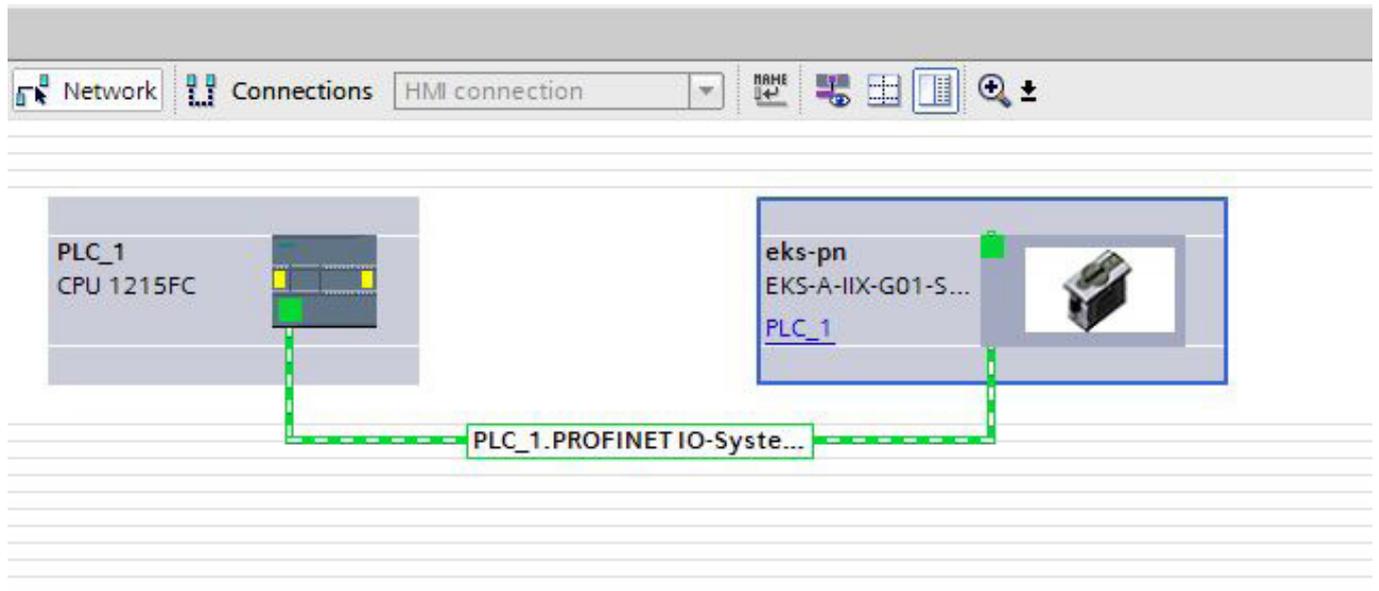


Figure 3: Network view for the EKS

The following PROFINET parameters must be set:

- Device name (factory settings from GSD file): [eks-pn].
- IP address: optionally fixed or dynamic

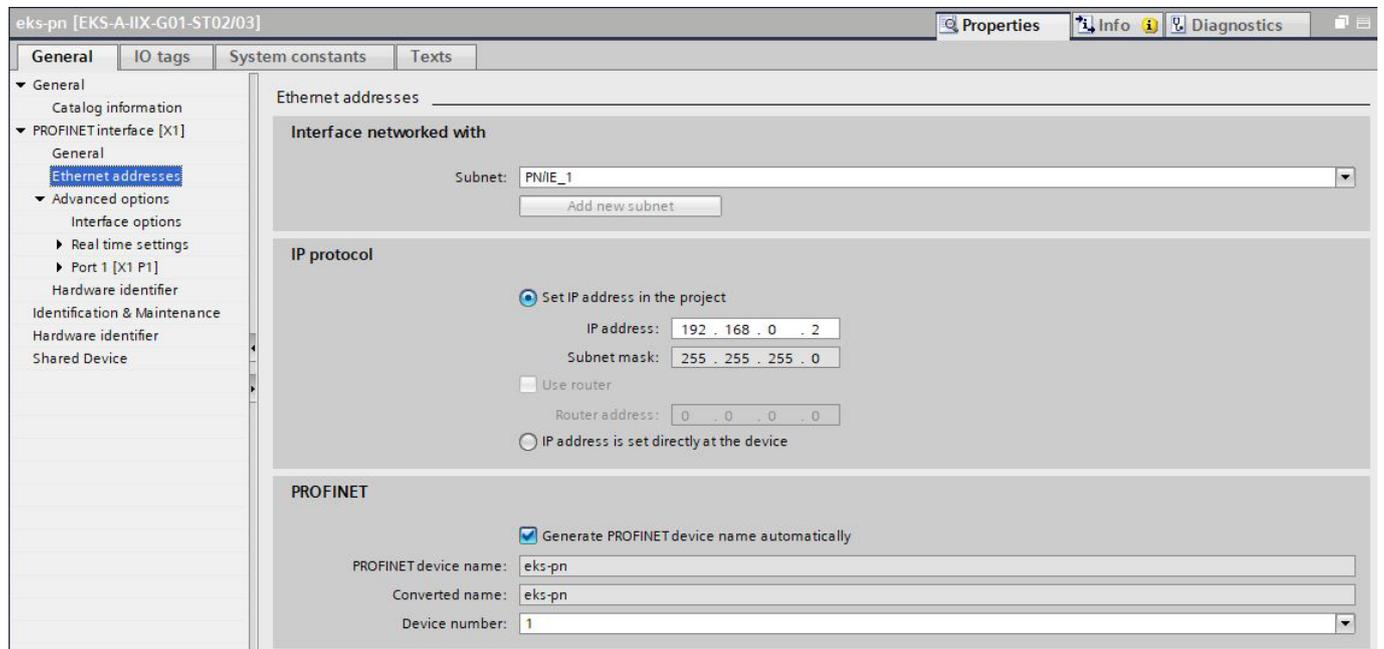


Figure 4: PROFINET parameters

- IO cycle real time settings
Update time: *Calculate update time automatically* (recommended)
Watchdog time: *Accepted update cycles without IO data: 3* (recommended)

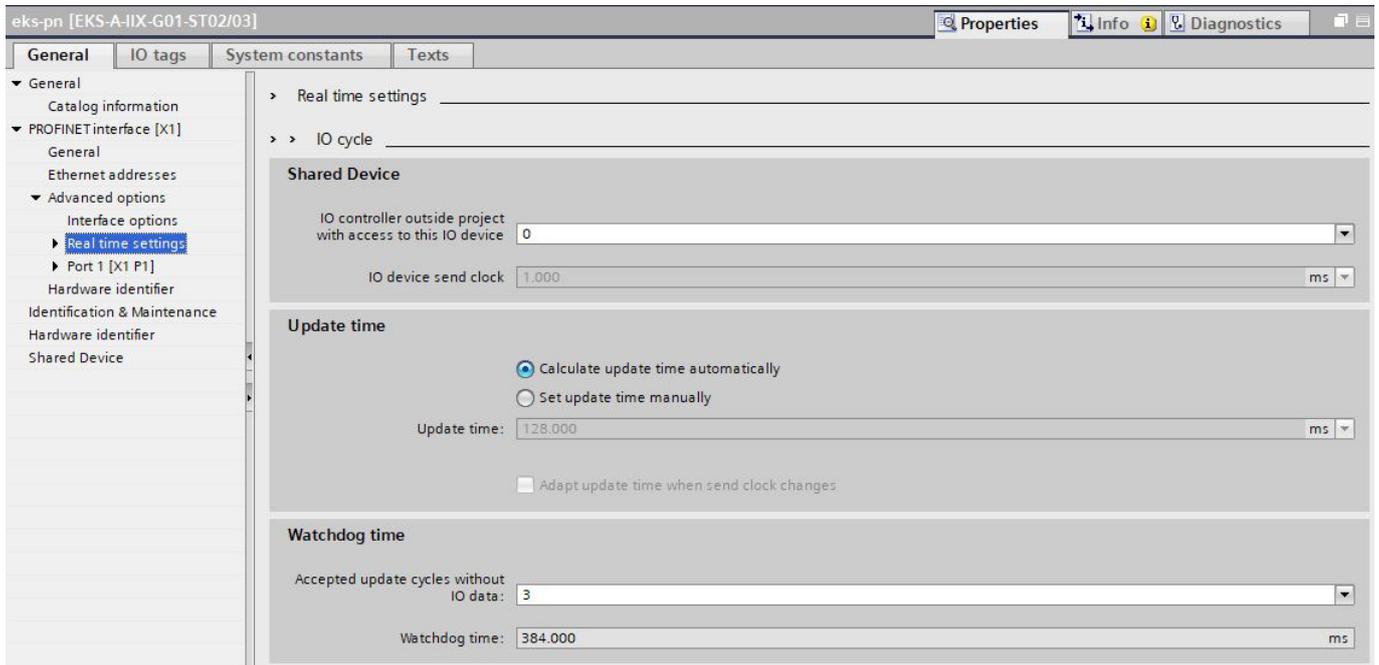


Figure 5: PROFINET *real time settings*

7. Configuring different submodules

Open the EKS in the *Device view*. The preconfigured submodules are *Read: 128 bytes* and *Write: 128 bytes*. With these settings you will obtain the contents of the entire memory in the Electronic-Key. If you require only a certain area on the Electronic-Key or you do not need the complete contents of the Electronic-Key, these modules can be changed. For this purpose, delete the corresponding submodule from the *Device view* and add the required submodule using the mouse and drag & drop.

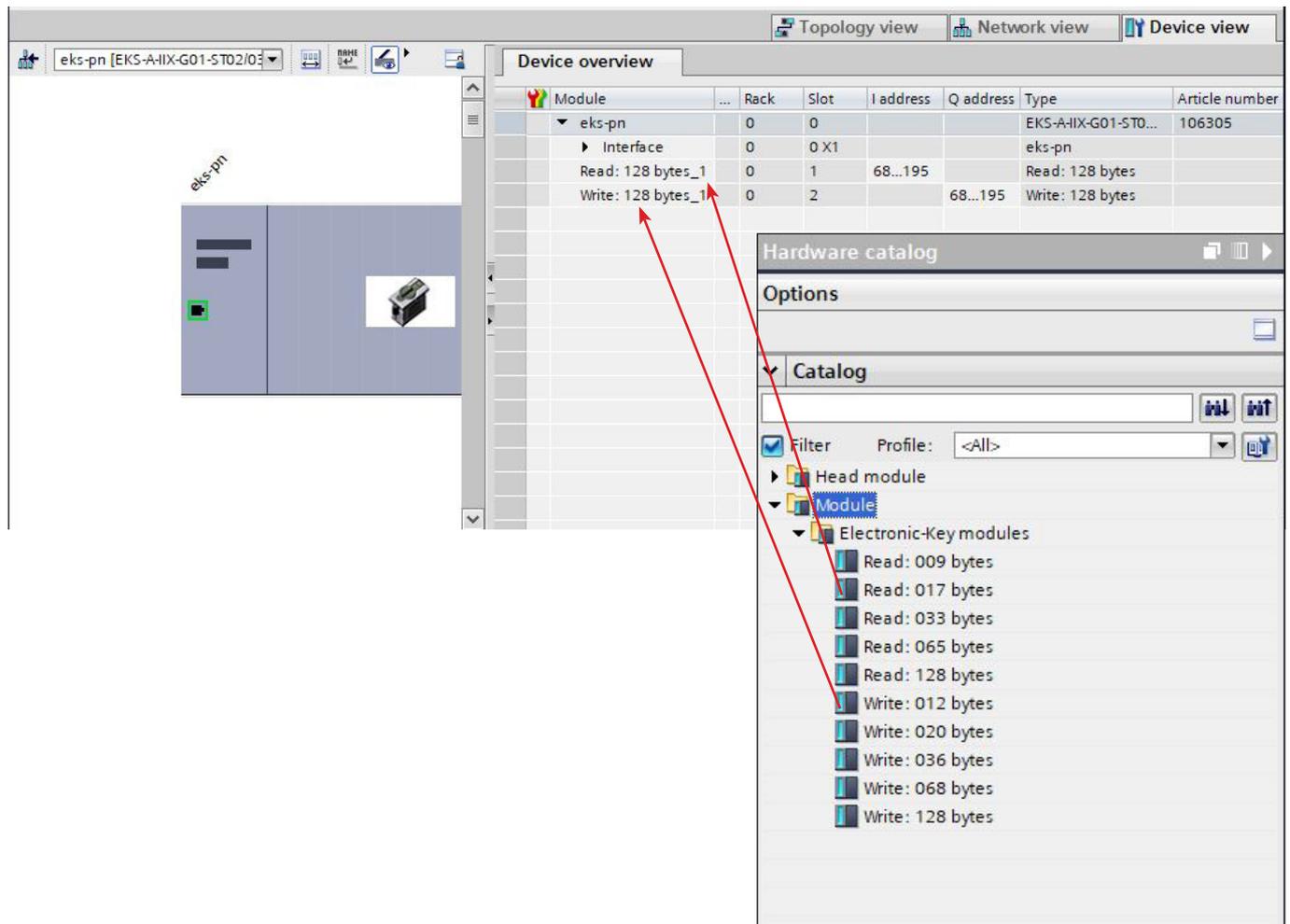


Figure 6: Replacing submodules (example)



NOTE!

For other parameter settings (start address/number of bytes), please refer to the manual.

8. Assigning PROFINET device names to the EKS

1. Open the *Device view* and select the bus module EKS. Use *Assign device name*.

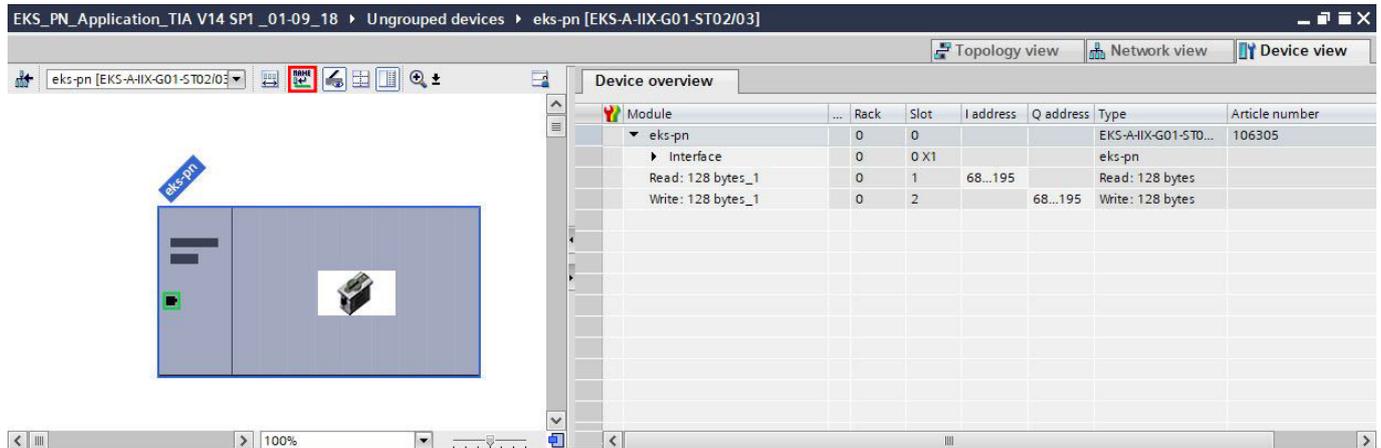


Figure 7: Device view

2. Use *Update list* to display all devices of the same type. Compare the MAC address on the type label with the MAC address of the devices available in the network and assign the PROFINET name to the MAC address using *Assign name*.

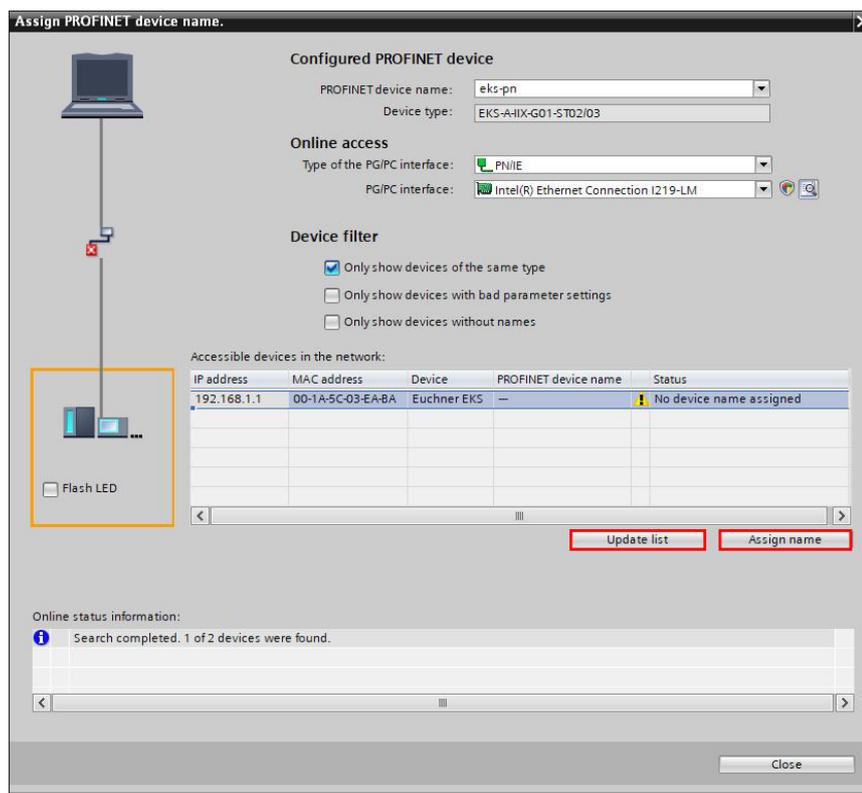


Figure 8: Assigning device name



TIP!

As an alternative to the MAC address comparison, you can use *Flash LED* to determine whether you have selected the correct device.

9. Using the TIA Portal library (from TIA V14 SP1)

In the library you will find templates you can copy to help you during programming. The memory in the Electronic-Key is read out using the *DPRD_DAT* block. The Electronic-Key memory can be written using the *DPWR_DAT* block.

Open the page with EKS applications in the download area at www.euchner.com and download the library for the EKS.

9.1. Retrieving the library

1. Change to the *Task Card* view (shortcut: *Ctrl+3*) and select *Libraries*.
2. Open the context menu with a right click on the *Global libraries* area and select *Retrieve library...* Select the folder with the downloaded library and retrieve it to the required destination folder. The library is updated while retrieving the library using TIA Portal V15 or later because it was prepared using TIA Portal V14 SP1.

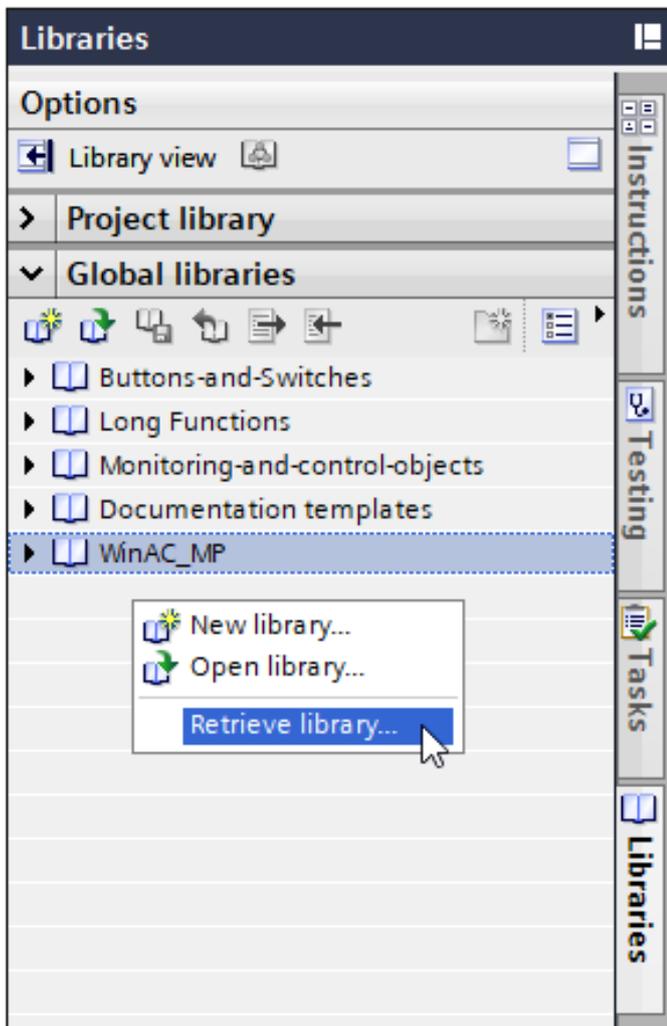


Figure 9: Retrieving library

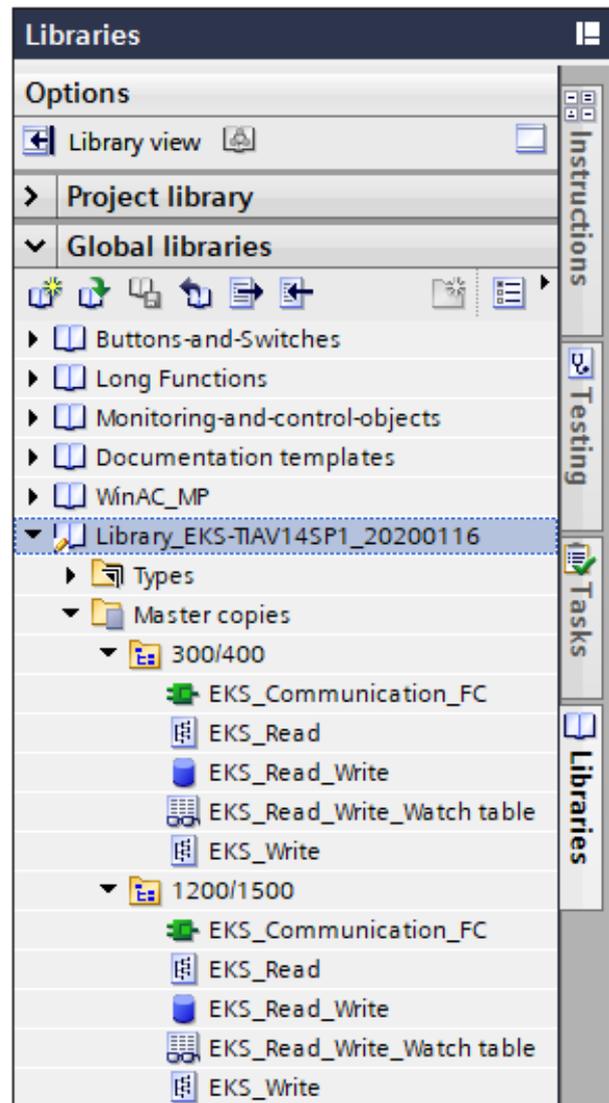


Figure 10: Opened library



TIP!

If the library has already been retrieved, select *Open library...* to add the library to the project.

9.2. Library for S7-1200/1500

9.2.1. Copying the blocks for the EKS

1. Open the *library* and copy the blocks from the folder *1200/1500* as per the numbers to the *project tree* folders.

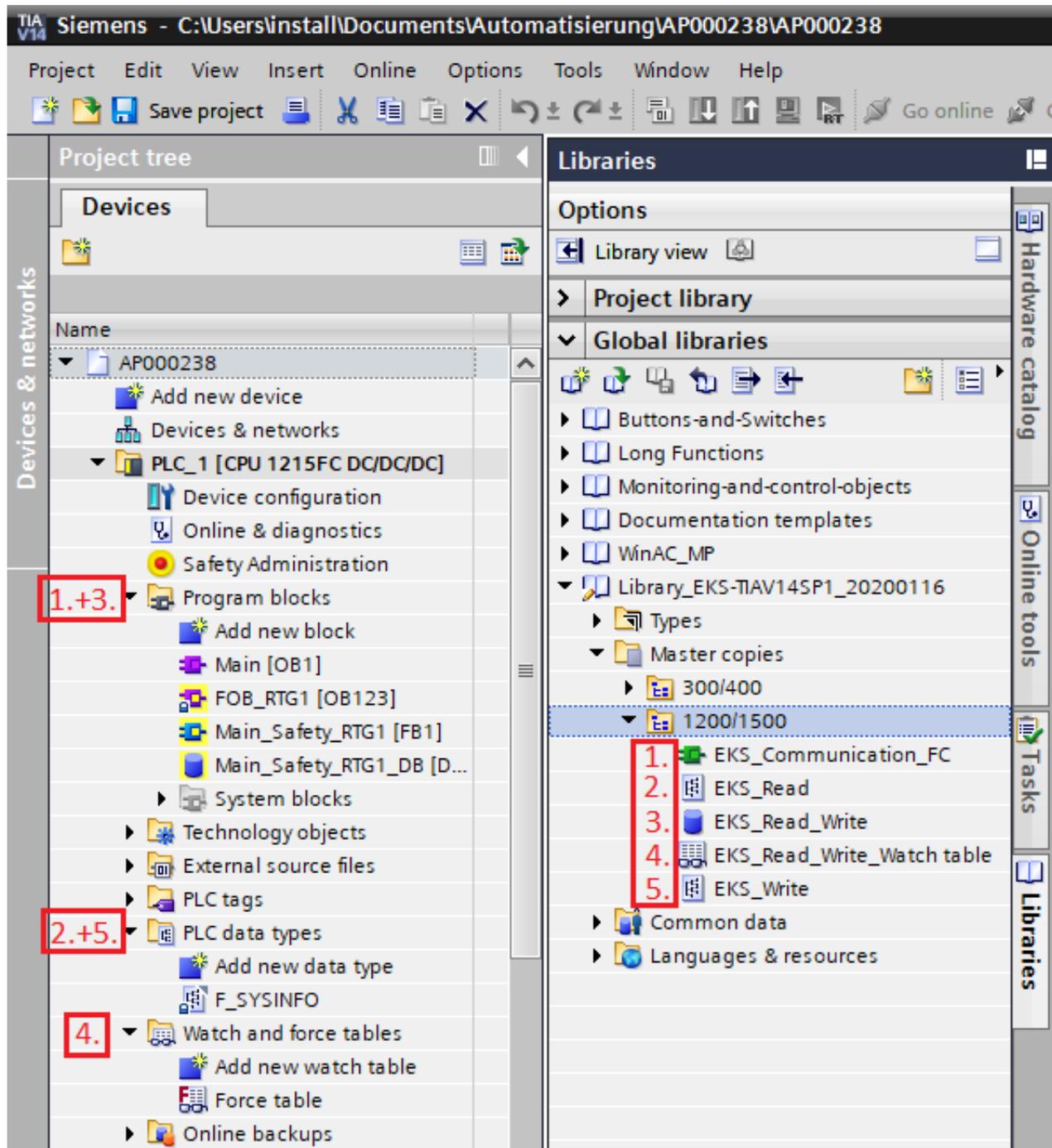


Figure 11: Copying blocks to the project



TIP!

Within your project you can also switch the language for the block comments between German and English.

9.2.2. Identifying the hardware identifiers of the submodules

On adding the submodules in the *Device view*, hardware identifiers are automatically assigned to the modules. Refer to the *System constants* tab. Select the corresponding module. The hardware identifiers are needed to read the data from the Electronic-Key using the *DPRD_DAT* block and to write data to the Electronic-Key using the *DPWR_DAT* block.

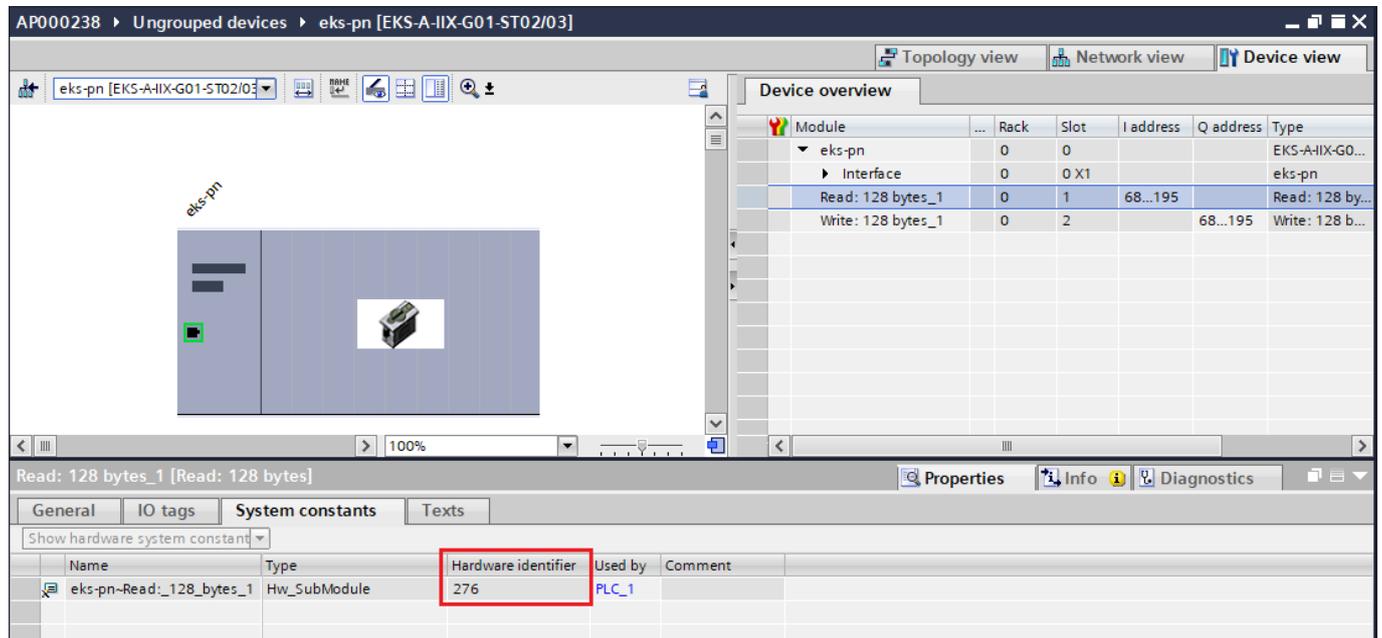


Figure 12: Submodule system constant (*hardware identifier*)

9.2.3. Calling and description of the EKS_Communication_FC in OB1

Next, the FC (*EKS_Communication_FC*) must be called in OB1 (Main). For this purpose drag the FC from the *project tree* directly to an OB1 network.

Parameter	Data type	Value	Description
EKS_PN_ReadModule	Word	276 / "eks-pn-Read:_128_bytes_1"	Hardware identifier from EKS submodule "Read"
EKS_PN_WriteModule	Word	277 / "eks-pn-Write:_128_bytes_1"	Hardware identifier from EKS submodule "Write"
EKS_ReadData	UDT (EKS_Read)	"EKS_Read_Write".ReadData	Loads the data read into the DB (EKS_Read_Write)
EKS_WriteData	UDT (EKS_Write)	"EKS_Read_Write".WriteData	Loads the data to be written into the EKS output area
Error_Read	Int	"EKS_Read_Write".Error_Read	"Read" error message
Error_Write	Int	"EKS_Read_Write".Error_Write	"Write" error message

Table 1: *EKS_Communication_FC*: description of module interface

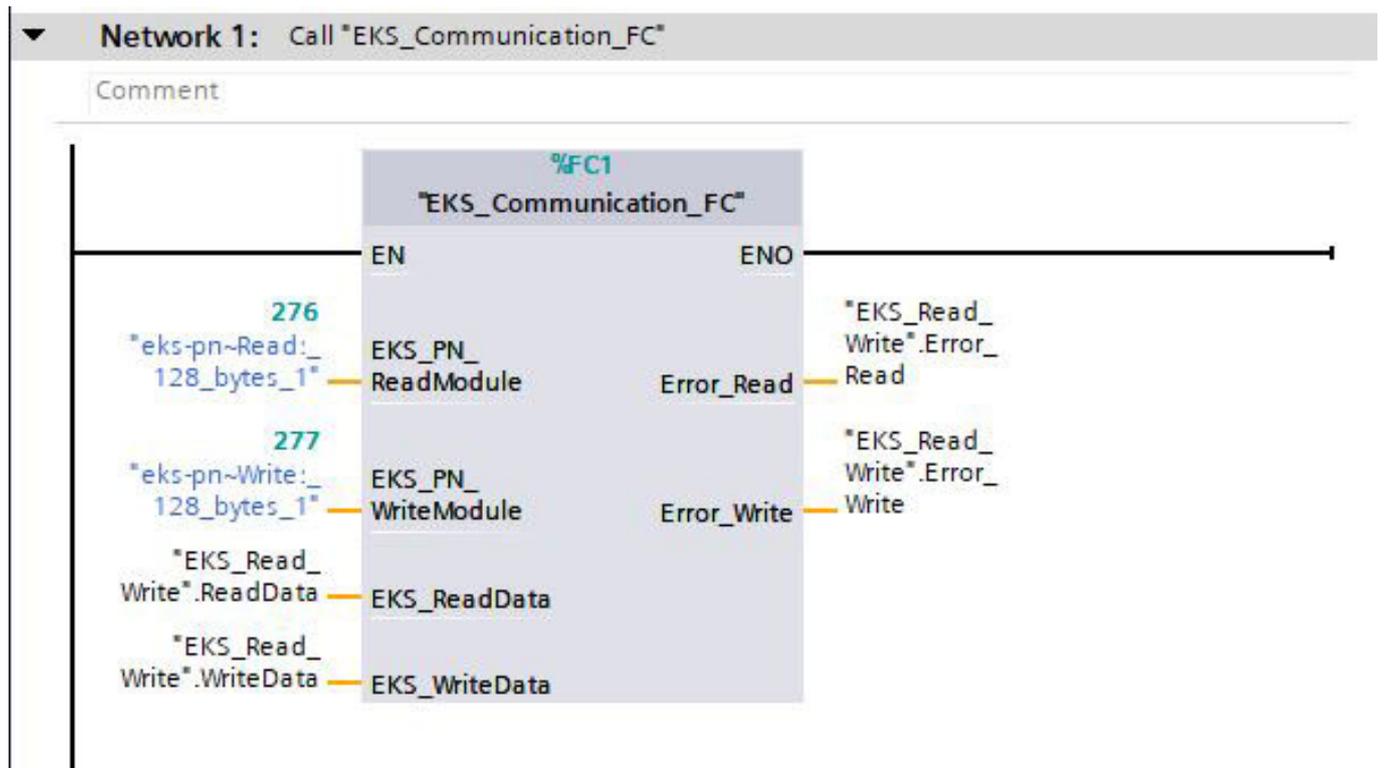


Figure 13: Complete call EKS_Communication_FC in OB1

9.3. Library for S7-300/400

9.3.1. Copying the blocks for the EKS

1. Open the *library* and copy the blocks from the folder *300/400* as per the numbers to the *project tree* folders.

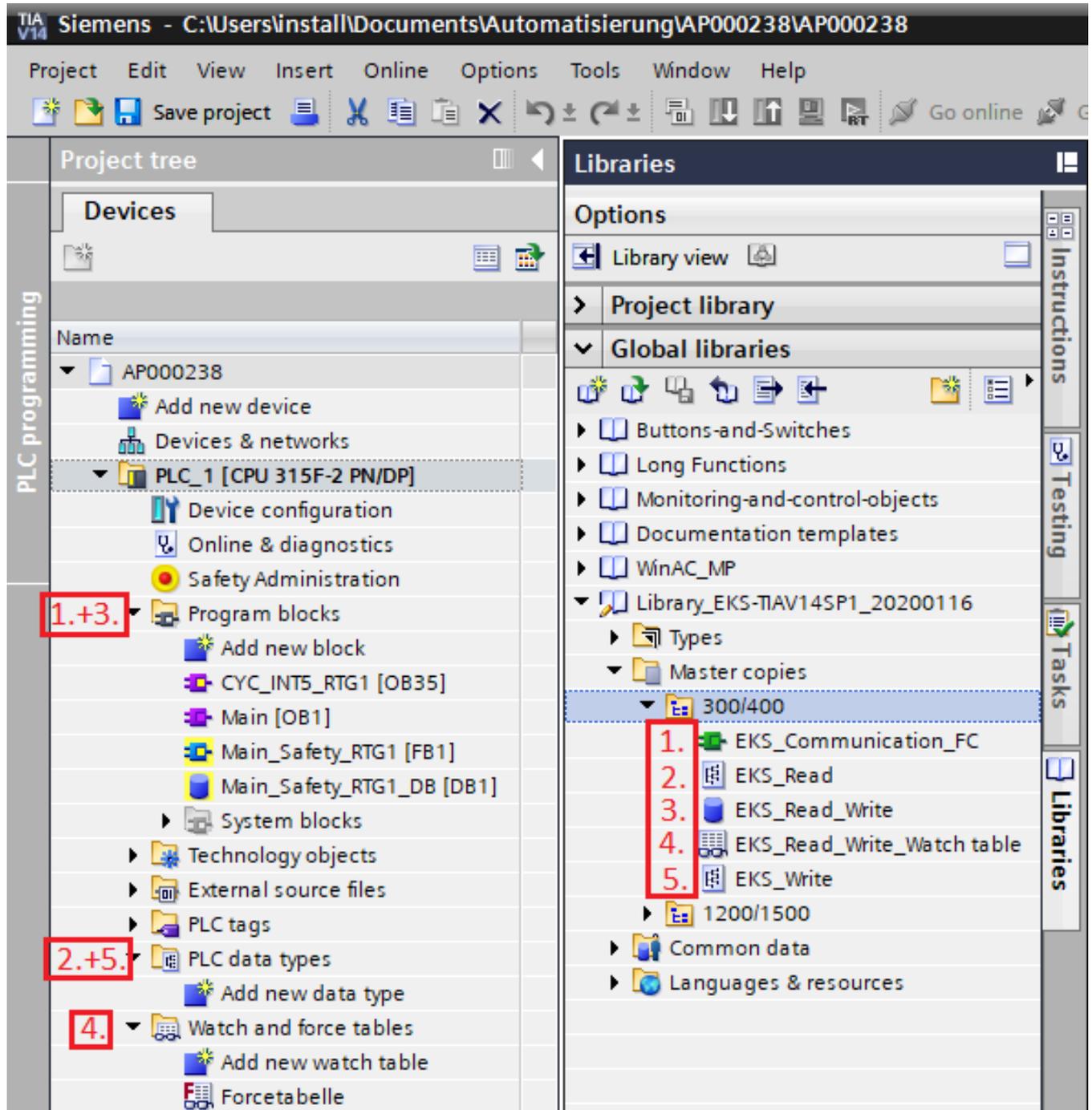


Figure 14: Copying blocks to the project



TIP!

Within your project you can also switch the language for the block comments between German and English.

9.3.2. Identifying input/output address of the submodules

On adding the submodules in the *Device view*, input and output addresses are automatically assigned to the modules. You can find them in the *Device view*. The input and output addresses are needed to read the data from the Electronic-Key using the *DPRD_DAT* block and to write data to the Electronic-Key using the *DPWR_DAT* block.

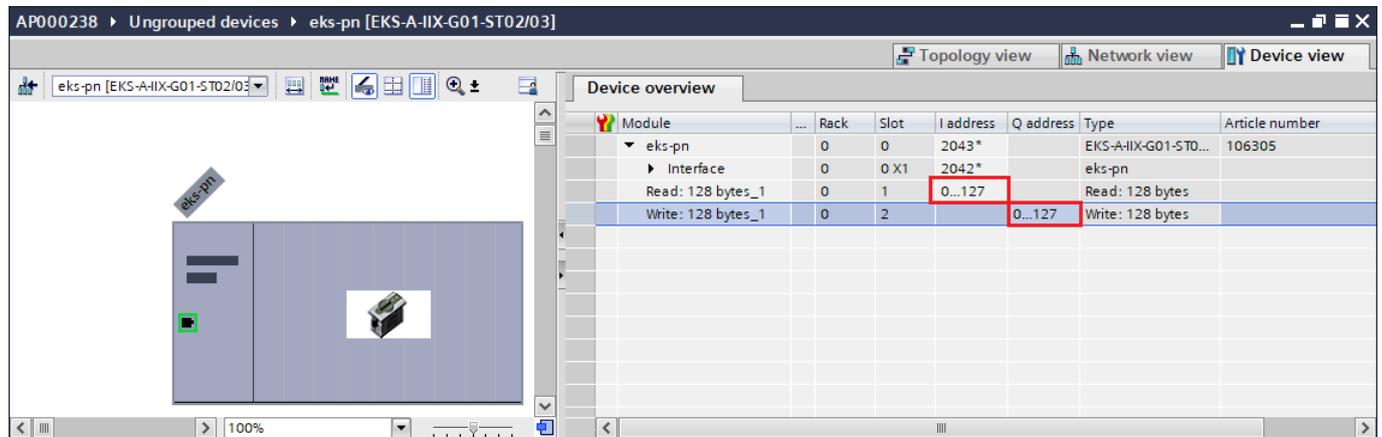


Figure 15: Submodule input/output address

9.3.3. Calling and description of the EKS_Communication_FC in OB1

Next, the FC (*EKS_Communication_FC*) must be called in OB1 (Main). For this purpose drag the FC from the *project tree* directly to an OB1 network.

Parameter	Data type	Value	Description
EKS_PN_ReadModule	Word	0	First byte of input address from EKS "Read" submodule
EKS_PN_WriteModule	Word	0	First byte of output address from EKS "Write" submodule
EKS_ReadData	UDT (EKS_Read)	"EKS_Read_Write".ReadData	Loads the data read into the DB (EKS_Read_Write)
EKS_WriteData	UDT (EKS_Write)	"EKS_Read_Write".WriteData	Loads the data to be written into the EKS output area
Error_Read	Int	"EKS_Read_Write".Error_Read	"Read" error message
Error_Write	Int	"EKS_Read_Write".Error_Write	"Write" error message

Table 2: EKS_Communication_FC: description of module interface

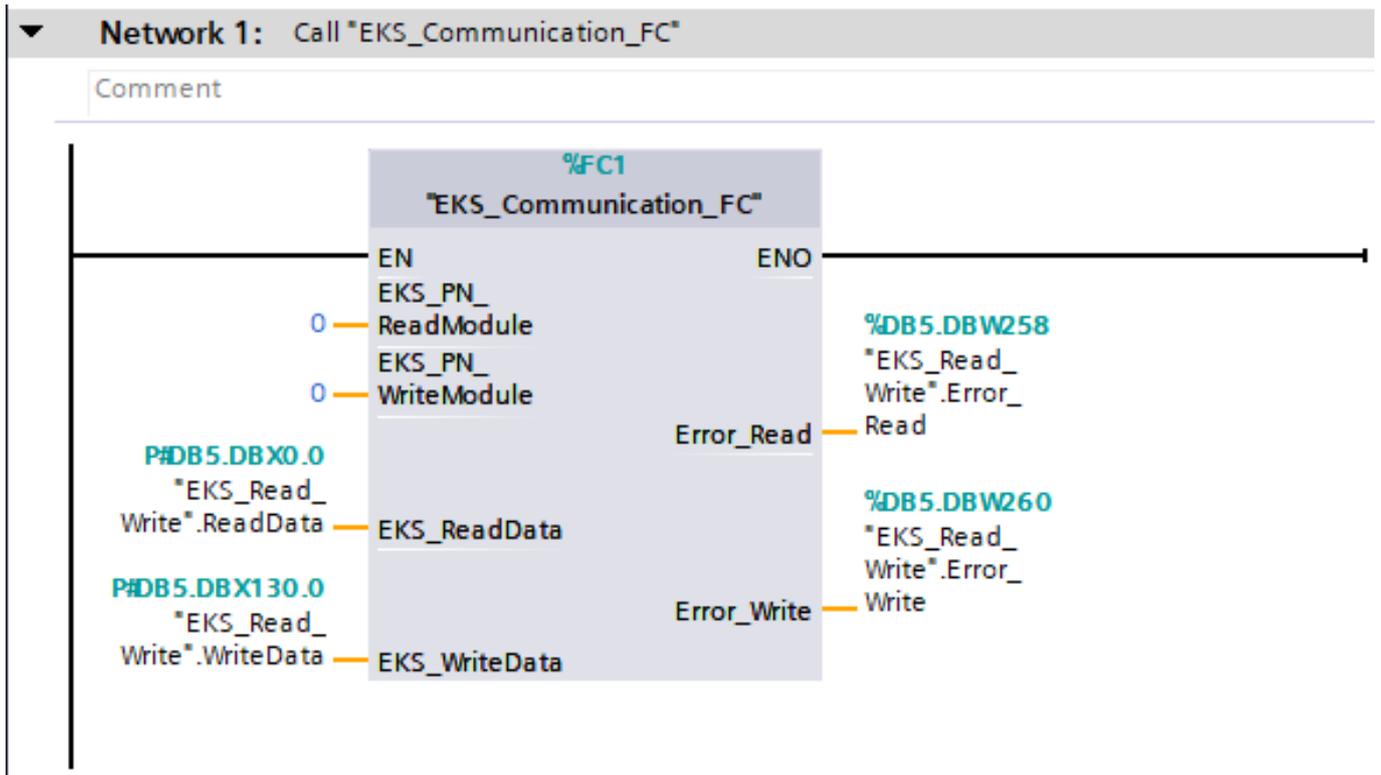


Figure 16: Complete call EKS_Communication_FC in OB1

10. Reading and writing Electronic-Key data

10.1. Transferring program to the PLC

Transfer the program including the hardware configuration to your control system.

10.2. Reading contents of the Electronic-Key memory using a watch and force table

In the example below, an extract of the status byte and the first 8 bytes of the Electronic-Key memory are displayed using a watch and force table (EKS_Read_Write). Go online and click *Monitor all* in the watch and force table. To read the Electronic-Key data, simply place the Electronic-Key in the Electronic-Key adapter. The Electronic-Key data are transferred cyclically to the PLC.

	Name	Address	Display format	Monitor value	Modify value		Comment
13							
14	// Read EKS Key						
15	"EKS_Read_Write".ReadData.Device_ready_for_operation		Bool	<input checked="" type="checkbox"/> TRUE			
16	"EKS_Read_Write".ReadData."Electronic-Key_detected"		Bool	<input checked="" type="checkbox"/> TRUE			
17	"EKS_Read_Write".ReadData.Job_finished		Bool	<input type="checkbox"/> FALSE			
18	"EKS_Read_Write".ReadData.Job_in_progress		Bool	<input type="checkbox"/> FALSE			
19	"EKS_Read_Write".ReadData."Receive_data[0]"		Character	'*'			
20	"EKS_Read_Write".ReadData.Receive_data[1]		Character	'Z'			
21	"EKS_Read_Write".ReadData.Receive_data[2]		Character	'*'			
22	"EKS_Read_Write".ReadData.Receive_data[3]		Character	'Z'			
23	"EKS_Read_Write".ReadData.Receive_data[4]		Character	'*'			
24	"EKS_Read_Write".ReadData.Receive_data[5]		Character	'Z'			
25	"EKS_Read_Write".ReadData.Receive_data[6]		Character	'*'			
26	"EKS_Read_Write".ReadData.Receive_data[7]		Character	'Z'			
27		<Add new>					

Figure 17: Reading Electronic-Key data example



NOTE!

It is to be noted that, due to a 16-bit limit imposed by Siemens, the first byte of Electronic-Key memory in the DB (EKS_Read_Write) is NOT in the array; instead, it must be listed as an extra byte.

10.3. Writing contents of the Electronic-Key memory using a watch and force table

The same watch and force table has been prepared such that data can also be written to the Electronic-Key. For this purpose the start address and the number of bytes must be defined (cf. Chapter 4.2). In this example the first 8 bytes of the Electronic-Key memory are written, as for reading. Complete the data and click *Modify now* (shortcut: Shift+F9). To write an Electronic-Key it is then necessary to set the bit "...Write_Electronic-Key" and to reset it after successful completion of the write process.

	Name	Address	Display format	Monitor value	Modify value		Comment
1	// Write EKS Key						
2	"EKS_Read_Write".WriteData."Write_Electronic-Key"		Bool	<input type="checkbox"/> FALSE	FALSE	<input checked="" type="checkbox"/>	
3	"EKS_Read_Write".WriteData.Start_address		Hex	16#00	16#00	<input checked="" type="checkbox"/>	
4	"EKS_Read_Write".WriteData.Number_of_bytes		Hex	16#08	16#08	<input checked="" type="checkbox"/>	
5	"EKS_Read_Write".WriteData.Transmit_data[0]		Character	'E'	'E'	<input checked="" type="checkbox"/>	
6	"EKS_Read_Write".WriteData.Transmit_data[1]		Character	'K'	'K'	<input checked="" type="checkbox"/>	
7	"EKS_Read_Write".WriteData.Transmit_data[2]		Character	'S'	'S'	<input checked="" type="checkbox"/>	
8	"EKS_Read_Write".WriteData.Transmit_data[3]		Character	'T'	'T'	<input checked="" type="checkbox"/>	
9	"EKS_Read_Write".WriteData.Transmit_data[4]		Character	'E'	'E'	<input checked="" type="checkbox"/>	
10	"EKS_Read_Write".WriteData.Transmit_data[5]		Character	'S'	'S'	<input checked="" type="checkbox"/>	
11	"EKS_Read_Write".WriteData.Transmit_data[6]		Character	'T'	'T'	<input checked="" type="checkbox"/>	
12	"EKS_Read_Write".WriteData.Transmit_data[7]		Character	'\$00'	'\$00'	<input checked="" type="checkbox"/>	

Figure 18: Writing Electronic-Key data example

	Name	Address	Display format	Monitor value	Modify value		Comment
1	// Write EKS Key						
2	"EKS_Read_Write".WriteData."Write_Electronic-Key"		Bool	<input type="checkbox"/> FALSE	FALSE	<input checked="" type="checkbox"/>	
3	"EKS_Read_Write".WriteData.Start_address		Hex	16#00	16#00	<input checked="" type="checkbox"/>	
4	"EKS_Read_Write".WriteData.Number_of_bytes		Hex	16#08	16#08	<input checked="" type="checkbox"/>	
5	"EKS_Read_Write".WriteData.Transmit_data[0]		Character	'E'	'E'	<input checked="" type="checkbox"/>	
6	"EKS_Read_Write".WriteData.Transmit_data[1]		Character	'K'	'K'	<input checked="" type="checkbox"/>	
7	"EKS_Read_Write".WriteData.Transmit_data[2]		Character	'S'	'S'	<input checked="" type="checkbox"/>	
8	"EKS_Read_Write".WriteData.Transmit_data[3]		Character	'T'	'T'	<input checked="" type="checkbox"/>	
9	"EKS_Read_Write".WriteData.Transmit_data[4]		Character	'E'	'E'	<input checked="" type="checkbox"/>	
10	"EKS_Read_Write".WriteData.Transmit_data[5]		Character	'S'	'S'	<input checked="" type="checkbox"/>	
11	"EKS_Read_Write".WriteData.Transmit_data[6]		Character	'T'	'T'	<input checked="" type="checkbox"/>	
12	"EKS_Read_Write".WriteData.Transmit_data[7]		Character	'\$00'	'\$00'	<input checked="" type="checkbox"/>	
13						<input type="checkbox"/>	
14	// Read EKS Key						
15	"EKS_Read_Write".ReadData.Device_ready_for_operation		Bool	<input checked="" type="checkbox"/> TRUE		<input type="checkbox"/>	
16	"EKS_Read_Write".ReadData."Electronic-Key_detected"		Bool	<input checked="" type="checkbox"/> TRUE		<input type="checkbox"/>	
17	"EKS_Read_Write".ReadData.Job_finished		Bool	<input type="checkbox"/> FALSE		<input type="checkbox"/>	
18	"EKS_Read_Write".ReadData.Job_in_progress		Bool	<input type="checkbox"/> FALSE		<input type="checkbox"/>	
19	"EKS_Read_Write".ReadData."Receive_data[0]"		Character	'E'		<input type="checkbox"/>	
20	"EKS_Read_Write".ReadData.Receive_data[1]		Character	'K'		<input type="checkbox"/>	
21	"EKS_Read_Write".ReadData.Receive_data[2]		Character	'S'		<input type="checkbox"/>	
22	"EKS_Read_Write".ReadData.Receive_data[3]		Character	'T'		<input type="checkbox"/>	
23	"EKS_Read_Write".ReadData.Receive_data[4]		Character	'E'		<input type="checkbox"/>	
24	"EKS_Read_Write".ReadData.Receive_data[5]		Character	'S'		<input type="checkbox"/>	
25	"EKS_Read_Write".ReadData.Receive_data[6]		Character	'T'		<input type="checkbox"/>	
26	"EKS_Read_Write".ReadData.Receive_data[7]		Character	'\$00'		<input type="checkbox"/>	
27		<Add new>				<input type="checkbox"/>	

Figure 19: Writing the Electronic-Key memory

11. Important note – please observe carefully!

This document is intended for a design engineer who possesses the requisite knowledge in safety engineering and knows the applicable standards, e.g. through training for qualification as a safety engineer. Only with the appropriate qualification is it possible to integrate the example provided into a complete safety chain.

The example represents only part of a complete safety chain and does not fulfill any safety function on its own. In order to fulfill a safety function, the energy switch-off function for the danger zone and the software must also be considered in the safety evaluation, for example.

The applications provided are only examples for solving certain safety tasks for protecting safety doors. The examples cannot be comprehensive due to the application-dependent and individual protection goals within a machine/installation.

If questions concerning this example remain open, please contact us directly.

According to the Machinery Directive 2006/42/EC, the design engineer of a machine or installation has the obligation to perform a risk assessment and take measures to reduce the risk. While doing this, the engineer must comply with the applicable national and international safety standards. Standards generally represent the current state-of-the-art. Therefore, the design engineer should continuously inform himself about changes in the standards and adapt his considerations to them. Relevant standards for functional safety include EN ISO 13849 and EN 62061. This application must be regarded only as assistance for the considerations about safety measures.

The design engineer of a machine/installation has the obligation to assess the safety technology himself. The examples must not be used for an assessment, because only a small excerpt of a complete safety function was considered in terms of safety engineering here.

In order to be able to use the safety switch applications correctly on safety doors, it is indispensable to observe the standards EN ISO 13849-1, EN ISO 14119 and all relevant C-standards for the respective machine type. Under no circumstances does this document replace the engineer's own risk assessment, and it cannot serve as the basis for a fault assessment.

In particular in relation to a fault exclusion, it must be noted that a fault can be excluded only by the machine's or installation's design engineer and this action requires justification. A general fault exclusion is not possible. More information about fault exclusion can be found in EN ISO 13849-2.

Changes to products or within assemblies from third-party suppliers used in this example can lead to the function no longer being ensured or the safety assessment having to be adapted. In any event, the information in the operating instructions on the part of EUCHNER, as well as on the part of third-party suppliers, must be used as the basis before this application is integrated into an overall safety function. If contradictions should arise between the operating instructions and this document, please contact us directly.

Use of brand names and company names

All brand names and company names stated are the property of the related manufacturer. They are used only for the clear identification of compatible peripheral devices and operating environments in relation to our products.

Euchner GmbH + Co. KG
Kohlhammerstraße 16
70771 Leinfelden-Echterdingen
info@euchner.de
www.euchner.com

Edition:
AP000238-02-01/20
Title:
Application EKS
Integration of EKS with PROFINET IO interface into TIA Portal
from V14 SP1

Copyright:
© EUCHNER GmbH + Co. KG, 01/2020

Subject to technical modifications; no responsibility is accepted for the accuracy of this information.