Documentation

Profinet connection
S7-1200/1500 ↔ SE-7xx

variable data mapping

via Hilscher NetJack 100 (IO-Device)

Documentation: 11th May 2021

Author: Lukas Jolbej

In this documentation the S7 Profinet connection (variable data mapping) to the Stange SE-7xx device with Hilscher NetJack 100 module is explained.

Used devices:

- Stange SE-702
- Siemens S7-1212C DC/DC/DC (6ES7212-1AE40-0XB0) as IO Controller
- Hilscher NetJack 100 (NJ 100DN-RE/PNS) as IO Device

connected via a 100 MBit/s switch

Used software:

- Siemens TIA Portal V15
- Windows 7 SP1
- Device version 7.0.2.10 for the SE-702
- Firmware version 4.1 for the S7-1212C
- Firmware version 3.12.0.2 for the Hilscher NetJack 100

Requirements:

- The feature must be licensed in the SE-7xx
- The Profinet name of the SE-7xx (Hilscher NetJack 100) and the IP addesses for both networking interfaces must be known
- SE-7xx (Hilscher NetJack 100) and S7 are located in the same network subnet

Please note that, depending on the S7 firmware version, a current version of TIA Portal may be required.

Inhaltsverzeichnis

GETTING STARTED 4

FUNCTION OVERVIEW 4
Functionality 4
What data can be transferred from the S7 to the SE-7xx (input mapping at the SE-7xx)? 4
What data can be transferred from the SE-7xx to the S7 (output mapping at the SE-7xx)? 4
CHECKING THE LICENSING STATUS 5
ACTIVATING THE S7 PROFINET INTERFACE IN THE SE-7XX 5
LIMITATIONS FOR CONFIGURING THE PROFINET DEVICE NAME 5
DATALOGGER CONFIGURATION (PLC STATEMENT LIST) 5

STRUCTURE OF THE EXAMPLE PROJECT 6

CONFIGURING THE HILSCHER NETJACK 100 IN TIA PORTAL 6
SETTING THE IP ADDRESSES IN TIA PORTAL 8
SETTING UP THE DATA MAPPING IN THE SE-7XX 9
MODULE CONFIGURATION IN TIA PORTAL 10
CREATING THE S7 DATA BLOCKS (DB) FOR STORING THE RECEIVE DATA/TRANSMIT DATA 11
CREATING THE S7 FUNCTION (FC) FOR DATA TRANSMISSION 12

Getting started

Function overview

Functionality

The S7 Profinet interface of the SE-7xx series via a Hilscher NetJack 100 is a licensable extension to the two Modbus interfaces at port 502 and 21303 (➔ S7 Modbus interface). These interfaces allow read and write access to data in the SE-7xx via Modbus TCP or Profinet.

The S7 Profinet interface offers two operating modes: fixed data mapping as well as variable data mapping.
In this documentation the variable mapping is described. For this variant there is no template available.
The entire configuration as well as the data exchange in both directions must therefore be set up manually.

With the help of the variable mapping, it is possible to define exactly which data should be read or written. The advantage is that the scope of the data to be read or written can be adapted to the project. This also saves Profinet modules on the S7, which can be used for other hardware instead.

Basically, the behavior of the SE-7xx changes by switching on the S7 Profinet interface in such a way that the digital control signals of the S7 have priority over corresponding entries in the PLC instruction list of the SE-7xx. Therefore, the entire digital logic may be implemented in the S7.
However, only the data areas that are specified in the mapping configuration are written or overwritten to the SE-7xx. If, for example, only actual values are written, the entire PLC configuration can remain in the SE-7xx.
For the actual values, please note that values of the CAN IO (SIOS or CAN base) always have priority.

The S7 Profinet interface (variant with fixed data mapping) is not compatible with the variant with variable mapping.
These are two different ways to exchange data via Profinet.
Only the variant with variable mapping supports the enlarged quantity structure (from device version 7.0.3.x).

What data can be transferred from the S7 to the SE-7xx (input mapping at the SE-7xx)?

- Boolean data ➔ Digital inputs of CAN peripherals (I), Digital outputs of CAN peripherals (O), Function inputs (FI), Function outputs (FO)

- 32 Bit floating-point values (REAL) ➔ Setvalues, Actual values, Analog variables

What data can be transferred from the SE-7xx to the S7 (output mapping at the SE-7xx)?

- Boolean data ➔ Digital inputs of CAN peripherals (I), Digital outputs of CAN peripherals (O), Function inputs (FI), Function outputs (FO)

- 32 Bit floating-point values (REAL) ➔ Setvalues, Actual values, Formula values, Analog variables, Control zone outputs (Y values), Y values heating, Y values cooling, Control zone actual values, Control zone setvalues, Lower Control zone limits, Upper Control zone limits, Control zone plus-tolerances, Control zone minus-tolerances, Limits, Plus-tolerances, Minus-tolerances, Target setvalues

Checking the licensing status

The correct licensing status of the S7 Profinet connection can be checked in the SE-7xx.
Configuration > ***Hardware Test*** > ***License Information*** > ***Profinet IO-Device*** will show the current status.
In case of a missing license this entry shows “No” and a license alarm will be activated.

Activating the S7 Profinet interface in the SE-7xx

The S7 Profinet interface must be enabled in the SE-7xx device. This takes place under ***Configuration*** > ***Hardware*** > ***Hardware options***. Change the setting ***Hardware module*** to ***Profinet IO-Device***. Now the submenu ***Profinet IO-Device*** can be opened. There you can select ***PARAM.*** on the left side and configure the ***Profinet name***, by default ***nj100repns***. The setting ***Data mapping*** must be set to ***Variable data mapping***.

The “fixed data mapping” (Predefined data blocks) is another operation mode of the S7 Profinet interface and is described in a separate documentation.

Only a 1:1 data transfer is possible. This means that an SE-7xx cannot be connected to several S7s and exchange data. Likewise, an S7 can only exchange data with one SE-7xx.[1]

Limitations for configuring the Profinet Device Name

Please consider the rules of the Profinet naming convention. The following rules must be respected when configuring the Profinet device name so that the communication can take place: (Source: https://support.industry.siemens.com)

- Limitation to a total length of 127 characters (letters “a” to “z”, digits “0” to “9”, hyphen or dot)
- A name component within the device name, i.e., a string between two dots, must not be longer than 63 characters
- No special characters like umlauts, parantheses, underscores, slashes, blank spaces
  The hyphen is the only allowed special character
- The device name may not contain any capital letters
- The device name may not start or end with the characters “-“ or “.”
- The device name may not start with digits
- The device name may not have the format n.n.n.n (n = 0…999)
- The device name may not start with the string “port-xyz-“ (x, y, z = 0…9)

Datalogger configuration (PLC statement list)

For proper functionality of the datalogger when enabling the S7 Profinet interface, the following two lines must be present in the SE-7xx PLC statement list:

```
L FO 768
R FO 1311
```

The PLC statement list can be found at ***Configuration*** > ***Functions*** > ***PLC statement list***. After adding those two lines apply the changes by selecting ***Apply*** (***Take Over***) and then save the changes with ***Back***.

These two lines are only necessary if the data logger is to be used. Since the function inputs are not necessarily written to during variable mapping, the configuration of the data logger may remain in the SE-7xx.

For more information on configuring the datalogger see the corresponding documentation.

[1] It would be technically possible to control a second SE-7xx. However, this is not supported by Stange and is therefore not discussed in detail here. For example, it should also be noted that the peripheral address range for the modules is limited for the S7-1200.

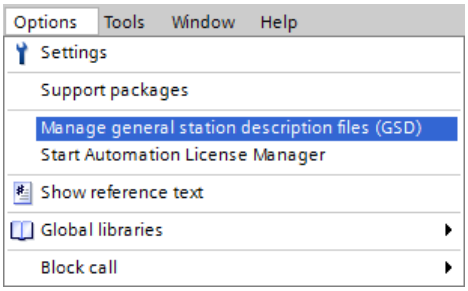
Structure of the example project

This documentation describes the integration of the Profinet connection to the SE-7xx in an empty project. Of course, it is also possible to extend an existing project with the Profinet connection. In this case, however, you must make sure that the address range of the S7 is sufficient for the required Profinet modules.

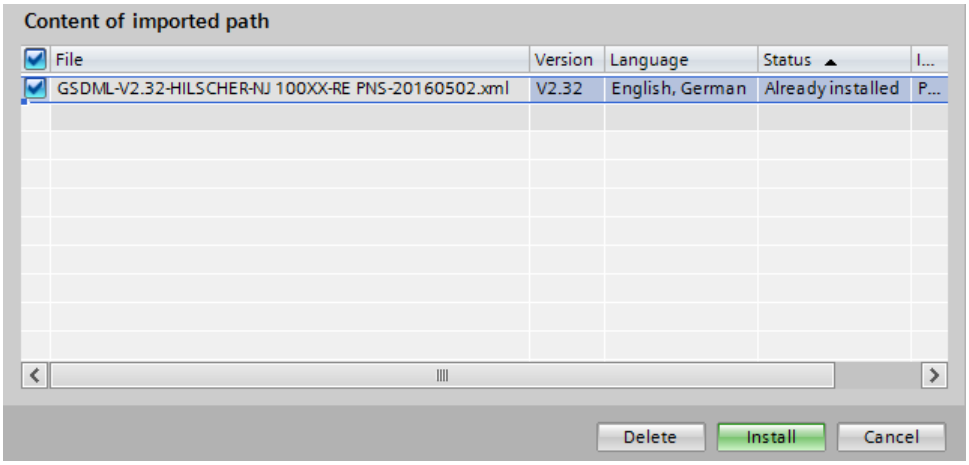
First create a new TIA Portal project and select the S7 under ***Add new device***. This is then displayed in the project navigation.

Configuring the Hilscher NetJack 100 in TIA Portal

So that the Hilscher NetJack 100 can be integrated into the project, the associated GSDML file must be imported. This is done via ***Options*** and ***Manage general station description files (GSD)***:

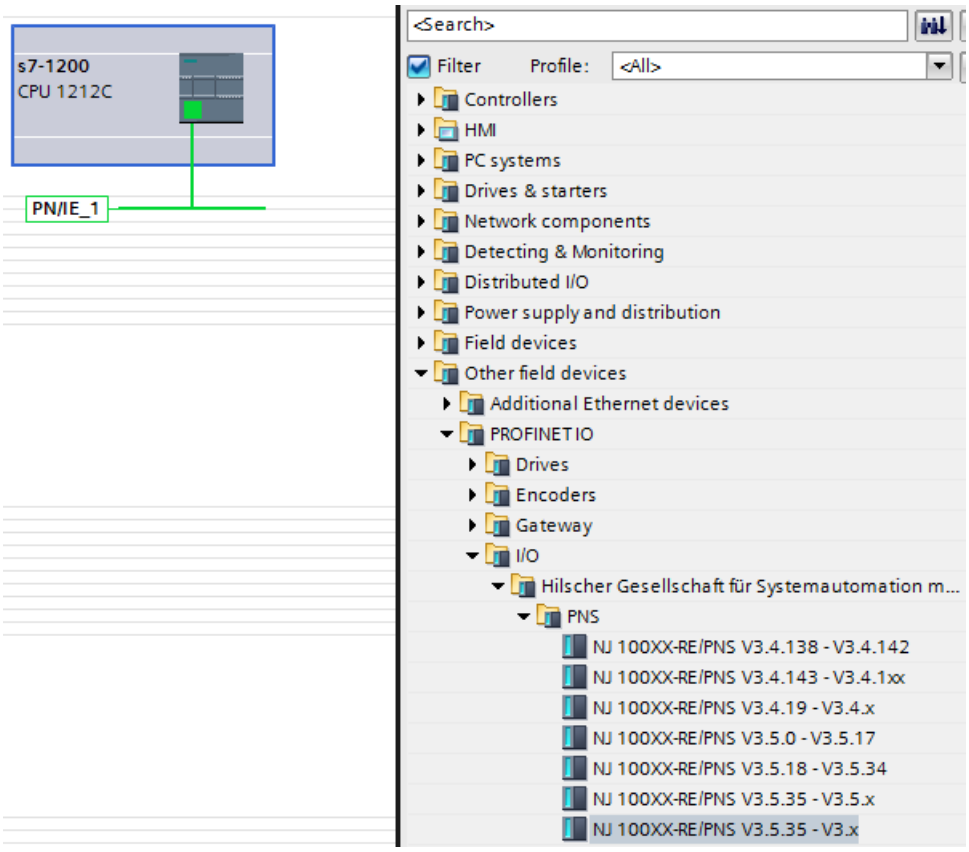


In the following window the location of the GSDML file is specified and the GSDML file is installed (the GSDML file is located in the S7 template package):



After the installation the window can be closed. The Hilscher NetJack 100 is now available in the hardware catalog and can be dragged to the free space under ***Devices & Networks***.

Please use the entry **NJ 100XX-RE/PNS V3.5.35 – V3.x**.



Via drag & drop the green boxes of the S7 and the Hilscher NetJack 100 (in the network view) are connected, alternatively the subnet can be selected in the properties of the Hilscher NetJack 100. The IP address can also be set there. By connecting S7 and Hilscher NetJack 100, the S7 is automatically configured as IO controller.

The most common cause of error is a Profinet name configured in the SE-7xx that does not match the configuration in TIA Portal. By default, the name ***nj100repns*** is set. This must then be changed in the properties, if necessary, so that the Profinet names in the SE-7xx and in the project are the same.

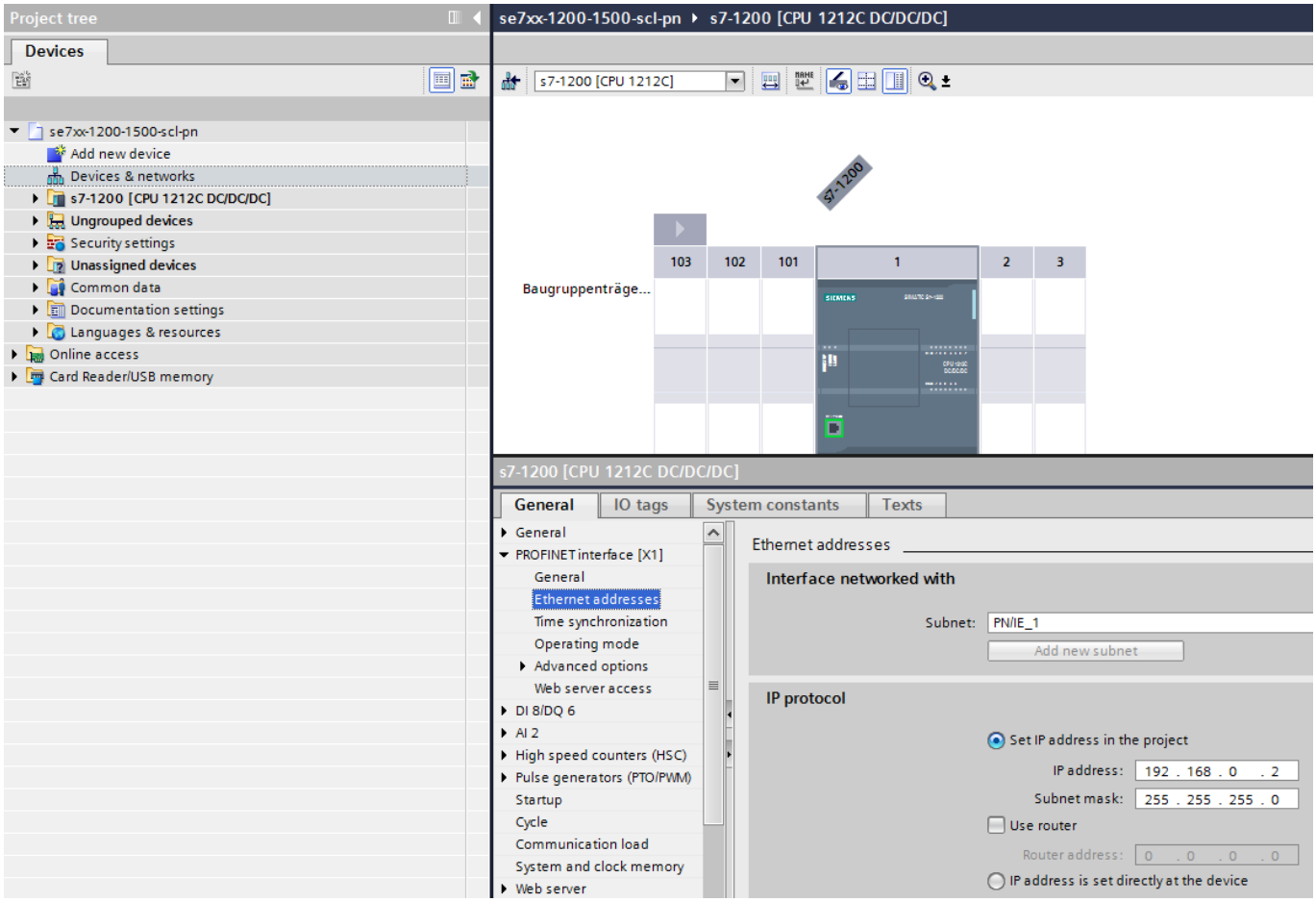
Setting the IP addresses in TIA Portal

To change the IP address of the S7, double-click on the S7 under ***Devices & networks*** and double-click again in the device view. Under the entry ***PROFINET interface*** the IP address, subnet mask and, if necessary, router address can be set.

Analog to this, a separate IP address must be set in the TIA Portal for the SE-7xx (Hilscher NetJack 100).
Please note that the IP address configured here does not correspond to the IP address configured in the SE-7xx. These are two different network interfaces, which therefore require two separate IP addresses.
Please set the IP address for Profinet only in the project - analog to the IP address of the S7; in particular, do not use the "Assign IP address" function under "Online & diagnostics" for the SE-7xx (Hilscher NetJack 100).

The SE-7xx (Hilscher NetJack 100) and the S7 must be in the same subnet (subnet mask), otherwise communication errors may occur.

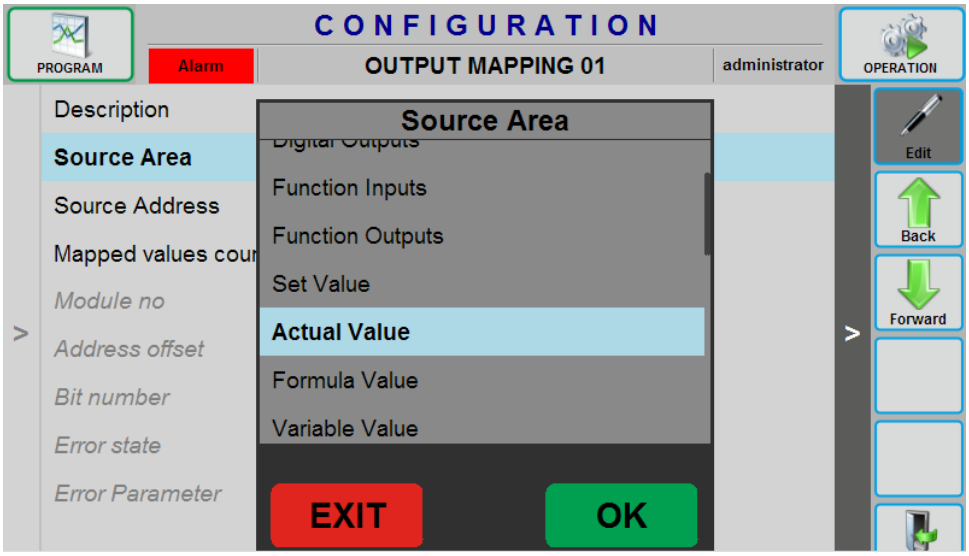
Both ports of the Hilscher NetJack 100 are equivalent; only one of them has to be connected to a Profinet compatible switch. The Hilscher NetJack 100 is only responsible for the Profinet functionality. For all other network functions of the SE-7xx, it must be connected to the switch via a separate cable (via the Ethernet socket directly on the device).



Setting up the data mapping in the SE-7xx

In the SE-7xx, up to 16 modules of 64 bytes each – in both data directions – can be configured. Under ***Configuration*** > ***Hardware*** > ***Hardware options*** > ***Profinet IO-Device*** the desired mapping can be set in the subitems ***Input mapping*** or ***Output mapping***.

As an example, the actual values 1-4, the setvalues 1-2 and the programmer status are to be transferred to the S7. Since this is output data (as seen from the SE-7xx), this configuration is done under ***Output mapping***. There, the right button ***Add*** can be selected to add a new mapping line, and ***Edit*** opens the definition of this mapping line. There a ***Description*** can be given (e.g., "Actual values 1-4") as well as under ***Source Area*** the desired area ***Actual Value***:



The ***Source Address*** specifies the first instance of the actual values to be mapped (here this would be ***1***) and the ***Mapped values count*** then corresponds to ***4***. This would complete the configuration of this mapping. The specification of the module number as well as the address offset within this module are calculated automatically and are important for mapping and accessing the data in a data block (DB) on the S7.

The configuration of the setvalue mapping follows the same scheme. To map the programmer status bits, you must know their exact location in the function outputs (FO) area. A list of all function inputs (FI) and function outputs (FO) can be found at the end of the SE-7xx operating manual.

In that listing you can see that the programmer status is mapped in FO 681-688. With this information the mapping entry for the programmer status can be configured. The source address corresponds to the first value of the mapped values (here: ***681***) and the number corresponds to the number of the individual transferred values; here this would be ***8***, since eight function outputs are to be transferred with this mapping entry. Now you can go ***Back*** again.

Important: If there are entries in the mapping overview whose column value ***Mod./Byte*** is specified with ***??/??***, this means that not enough modules have been configured for the specified mappings. To do this, go ***Back*** once and then ***PARAM.*** can be selected using the left tab. Here the number of output modules must be determined and adjusted manually. In the above example we have four actual values each 4 bytes, two set values each 4 bytes as well as 8 bits (equal to 1 byte), so a total of 25 bytes. Only 64-byte modules may be configured on the S7. Since only one 64-byte module is required for 25 bytes, a value of ***1*** can be entered for number of output modules.
For every 64 bytes that are newly started, another module is required - for 65 bytes, for example, a total of two modules.

Now the value should be displayed under ***Mod./Byte*** for all output mappings. The specification 01/24 means then that this mapping is in the first 64-byte module at module offset byte 24. If for example other bits should be inserted behind the programmer status bits, then the exact start bit within the module can be determined in the mapping configuration.

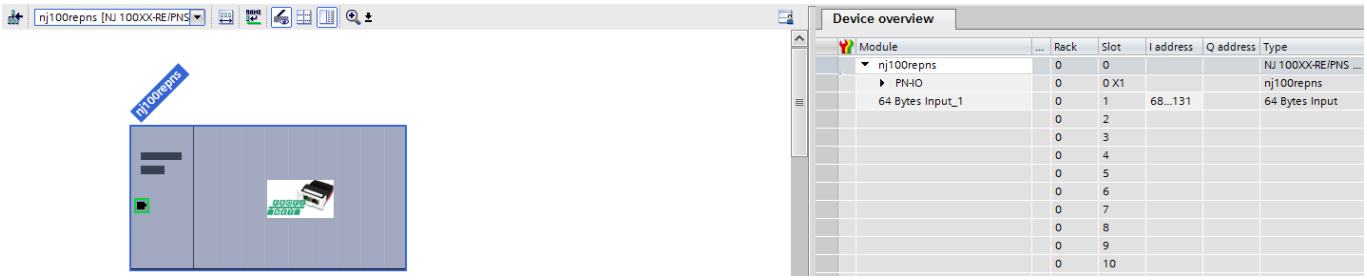
The configuration of the input mappings follows the same scheme. Reserve bytes can be inserted to align the data, in order to compensate for a possible shift of the entries in the S7-DB. Floats are inserted in the SE-7xx byte-aligned; however, this does not happen across module boundaries.

Module configuration in TIA Portal

For the data exchange between the SE-7xx (Hilscher NetJack 100) and the S7, Profinet modules must be configured. For this, exactly as many input modules with 64 bytes each and output modules with 64 bytes each are required as are specified in the mapping configuration of the SE-7xx. In the previous example, only one output module is required, which corresponds to one input module on the S7. The number of input modules in the SE-7xx (output modules on the S7) must then be calculated according to the same scheme based on the values to be mapped.

The modules are simply dragged from the hardware catalog into the project in the ***Device view***. Only 64-byte modules may be used. When assigning addresses, it must be ensured that the input and output modules are each addressed in one piece and that there are no address gaps between the modules. However, the start address of the input modules is freely selectable and independent of the start address of the output modules, which is also freely selectable.

If no data is transferred in one direction (e.g., from S7 to SE-7xx), the corresponding number of modules in the SE-7xx can be set to 0. In this case, no corresponding modules need to be configured in the S7.



After the module configuration has been completed, the basic Profinet communication between SE-7xx and S7 can be tested. For this purpose, the configuration screen is exited on the SE-7xx by selecting ***OPERATION*** and the S7 is compiled and loaded.

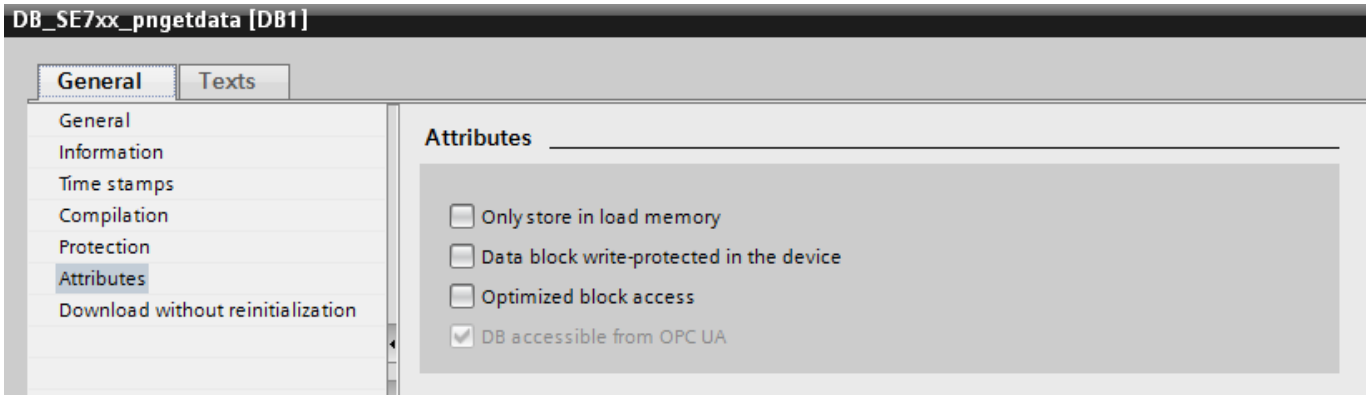
In the project tree in TIA Portal only green boxes with checkmarks should be visible now (in online view). At the Hilscher NetJack 100 the LEDs SF and BF should go off, only SYS should be permanently green.
If this is the case, the basic Profinet communication between both devices is established and you can continue with the next step.

If errors occur, first check whether the configured Profinet name of the Hilscher NetJack 100 corresponds to the configured Profinet name in the SE-7xx, by default ***nj100repns***. If this is equal, check again the ***Limitations for configuring the Profinet Device Name*** on page **Fehler! Textmarke nicht definiert.** and change the name if needed. Even if the S7 seems to accept the name, these restrictions must always be observed.
Finally, make sure that the respective numbers of modules in the SE-7xx and in the S7 match.

Creating the S7 data blocks (DB) for storing the receive data/transmit data

As soon as the Profinet communication is established, the project can be continued in TIA Portal. The next step is to create a DB that stores the input data from the SE-7xx. For this purpose, a new global DB is created under ***Program blocks*** and ***Add new block***. The name is set to ***DB_SE7xx_pngetdata*** in this example – this way it is directly recognizable that the data received from the SE-7xx via Profinet is stored in this DB.

After creating the DB, the item ***Optimized block access*** must first be switched off in its properties under ***Attributes***. Only now can the individual entries be addressed via their byte addresses.



Now the individual mapping lines can be mapped in the S7 DB and the DB can be compiled:

DB_SE7xx_pngetdata

	Name	Data type	Offset
1	Static		
2	ActualValues_1-4	Array[1..4] of Real	0.0
3	SetValues_1-2	Array[1..2] of Real	16.0
4	Programmer_Status	Array[681..688] of Bool	24.0
5	<Add new>		

Here it is essential to ensure that the specified offsets correspond to the offsets calculated in the SE-7xx. For example, a DB offset of 83.0 would correspond to module 2, byte offset 19 in the SE-7xx.
In general: DB offset = (module number-1) × 64 + module byte offset.

In this example, arrays were used in the S7-DB to prevent possible filling bytes that could occur in a Struct. If individual mappings do not match offset-wise, reserve bytes can be inserted in the SE-7xx; it may (also) be necessary to insert reserve bytes in the S7-DB. It is therefore recommended to map floats before bools.

For the data from the S7 to the SE-7xx (output data of the S7, input data of the SE-7xx) a new DB can be created. This could be called, e.g., ***DB_SE7xx_pnsetdata*** and accordingly contain the mapping lines of the opposite direction.

Creating the S7 function (FC) for data transmission

To fill the DB *DB_SE7xx_pngetdata* with data, the FC *_datatransfer* is now created for the data transfer. This FC is then called in *Main* [OB1]. In this example SCL is used as the block language.

For reading the data the S7 function *POKE_BLK* is used. This is called in *_datatransfer* to write the receive data from the Profinet modules into the receive DB:

```
POKE_BLK(area_src:=_byte_in_,
         dbNumber_src:=_dint_in_,
         byteOffset_src:=_dint_in_,
         area_dest:=_byte_in_,
         dbNumber_dest:=_dint_in_,
         byteOffset_dest:=_dint_in_,
         count:=_dint_in_);
```

Since it can now be assumed that the mapping of the output data of the SE-7xx matches the DB offsets in the S7, only one call of this function is required to write all data to the DB. This would also be true if more than one module is needed, since the data is read across modules.

The first three parameters of *POKE_BLK* specify the source of the data. Since the data is located in the Profinet input modules of the S7, the input type *16#81* for "Inputs" is selected for the parameter *area_src* (the valid values can also be found in the F1 help). Because there is no source DB, *dbNumber_src* is *0* here. *byteOffset_src* specifies the first byte in the selected input type from which to read. This is to be looked up in the module configuration of the Hilscher NetJack 100. The start value of the first input module corresponds to the byte offset searched for and must be entered here, in this example *68*. This means that the incoming data are mapped from IB68.0.

The next three parameters specify the destination of the data. These are to be written into the receive DB. *area_dest* is then *16#84*, *dbNumber_dest* corresponds to the DB number - here *1*. *byteOffset_dest* specifies the byte offset in the destination. Since the destination DB is written from the beginning, this value is *0*.

The last parameter *count* finally specifies how many bytes are to be transferred. In this example, four actual values, two set-points and 8 bits (1 byte) for the programmer status are transferred. This makes a total of *25* bytes.

The complete function call to write the received Profinet data into the receive DB then looks like this:

```
POKE_BLK(area_src:=16#81,
         dbNumber_src:=0,
         byteOffset_src:=68,
         area_dest:=16#84,
         dbNumber_dest:=1,
         byteOffset_dest:=0,
         count:=25);
```

The blocks can now be compiled and loaded to the S7. The received data are now written cyclically to the data block *DB_SE7xx_pngetdata* and can be used further from there.

Sending data from the S7 to the SE-7xx is done very similarly. Only another call of *POKE_BLK* must be added and the parameters must be set accordingly.
The source of the data would then be *DB_SE7xx_pnsetdata* and the destination the outputs of the S7 (*16#82*). Accordingly, the DB number of *DB_SE7xx_pnsetdata* is to be entered as source DB and the value *0* as destination DB.