



ibaPDA-Request-S7-UDP

Request Data Interface to SIMATIC S7 via UDP

Manual
Issue 2.1

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	www.iba-ag.com

Unless explicitly stated to the contrary, it is not permitted to pass on or copy this document, nor to make use of its contents or disclose its contents. Infringements are liable for compensation.

© iba AG 2025, All rights reserved.

The content of this publication has been checked for compliance with the described hardware and software. Nevertheless, discrepancies cannot be ruled out, and we do not provide guarantee for complete conformity. However, the information furnished in this publication is updated regularly. Required corrections are contained in the following regulations or can be downloaded on the Internet.

The current version is available for download on our web site www.iba-ag.com.

Version	Date	Revision	Author	Version SW
2.1	12-2025	iba block family ibaREQ3sym added	st, mm	8.11.0

Windows® is a brand and registered trademark of Microsoft Corporation. Other product and company names mentioned in this manual can be labels or registered trademarks of the corresponding owners.

Contents

1	About this documentation	6
1.1	Target group and previous knowledge	6
1.2	Notations	7
1.3	Used symbols.....	8
2	System requirements	9
3	ibaPDA-Request-S7-UDP.....	12
3.1	How much data can be transferred?	13
3.2	Configuration and engineering SIMATIC S7	13
3.2.1	Configuration and engineering SIMATIC S7-300, S7-400 and WinAC	13
3.2.1.1	Configuration in STEP 7.....	14
3.2.2	Configuration and engineering SIMATIC S7-1500	22
3.2.2.1	Configuration in STEP 7 with the iba block family ibaREQ.....	22
3.2.2.2	Configuration in STEP 7 using the ibaREQsym iba block family	24
3.2.2.3	Configuration in STEP 7 using the ibaREQ3sym iba block family	25
3.3	Configuration and engineering ibaPDA.....	27
3.3.1	General interface settings.....	27
3.3.2	Adding a module.....	28
3.3.3	General module settings.....	29
3.3.4	Connection settings	30
3.3.4.1	Connection mode TCP/IP	31
3.3.4.2	Connection mode PC/CP	33
3.3.4.3	Connection mode TCP/IP S7-1x00	36
3.3.5	Signal configuration	39
3.3.5.1	Selection via the absolute address of the operands.....	40
3.3.5.2	Selection via the symbolic operand addresses	41
3.3.5.3	Selection of CFC connectors	44
3.3.6	Module S7 Request.....	45
3.3.7	Module S7 UDP Request Decoder	46
3.3.8	Module diagnostics.....	49
3.3.9	Address books	51
3.3.9.1	Creating address books offline from S7 project.....	52
3.3.9.2	Creating address books online from S7-1200/1500 CPU	54

4	Description of Request blocks	55
4.1	ibaREQ iba block family	55
4.1.1	Device configuration	57
4.1.2	ibaREQ_M (FB140).....	59
4.1.3	ibaREQ_M (FB1400).....	62
4.1.4	ibaREQ_UDPact (FB145)	64
4.1.5	ibaREQ_UDPact (FB1410)	66
4.1.6	ibaREQ_UDPint (FB146).....	68
4.1.7	ibaREQ_UDPext3 (FB147)	73
4.1.8	ibaREQ_UDPext4 (FB148)	74
4.1.9	ibaREQ_UDP2 (FB1406)	75
4.2	ibaREQsym iba block family	79
4.2.1	ibaREQsym_M	80
4.2.2	ibaREQsym_UDP	81
4.3	ibaREQ3sym iba block family	88
4.3.1	ibaREQ3sym_M	89
4.3.2	ibaREQ3sym_UDP	90
5	Diagnostics.....	97
5.1	License	97
5.2	Visibility of the interface.....	97
5.3	Log files.....	98
5.4	Connection diagnostics with PING.....	99
5.5	Connection diagnostics with PG/PC interface	100
5.6	Connection table	101
5.7	Diagnostic modules	102
6	Appendix	107
6.1	iba S7 library	107
6.1.1	iba S7 library for SIMATIC Manager	107
6.1.1.1	Integrating the library into SIMATIC Manager	108
6.1.1.2	Using the blocks in SIMATIC Manager	109

6.1.2	iba S7 library for SIMATIC TIA portal.....	110
6.1.2.1	Integrating the library into TIA Portal	111
6.1.2.2	Using the blocks in TIA Portal	112
6.2	Application examples.....	113
6.3	S7 cycle time measurements	114
6.4	Adaptation to the renumbered system functions.....	115
6.5	Setting PG/PC interface/defining new access point.....	117
6.6	S7 routing	120
6.6.1	Routing from Ethernet to Ethernet	120
6.6.1.1	Configuration of STEP 7/NetPro	121
6.6.1.2	Configuration of TIA Portal	123
6.6.1.3	Configuration of ibaPDA	124
6.6.2	Routing from Ethernet to PROFIBUS	126
6.6.2.1	Configuration of STEP 7/NetPro	126
6.6.2.2	Configuration of TIA Portal	128
6.6.2.3	Configuration of ibaPDA	129
7	Support and contact.....	131

1 About this documentation

This documentation describes the use of the Request data interface to SIMATIC S7 via UDP.

The product *ibaPDA-Request-S7-UDP* is an extension of *ibaPDA* that allows selective to S7 symbols and S7 operands during data acquisition from SIMATIC S7 CPUs. Data is transmitted using the UDP network protocol (User Data Protocol). This documentation only describes the extensions and deviations. For all other functions and operating instructions, please refer to the documentation for *ibaPDA* and *ibaPDA-Interface-S7-TCP/UDP*.

Other documentation



This documentation provides supplementary information to the *ibaPDA* and *ibaPDA-Interface-S7-TCP/UDP* documentations.

1.1 Target group and previous knowledge

This documentation is aimed at qualified professionals who are familiar with handling electrical and electronic modules as well as communication and measurement technology. A person is regarded as professional if he/she is capable of assessing safety and recognizing possible consequences and risks on the basis of his/her specialist training, knowledge and experience and knowledge of the standard regulations.

This documentation in particular addresses persons who are concerned with the configuration, test, commissioning or maintenance of Programmable Logic Controllers of the supported products. For the handling *ibaPDA-Request-S7-UDP* the following basic knowledge is required and/or useful:

- Windows operating system
- Basic knowledge of *ibaPDA*
- Knowledge of configuration and operation of the relevant control system

1.2 Notations

In this manual, the following notations are used:

Action	Notation
Menu command	Menu <i>Logic diagram</i>
Calling the menu command	<i>Step 1 – Step 2 – Step 3 – Step x</i> Example: Select the menu <i>Logic diagram – Add – New function block</i> .
Keys	<Key name> Example: <Alt>; <F1>
Press the keys simultaneously	<Key name> + <Key name> Example: <Alt> + <Ctrl>
Buttons	<Key name> Example: <OK>; <Cancel>
Filenames, paths	<i>Filename, Path</i> Example: <i>Test.docx</i>

1.3 Used symbols

If safety instructions or other notes are used in this manual, they mean:

Danger!



The non-observance of this safety information may result in an imminent risk of death or severe injury:

- Observe the specified measures.
-

Warning!



The non-observance of this safety information may result in a potential risk of death or severe injury!

- Observe the specified measures.
-

Caution!



The non-observance of this safety information may result in a potential risk of injury or material damage!

- Observe the specified measures
-

Note



A note specifies special requirements or actions to be observed.

Tip



Tip or example as a helpful note or insider tip to make the work a little bit easier.

Other documentation



Reference to additional documentation or further reading.

2 System requirements

The following system requirements apply for the use of the data interface *ibaPDA-Request-S7-UDP*:

- *ibaPDA* v8.11.0 or higher
- Basic license for *ibaPDA* + license for *ibaPDA-Interface-S7-TCP/UDP* + license for *ibaPDA-Request-S7-UDP*
- SIMATIC S7 controller S7-300, S7-400, S7-400H, S7-1500, WinAC, for access to optimized data blocks S7-1500 firmware V3 or later
- In case PC/CP connections are used:
 - SIMATIC STEP 7 or SIMATIC NET, or
 - SIMATIC TIA Portal
- SIMATIC CFC (beginning with V6.0), if signals are to be chosen by drag & drop¹⁾

For integrating the Request blocks in the S7 program:

- SIMATIC STEP 7 V5.4 SP5 or higher, or
- SIMATIC STEP 7 (TIA Portal) V16 or higher (function block libraries for older versions may be available on request), V18 or higher for access to optimized data blocks

For further requirements for the used computer hardware and the supported operating systems, refer to the *ibaPDA* documentation.

System restrictions

- Access to S7-1200 controllers is not supported.
- Connectors of CFC blocks, which have constant values assigned, have no operand address. They are marked as constant in the address book and cannot be selected as signal.
- If function blocks (FB) are used in CFC, the internal static variables of the FB are also displayed in the address book, because they are treated in exactly the same way as connectors by the compiler. These are to be ignored.
- For data acquisition, *ibaPDA* supports only the following data types:
 - BOOL, BYTE, WORD, DWORD, INT, DINT, REAL, TIME, CHARAll other data types exist in the address book but cannot be entered in the signal list.
- For functions (FC) with connections of the data type STRING, POINTER, STRUCT or ANY under CFC, the interpretation of the SCL code does not work, as there are no references to the data types available in the source.

¹⁾ can only be used with SIMATIC STEP 7 v5.x

ibaREQsym block family

- TIA Portal V18 or higher
- Firmware V3.0 or higher
- Supported items: E, A, M and items from data blocks, only elementary data types and individual items from structured data types
- Variables must have the attribute "Accessible from HMI/OPC UA/Web API" or "Writable from HMI/OPC UA/Web API".
- Max. length of symbolic variable name (fully qualified name including namespace): 254 UTF-16 characters
- Max. 10 Request instances can be simultaneously active per S7 CPU.
- Max. 2000 symbols can be simultaneously requested per S7 CPU.

Further information about the "ResolveSymbols" and "MoveResolvedSymbolsToBuffer" functions can be found in the Siemens documentation.

ibaREQ3sym block family

- TIA Portal V19 or higher
- Firmware V3.0 or higher
- Supported items: E, A, M and items from data blocks, only elementary data types and individual items from structured data types
- Variables must have the attribute "Accessible from HMI/OPC UA/Web API" or "Writable from HMI/OPC UA/Web API".
- Max. length of symbolic variable name (fully qualified name including namespace): 254 UTF-16 characters
- Max. 10 Request instances can be simultaneously active per S7 CPU.
- Max. 2000 symbols can be simultaneously requested per S7 CPU.

License information

Order no.	Name	Description
31.001040	ibaPDA-Interface-S7-TCP/UDP	Extension license for an <i>ibaPDA</i> system for one TCP/IP and UDP/IP interface with 64 connections
31.101040	one-step-up-Interface-S7-TCP/UDP	Extension license for 64 additional S7-TCP/UDP connections of an <i>ibaPDA-Interface-S7-TCP/UDP</i> interface (max. 3 extension licenses)
31.001311	ibaPDA-Request-S7-UDP	Extension license for an <i>ibaPDA</i> system for using Request-S7 with the <i>ibaPDA-Interface-S7-TCP/UDP</i> interface with 2 connections
31.101311	one-step-up-Request-S7-UDP	Extension license for 2 additional Request-S7-UDP connections of the <i>ibaPDA-Request-S7-UDP</i> interface (max. of 127 connections)

The use of *ibaPDA-Request-S7-UDP* requires the existence of an *ibaPDA-Interface-S7-TCP/UDP* license. For each active Request module, one S7-TCP/UDP connection and one Request-S7-UDP connection are occupied.

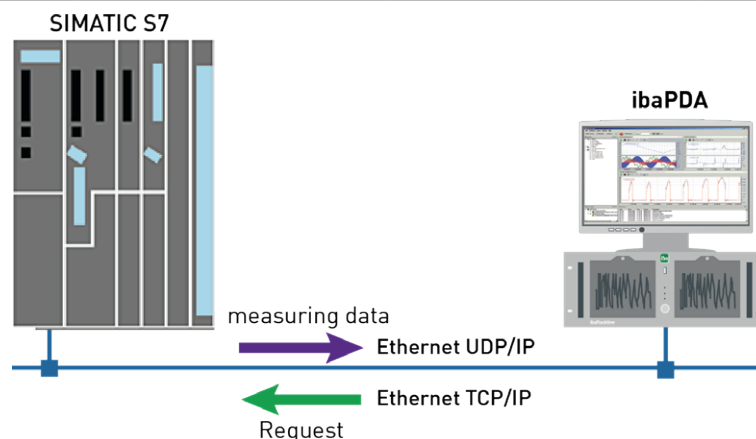
3 ibaPDA-Request-S7-UDP

The *ibaPDA-Request-S7-UDP* interface is suitable for the measurement data acquisition with free variable selection over standard network cards via UDP. The measurement data is sent actively from the controller to *ibaPDA*. For this purpose, several Request blocks have to be integrated in the S7 program for each connection. These Request blocks are used to cyclically send the current values of the S7 operands selected by the user within *ibaPDA* to be recorded in *ibaPDA*. When modifying the signal selection, no modification in the S7 program is required.

You can select the signals to be measured either with the absolute operand address or with the symbolic name with support of the *ibaPDA* address book browser. This browser allows to access to all defined symbols of the attached STEP 7 project.

You can use optimized data blocks for S7-1500 controllers. Signals within these data blocks can only be addressed via their symbolic name and not via the address or operand. For CPUs with firmware V3 or higher, it is also possible to access this data with special request blocks.

Block family	S7 CPU firmware	TIA Portal	Access to optimized data blocks
ibaREQ	unlimited	V16 or higher	no
ibaREQsym	V3 or higher	V18 or higher	yes



On side of the SIMATIC S7, you can use a PROFINET interface integrated on the CPU as well as an additional Ethernet capable communication processor (e.g. CP343-1, CPU343-1 LEAN, CP443-1, ...). The data volume that can be reached depends besides of many other factors significantly on the selected interface. Generally, interfaces integrated on the CPU are more powerful than communication processors, since the connection via the backplane bus represents a performance shortage. Especially for the modules of the S7-300 family, there are considerable shortages. For detailed information, please refer to the Siemens device and system manuals.

3.1 How much data can be transferred?

The amount of transferable data per module is limited by various parameters:

- **Maximum telegram size**

A maximum of 1466 bytes of user data can be transferred per telegram and thus per connection.

- **Maximum number of pointers**

The data to be transferred is defined using ANY pointers, which are transferred from *ibaPDA* to the controller. Here, connected operands (i.e. successive addresses) are represented by a common pointer. Depending on the size of the REQ_DB, a different number of pointers can be managed.

- With S7-300, S7-400, WinAC controls the length can be freely selected.

5280 bytes: up to 128 pointers

9120 bytes: up to 512 pointers

14240 bytes: up to 1024 pointers (maximum)

- Only a fixed length is provided for S7-1500 controllers:

9120 bytes: up to 512 pointers

Example

If you use a REQ_DB with a length of 9120 bytes, up to 512 pointers can be used, which may write a total address space of 1466 bytes.

If 512 distributed individual bytes of operands are to be acquired, all 512 pointers are required: Only 512 bytes can be acquired, although the maximum telegram size of 1466 bytes has not yet been reached.

If the 512 byte operands are located on successive addresses, only 1 pointer is required.

There are 511 pointers available to address the remaining 1466 bytes - 512 bytes = 954 bytes.

You can find the maximum telegram size and the current maximum number of pointers in the *S7 Request info* tab under *Diagnostics*.

See also [🔗 Module diagnostics](#), page 49.

3.2 Configuration and engineering SIMATIC S7

In the following, we describe the configuration and engineering on the SIMATIC S7 side.

You should distinguish whether this configuration is done with the SIMATIC Manager (STEP 7 Version ≤ V5) or with the SIMATIC TIA Portal.

3.2.1 Configuration and engineering SIMATIC S7-300, S7-400 and WinAC

On the SIMATIC side, carry out the following configuration steps:

- **Configuration Software (STEP 7 V5):**

Integration of the Request blocks in the S7 program

- **Connection configuration:**

If a communication processor CP x43-1 is being used, a programmed connection has to be configured in NetPro. This is not necessary when using a PN interface integrated on the CPU.

3.2.1.1 Configuration in STEP 7

In the following, we describe the configuration of the Request blocks in STEP 7 V5.

Copy the required blocks from the iba S7 library to the blocks folder of your STEP 7 project, see [iba S7 library](#), page 107.

Note



The request blocks do not support multi-instance calls.

Note



If the block numbers in your project are already occupied, assign new numbers to the blocks from the iba S7 library when copying.

Adapt the blocks ibaREQ_UDPint, ibaREQ_UDPext3 and ibaREQ_UDPext4 in the following cases:

- For the block ibaREQ_UDPact, another function block number than FB145 is used.
- For the ibaUDT_UDPact data type, another number than UDT145 is used.
- For the Siemens function blocks of the standard library or the SIMATIC NET CP library, other numbers than the standard numbers are used. The relevant Siemens function blocks are:
 - when using ibaREQ_UDPint (FB146):
TCON (FB65), TDISCON (FB66), TUSEND (FB67), TCON_PAR (UDT65), TADDR_PAR (UDT66)
 - when using ibaREQ_UDPext3 (FB147):
AG_SEND (FC5)
 - when using ibaREQ_UDPext4 (FB148):
AG_LSEND (FC50)

For further information on customizing, see [Adaptation to the renumbered system functions](#), page 115.

3.2.1.1.1 CPU S7-300/S7-400/WinAC with integrated PN interface

The following blocks are required:

- ibaREQ_M (FB140), see [ibaREQ_M \(FB140\)](#), page 59
- ibaREQ_UDPact (FB145), see [ibaREQ_UDPact \(FB145\)](#), page 64
- ibaREQ_UDPint (FB146), see [ibaREQ_UDPint \(FB146\)](#), page 68
- ibaREQ_DB (DB15)
- ibaUDT_UDPact (UDT145)

Note

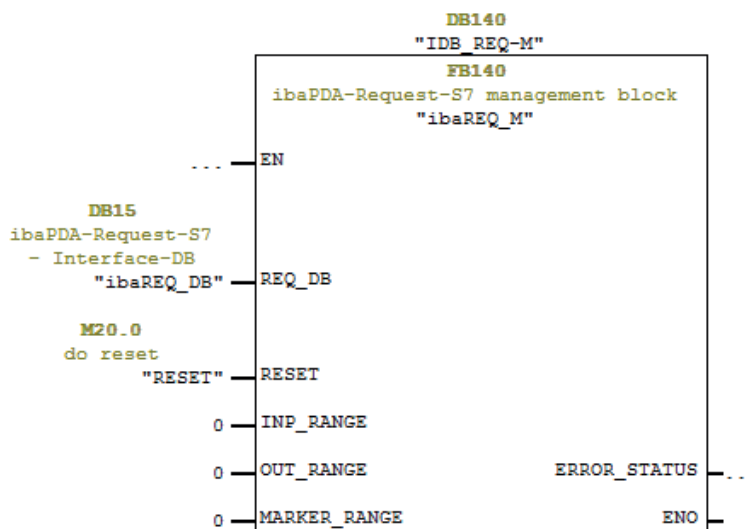


Only use Request blocks from the latest iba S7 library!

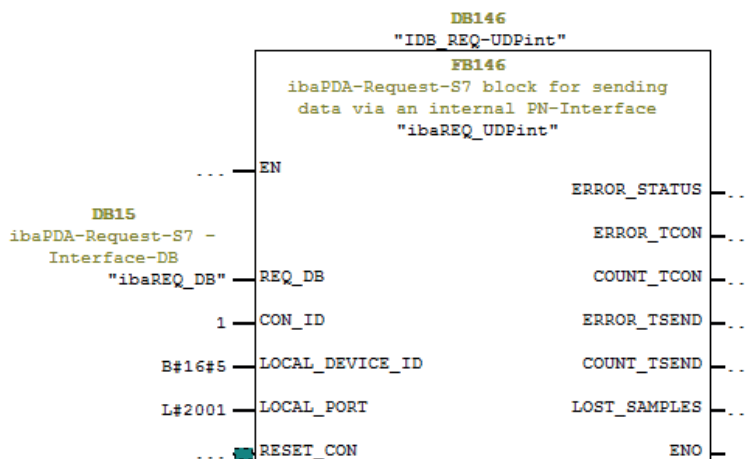
Request blocks in application examples can be outdated and, thus, cause errors.

For each Request module

1. Call the ibaREQ_M (FB140) preferably within the OB1.



2. Call the ibaREQ_UDPint (FB146) preferably within a cyclic interrupt OB (OB3x).



For each additional Request module

- In the blocks folder, a data block ibaREQ_DB (DB15) has to be available for each Request module. Copy the data block and assign a new unique DB number.
- Within the OB1, the ibaREQ_M (FB140) has to be called once more with a new DB number (input REQ_DB) for each Request module.
- Within a cyclic interrupt OB (OB3x), the ibaREQ_UDPint (FB146) has to be called once more with the new DB number (input REQ_DB) for each Request module.
- Make sure that all instance data blocks are unique and that unique values are assigned for the CON_ID and LOCAL_PORT parameters.

Final

- Load all blocks into the S7 CPU and restart the S7 CPU.

3.2.1.1.2 CPU S7-300 with CP343-1

The following function blocks are required:

- ibaREQ_M (FB140), see ↗ *ibaREQ_M (FB140)*, page 59
- ibaREQ_UDPact (FB145), see ↗ *ibaREQ_UDPact (FB145)*, page 64
- ibaREQ_UDPext3 (FB147), see ↗ *ibaREQ_UDPext3 (FB147)*, page 73
- ibaREQ_DB
- ibaUDT_UDPact (UDT145)

Note

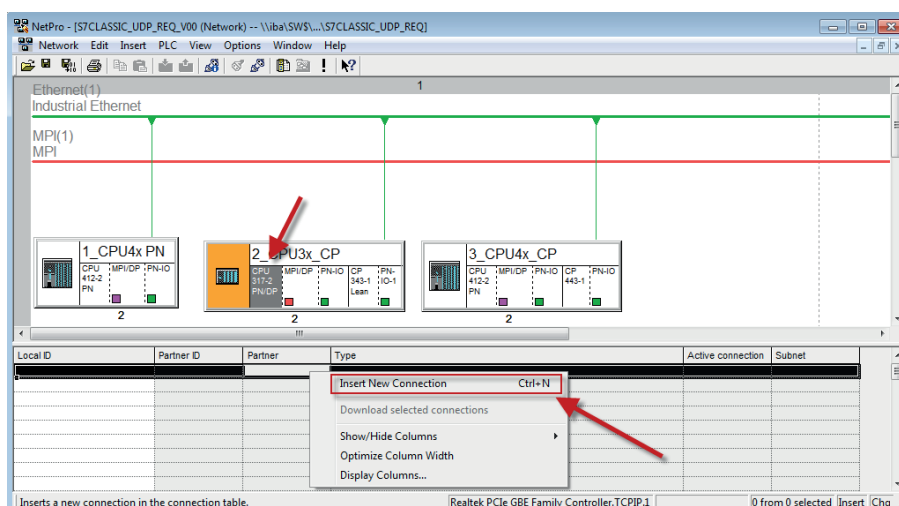


Only use Request blocks from the latest iba S7 library!

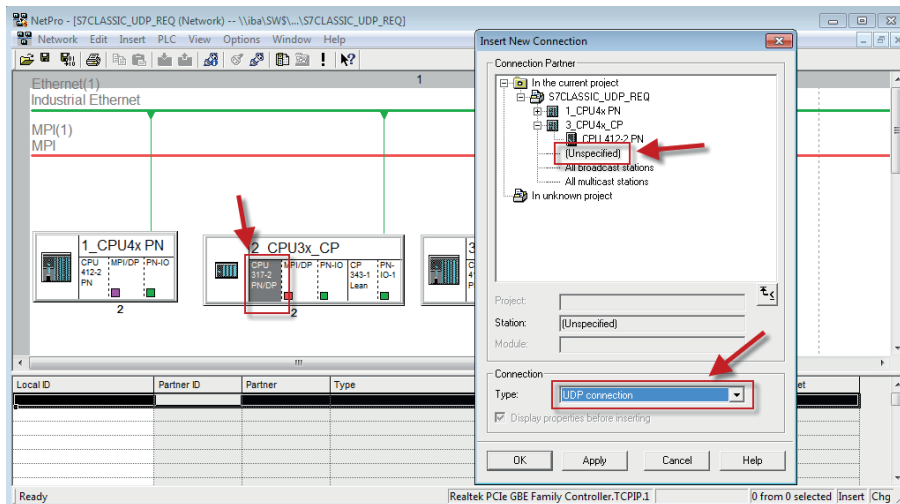
Request blocks in application examples can be outdated and, thus, cause errors.

For each Request module

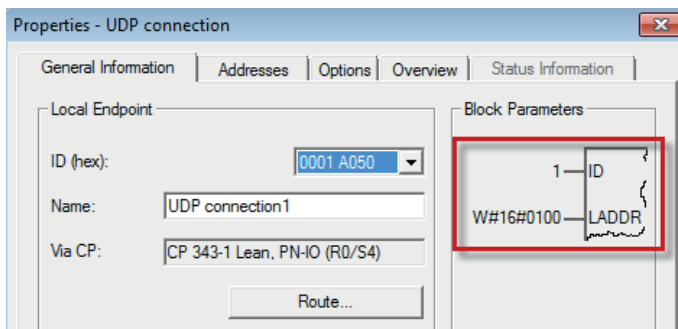
1. Configure a new connection in NetPro.



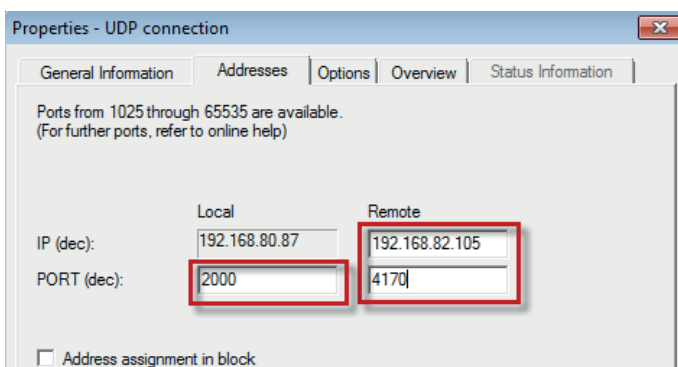
2. Select the connection partner *unspecified* and connection type *UDP connection*.



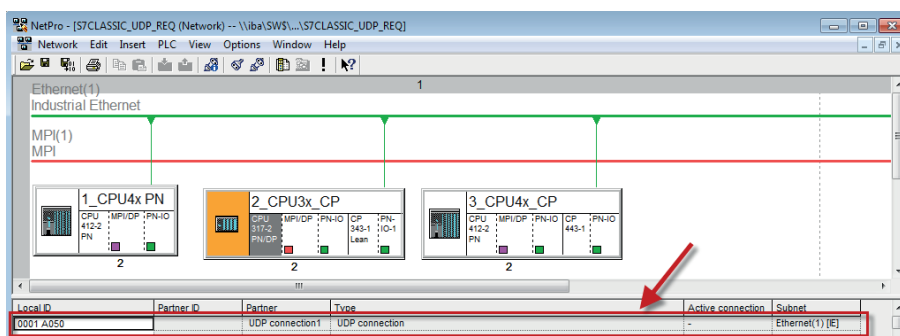
3. The automatically assigned function block parameter for the connection ID (ID) and the hardware starting address (LADDR) are needed later in step 6.



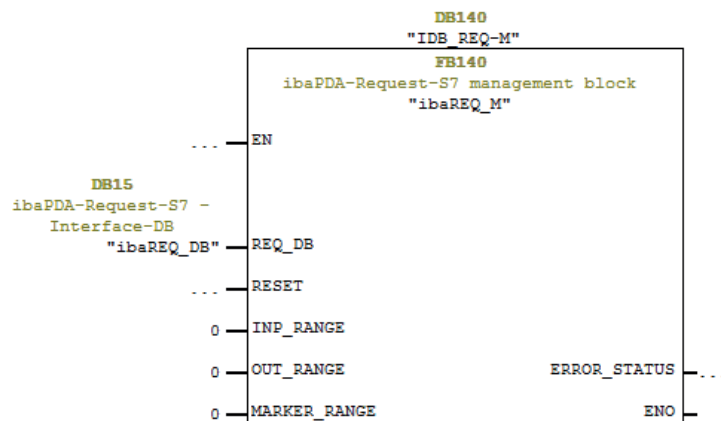
4. Enter the IP address of the *ibaPDA* computer as partner IP address and the configured port number (standard: 4170) and select a unique local port number.



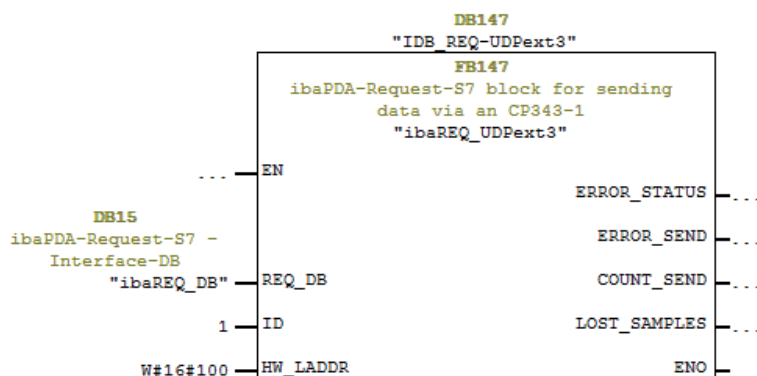
→ The connection table of the CPU now displays the newly configured connection.



5. Call the ibaREQ_M (FB140) preferably within the OB1.



6. Call the ibaREQ_UDPext3 (FB147) preferably within a cyclic interrupt OB (OB3x).



For each additional Request module

- In the blocks folder, an ibaREQ_DB (DB15) data block has to be available for each Request module. Copy the data block and assign a new unique DB number.
- Configure a separate connection for each Request module (steps 1 to 4). Assign different local port numbers.
- Within the OB1, the ibaREQ_M (FB140) has to be called once more with a new DB number (input REQ_DB) for each Request module (step 5).
- Within a cyclic interrupt OB (OB3x), the ibaREQ_UDPext3 (FB147) has to be called once more with the new DB number (input REQ_DB) for each Request module (step 6).
- Make sure that all instance data blocks are unique and that unique values are assigned for the ID and HW_LADDR parameters.

Final

- Load all blocks into the S7 CPU and restart the S7 CPU.

3.2.1.1.3 CPU S7-400 with CP443-1

The following blocks are required:

- ibaREQ_M (FB140), see ↗ *ibaREQ_M (FB140)*, page 59
- ibaREQ_UDPact (FB145), see ↗ *ibaREQ_UDPact (FB145)*, page 64
- ibaREQ_UDPext4 (FB148), see ↗ *ibaREQ_UDPext4 (FB148)*, page 74
- ibaREQ_DB
- ibaUDT_UDPact (UDT145)

Note

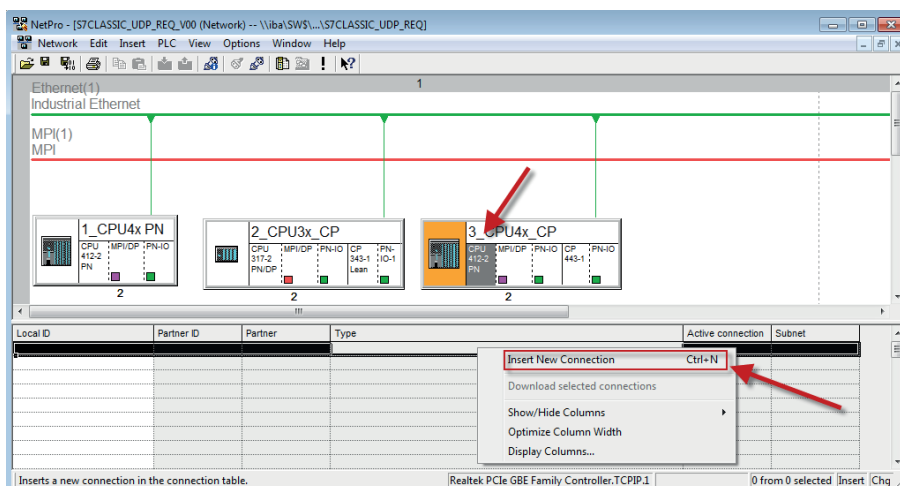


Only use Request blocks from the latest iba S7 library!

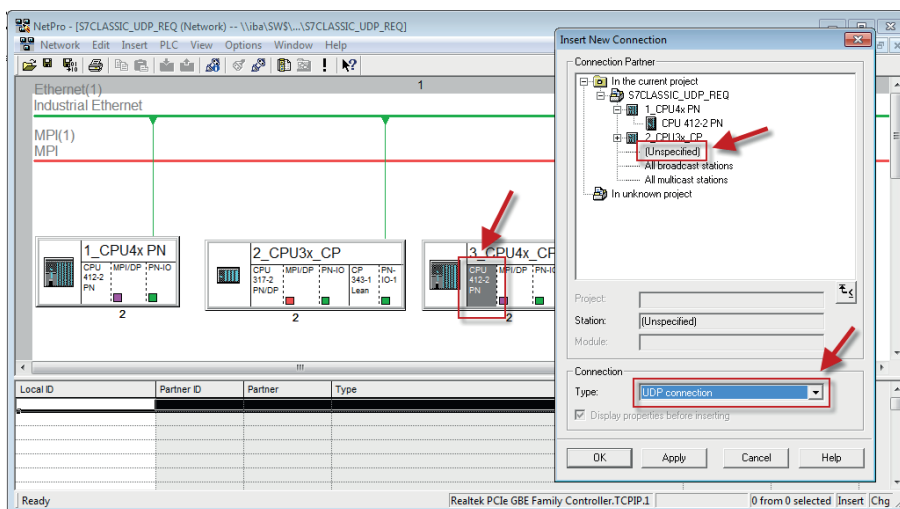
Request blocks in application examples can be outdated and, thus, cause errors.

For each Request module

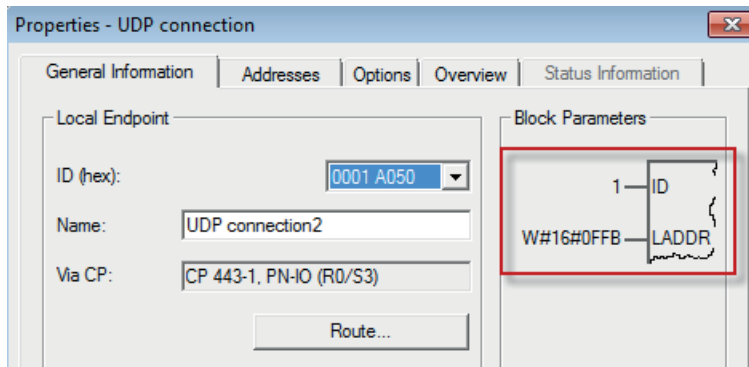
1. Configure a new connection in NetPro.



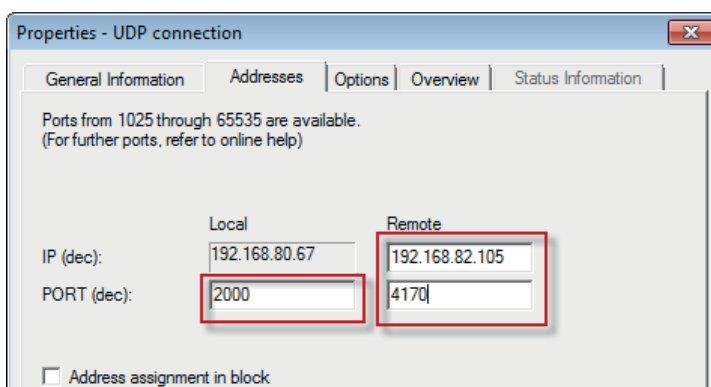
2. Select the connection partner *unspecified* and connection type *UDP connection*.



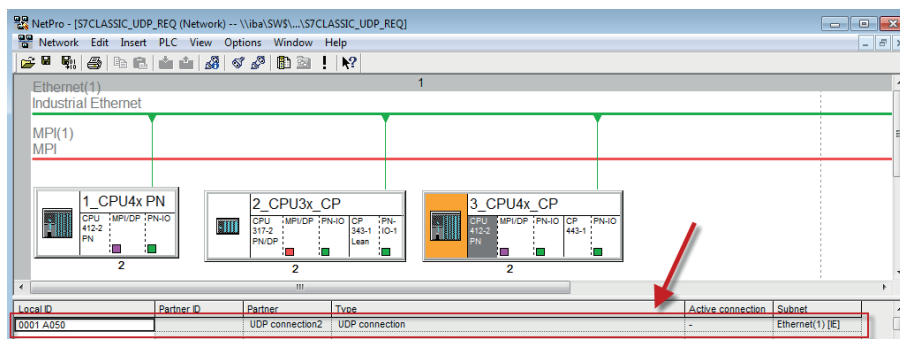
3. The automatically assigned function block parameters are needed later in step 6.



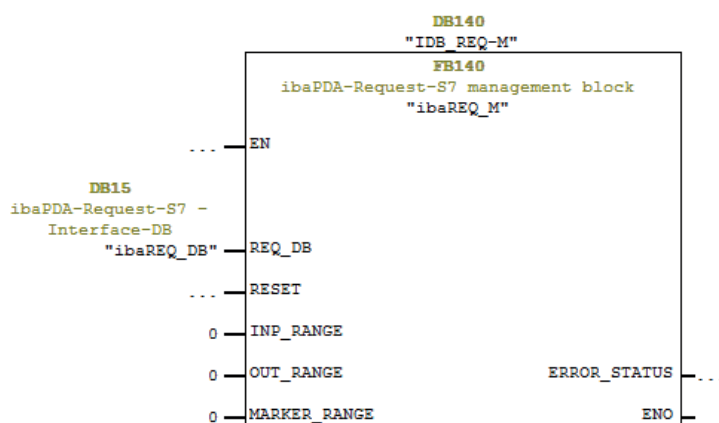
4. Enter the IP address of the *ibaPDA* computer as partner IP address and the configured port number (standard: 4170) and select a unique local port number.



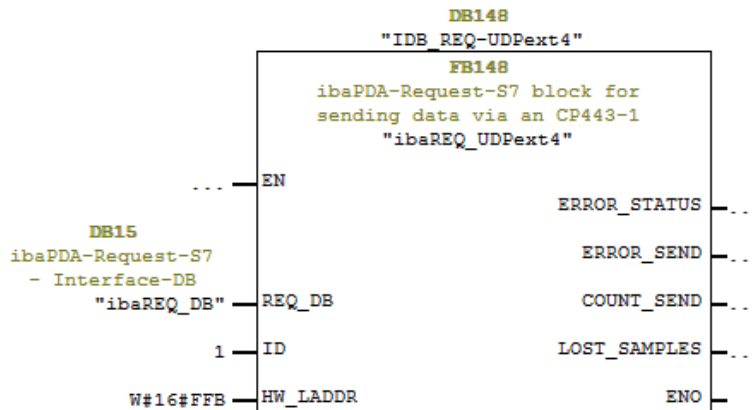
→ The connection table of the CPU now displays the newly configured connection.



5. Call the *ibaREQ_M* (FB140) preferably within the OB1.



6. Call the ibaREQ_UDPext4 (FB148) preferably in the context of a cyclic interrupt OB (OB3x).



For each additional Request module

- In the blocks folder, a data block ibaREQ_DB (DB15) has to be available for each Request module. Copy the data block and assign a new unique DB number.
- Configure a separate connection for each Request module (steps 1 to 4). Assign different local port numbers.
- Within the OB1, the ibaREQ_M (FB140) has to be called once more with the new DB number (input REQ_DB) for each Request module (step 5).
- Within a cyclic interrupt OB (OB3x), the ibaREQ_UDPext4 (FB148) has to be called once more with a new DB number (input REQ_DB) for each Request module (step 6).
- Make sure that all instance data blocks are unique and that unique values are assigned for the ID and HW_LADDR parameters.

Final

- Load all blocks into the S7 CPU and restart the S7 CPU.

3.2.2 Configuration and engineering SIMATIC S7-1500

Carry out the following configuration steps out on the SIMATIC TIA Portal side:

- Configuration software:
Integration of the Request blocks in the S7 program
- Device configuration:
Setting the CPU protection properties

3.2.2.1 Configuration in STEP 7 with the iba block family ibaREQ

The following describes the configuration of the Request blocks in STEP 7.

For each Request module

1. Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see [iba S7 library](#), page 107.
 - ibaREQ_M (FB1400), see [ibaREQ_M \(FB1400\)](#), page 62
 - ibaREQ_UDP2 (FB1406), see [ibaREQ_UDP2 \(FB1406\)](#), page 75
 - ibaREQ_UDPact (FB1410), see [ibaREQ_UDPact \(FB1410\)](#), page 66
 - ibaREQ_DB (DB15)
 - ibaREQ_DB-Interface (PLC data type)

Note



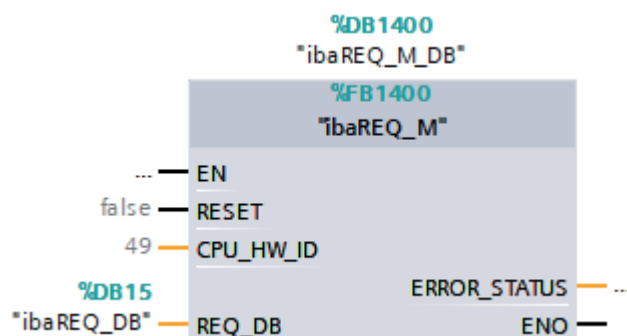
Only use Request blocks from the latest iba S7 library!
Request blocks in application examples can be outdated and, thus, cause errors.

Note

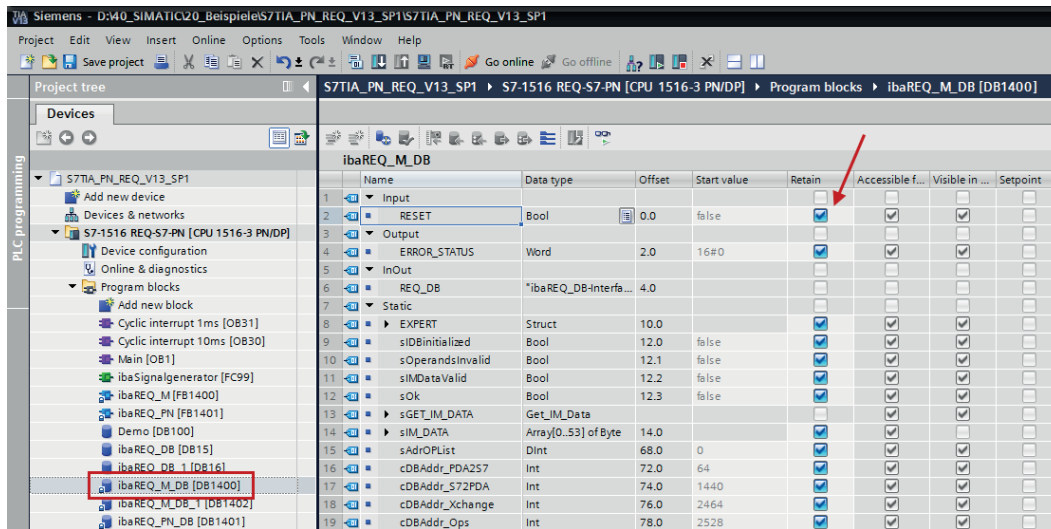


The request blocks do not support multi-instance calls.

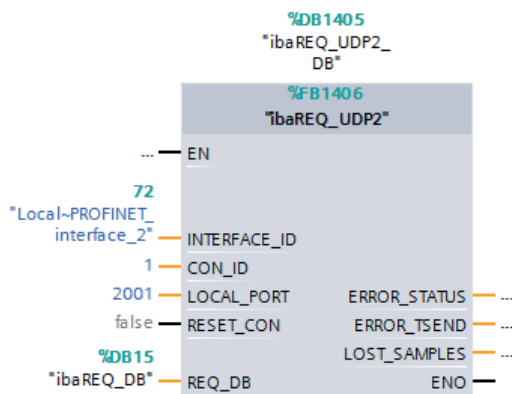
2. Call the ibaREQ_M (FB1400) preferably within the OB1.



3. Enable the *Retain* option for the entire instance block that you have just created.



4. Call the ibaREQ_UDP2 (FB1406) preferably within a cyclic interrupt OB (OB3x).



For each additional Request module

- In the blocks folder, a data block ibaREQ_DB (DB15) has to be available for each Request module. Copy the data block and assign a new unique DB number.
- Within the OB1, the ibaREQ_M (FB1400) has to be called once more with the new DB number for each Request module.
- Within the cyclic interrupt OB (OB3x), the ibaREQ_UDP2 (FB1406) has to be called once more with the new DB numbers for each Request module.
- Make sure that all instance data blocks are unique and that unique values are assigned for the CON_ID and LOCAL_PORT.

Final

- Load all blocks into the S7 CPU and restart the S7 CPU.

3.2.2.2 Configuration in STEP 7 using the ibaREQsym iba block family

This section describes how to configure the Request blocks in TIA Portal STEP 7.

For each Request module

- Copy the following blocks from the iba S7 library to the function block folder of your STEP 7 project, see [iba S7 library](#), page 107.
 - ibaREQsym_M, see [ibaREQsym_M](#), page 80
 - ibaREQsym_UDP, see [ibaREQsym_UDP](#), page 81
 - ibaREQsym_DB_PDA
 - ibaREQsym-Interface (PLC data type)

Note



Only use Request blocks from the latest iba S7 library!

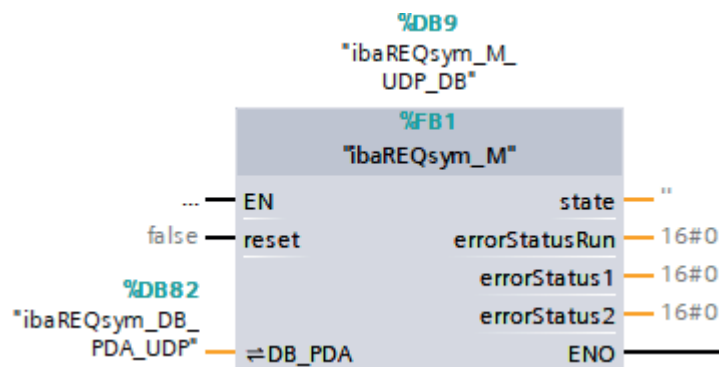
Request blocks in application examples can be outdated and, thus, cause errors.

Note

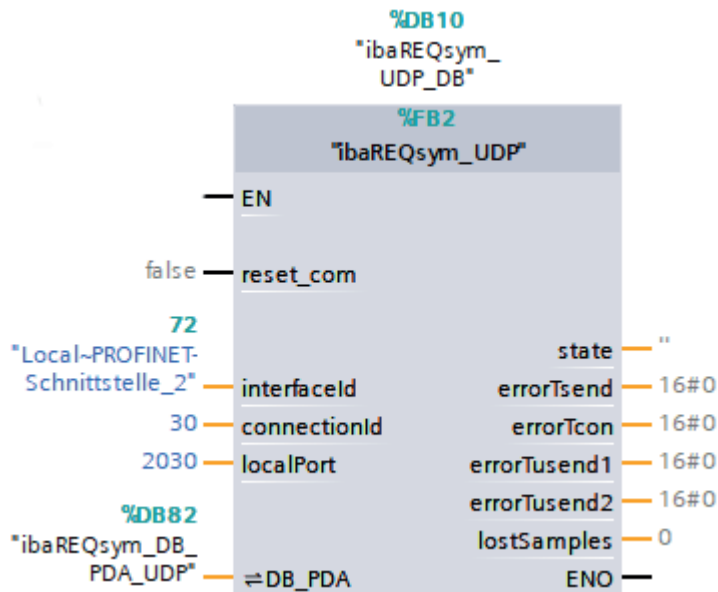


The request blocks do not support multi-instance calls.

- Call ibaREQsym_M preferably within OB1.



3. Call ibaREQsym_UDP preferably a cyclic interrupt OB (OB3x).



For each additional Request module

- An ibaREQsym_DB_PDA data block must be available in the function block folder for each Request module. Copy the data block and assign a new unique DB number.
- Within OB1 or a cyclic interrupt OB (OB3x), ibaREQsym_M and ibaREQsym_UDP have to be called once more with the new DB numbers for each Request module.
- Make sure that all instance data blocks are unique and that unique values are assigned for the connectionId and localPort parameters.

Completion

- Load all blocks to the S7 CPU and restart the S7 CPU.

3.2.2.3 Configuration in STEP 7 using the ibaREQ3sym iba block family

This section describes how to configure the Request blocks in TIA Portal STEP 7.

For each Request module

1. Copy the following blocks from the iba S7 library to the function block folder of your STEP 7 project, see [iba S7 library](#), page 107.
 - ibaREQ3sym_M, see [ibaREQ3sym_M](#), page 89
 - ibaREQ3sym_UDP, see [ibaREQ3sym iba block family](#), page 88
 - ibaREQ3sym_DB_PDA
 - ibaREQ3sym-Interface (PLC data type)

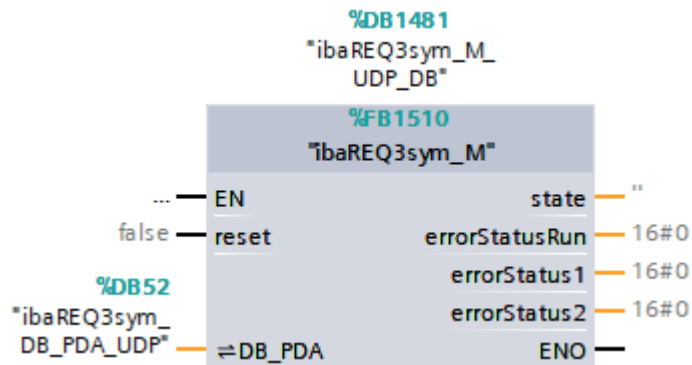
Note



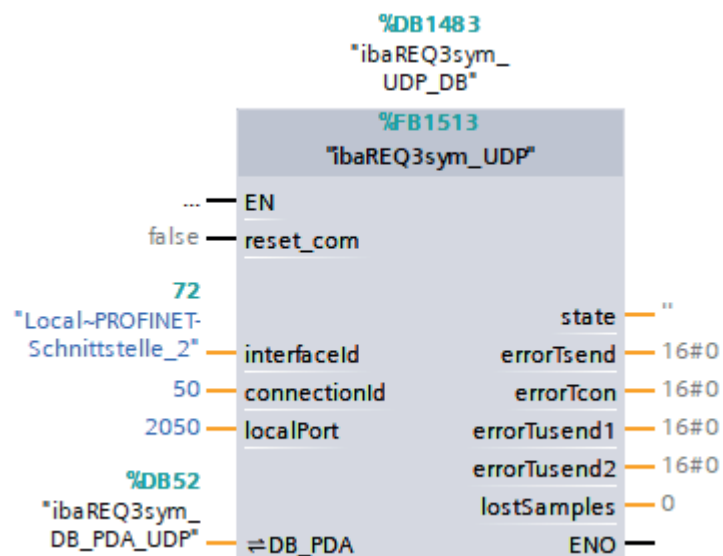
Only use Request blocks from the latest iba S7 library!

Request blocks in application examples can be outdated and, thus, cause errors.

- Call ibaREQ3sym_M preferably within OB1.



- Call ibaREQ3sym_UDP preferably a cyclic interrupt OB (OB3x).



For each additional Request module

- An ibaREQ3sym_DB_PDA data block must be available in the function block folder for each Request module. Copy the data block and assign a new unique DB number.
- ibaREQ3sym_M and ibaREQ3sym_UDP have to be called once more with the new DB numbers for each Request module.
- Make sure that all instance data blocks are unique and that unique values are assigned for the connectionId and localPort parameters.

Completion

- Load all blocks to the S7 CPU and restart the S7 CPU.

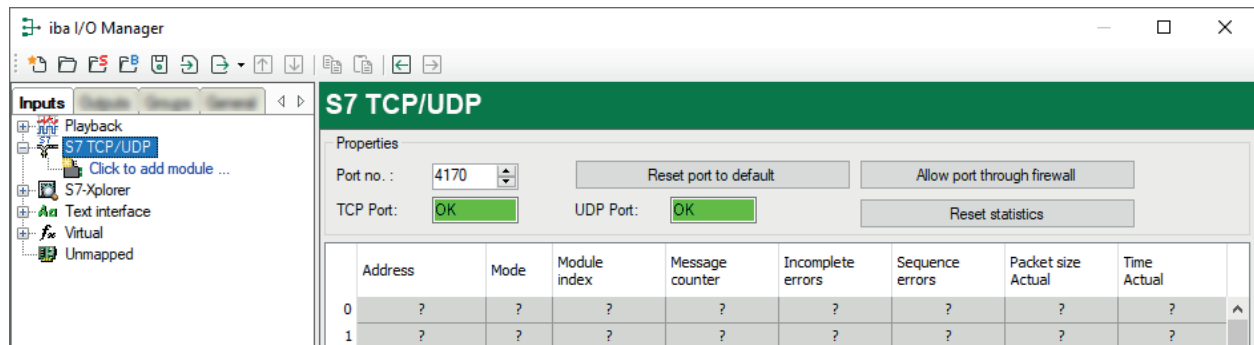
3.3 Configuration and engineering ibaPDA

3.3.1 General interface settings

If all system requirements are fulfilled, *ibaPDA* displays the *S7 TCP/UDP* interface in the interface tree of the I/O Manager. *ibaPDA-Request-S7-UDP* is a module (*S7 Request*) of this interface.

If you select the data interface in the tree, you can see an overview of diagnostics information on the configured connections between *ibaPDA* and the controllers.

The interface has the following features and configuration options.



Port no.

Used port on the computer. You can change the port number, but in the S7 project engineering and in *ibaPDA* you must use the same port to establish a connection.

The default port number is 4170.

<Reset port to default>

Use this button to reset the port to the default port number.

<Allow ports through firewall>

When installing *ibaPDA*, the default port numbers of the used protocols are automatically entered in the firewall. If you change the port number or enable the interface subsequently, you have to enable this port in the firewall with this button.

TCP Port/UDP Port

Displays the port status.

- OK: You can open the socket on this port.
- ERROR: Conflicts occur, e.g. the port is already occupied.

<Reset statistics>

Click this button to reset the calculated times and error counters in the table to 0.

Connection table

For each connection, the table shows the connection status, the current values for the update time (actual value, average, min. and max.) as well as the data size. In addition, there is an error counter for the individual connections during the acquisition.

See ➤ *Connection table*, page 101.

By double-clicking on a row, you can open the configuration of the corresponding module.

For information about the connection diagnostics, see [↗ Diagnostics](#), page 97.

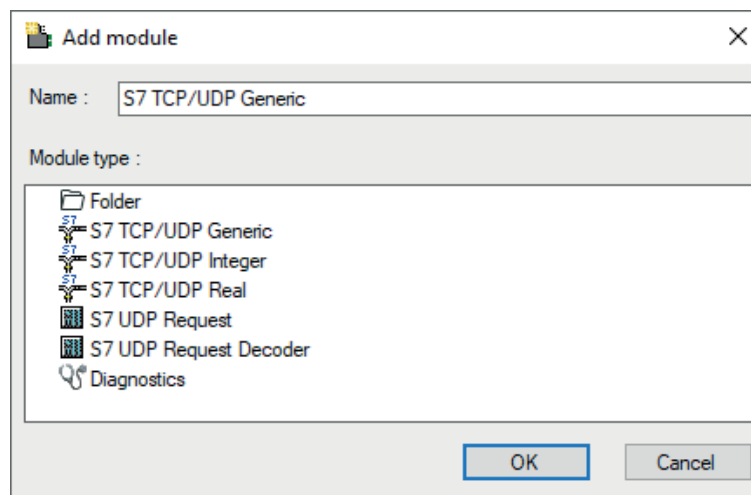
Other documentation



For more information about the interface *ibaPDA-Interface-S7-TCP/UDP*, see the corresponding manual.

3.3.2 Adding a module

1. Click on the blue link *Click to add module* located under each data interface in the *Inputs* or *Outputs* tab.
2. Select the desired module type in the dialog box and assign a name via the input field if required.
3. Confirm the selection with <OK>.



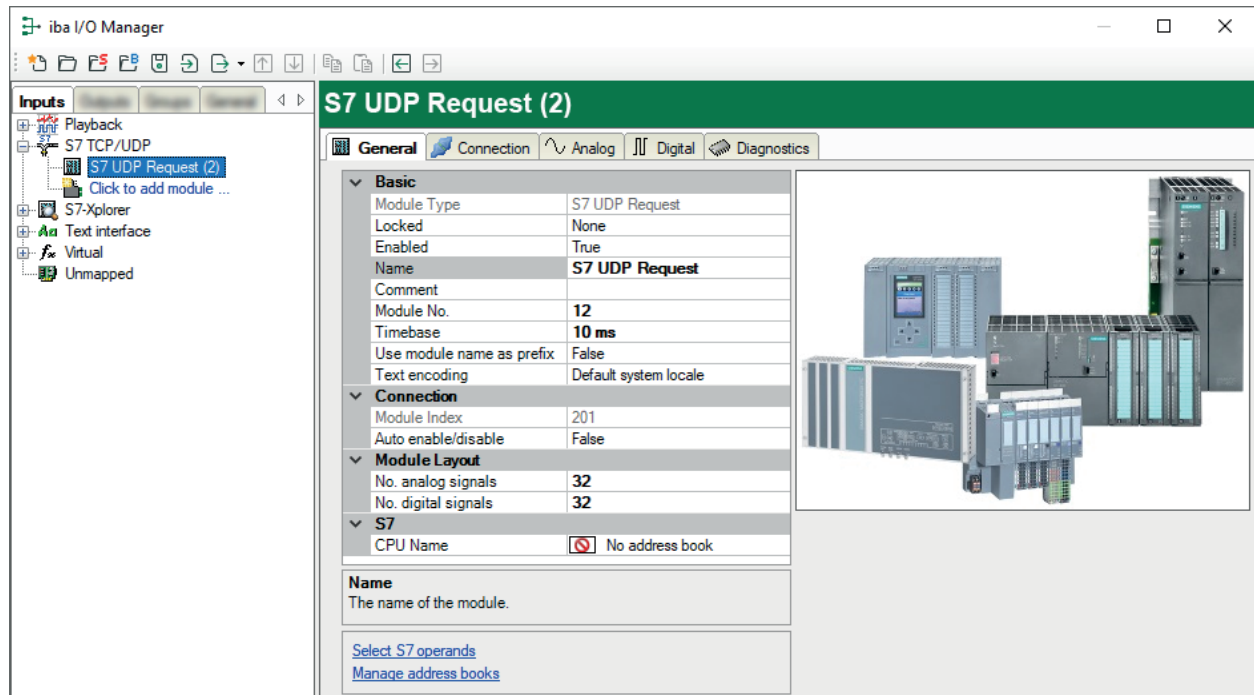
Module name	Description
S7 UDP Request	Request module for a max. of 1024 analog signals and 1024 digital signals.
S7 UDP Request Decoder	Request module for a max. of 11728 digital signals which are transmitted as 733 words (1466 Byte).

Table 1: Module overview of the Request-S7-UDP interface

3.3.3 General module settings

To configure a module, select it in the tree structure.

All modules have the following setting options.



Basic settings

Module Type (information only)

Indicates the type of the current module.

Locked

You can lock a module to avoid unintentional or unauthorized changing of the module settings.

Enabled

Enable the module to record signals.

Name

You can enter a name for the module here.

Comment

You can enter a comment or description of the module here. This will be displayed as a tooltip in the signal tree.

Module No.

This internal reference number of the module determines the order of the modules in the signal tree of *ibaPDA* client and *ibaAnalyzer*.

Timebase

All signals of the module are sampled on this timebase.

Use module name as prefix

This option puts the module name in front of the signal names.

Text encoding

You can select the type of text encoding or the code page here for a correct interpretation and display of the received text data for inputs as well as of the text data to be sent for outputs. Available for selection are, beside system locale according to the Windows system settings (default) and UTF-8 Unicode, all other encodings.

Connection**Module index (information only)**

Internal reference number of the module.

Auto enable/disable

If TRUE, the acquisition is started, even if no connection can be established to the S7-CPU. The module is deactivated. During the acquisition, *ibaPDA* tries to reconnect to the S7-CPU. When it succeeds, the acquisition is restarted with this module enabled.

If FALSE, the acquisition is not started, if a connection to the configured S7-CPU cannot be established.

Module Layout**No of analog signals/digital signals**

Define the number of configurable analog and digital signals in the signal tables. The default value is 32 for each. The maximum value is 1024. The signal tables are adjusted accordingly.

S7**CPU Name**

Select the S7-CPU that is connected to this module. When selecting a S7-CPU (incl. the address book), you can select the signals symbolically. Otherwise, you select the signals via the S7 operand.

This requires that address books have already been generated. Otherwise, the selection list is empty. Using *Create address book* in the selection list, you get directly to the address book generator, see [🔗 Creating address books offline from S7 project](#), page 52.

3.3.4 Connection settings

Configure the connection of the module to the controller in the *Connection* tab.

ibaPDA supports the following controllers, connection modes and selection methods:

Controller	Connection mode		
	TCP/IP	PC/CP	TCP/IP S7-1x00
S7-300	X	X	-
S7-400	X	X	-
S7-1500	X	-	X

Configure different settings depending on the connection mode.

3.3.4.1 Connection mode TCP/IP

This mode activates a connection via the standard network interface of the computer.

S7 UDP Request (2)

General **Connection** Analog Digital Diagnostics

Connection

Connection mode: TCP/IP Connection type: PG connection Timeout (s): 15

Address: 192.168.123.1 Rack: 0 Slot: 0 Test

☐ Activate S7 routing

DB: 15

CPU Name: No address book ☒ Detect S7 restart (This applies to all S7 request modules)

Connection mode

Selection of the TCP/IP connection mode

Connection type

Selection of the connection type PG, OP, or other connections (determines which type of connection resource is occupied on the CPU).

Timeout

Specify a value for the timeout in seconds for establishing the connection and for read access. If the time set here is exceeded, *ibaPDA* declares the controller as not accessible or not responsive.

Address

IP address of the controller

Rack

Rack number of the controller (default: 0)

Slot

Slot number of the controller in the rack
(Use "0" for S7-1500 CPUs.)

Activate S7 routing

Activate this option if the S7-CPU and the *ibaPDA* computer are not in the same network, but only communicate over a gateway that supports S7 routing. Such a gateway can be e.g. an IE/PB link, over which a S7-CPU can be reached without an Ethernet connection.

Two additional input fields appear:

- Address of device acting as gateway: Enter the IP address of the gateway.
- S7 subnet ID of target net: Enter subnet ID from STEP 7 NetPro or TIA Portal.

For more information on S7 routing, see [S7 routing](#), page 120.

DB

Number of the data block used as *ibaPDA* communication interface (ibaREQ_DB)

CPU Name

Selection of the linked address book

Detect S7 restart

The current request configuration is stored in a data block on the CPU. In case the *Detect S7 restart* option is enabled, *ibaPDA* can detect if this data block has been deleted or overwritten, e.g. as a result of loading the offline program or due to a cold restart, and restarts the data acquisition. The configuration data are transferred again. This does not affect a warm restart of the CPU.

<Test>

ibaPDA tests the connection to the CPU and displays available diagnostic data.

S7 UDP Request (2)

General | **Connection** | Analog | Digital | Diagnostics

Connection mode: TCP/IP | Connection type: PG connection | Timeout (s): 15

Address: 192.168.123.1 | Rack: 0 | Slot: 0 | Test

☐ Activate S7 routing

DB: 15

CPU Name: No address book | ☒ Detect S7 restart (This applies to all S7 request modules)

Connection established
 MLFBNr of PLC is: **6ES7 412-2EK06-0AB0**
 PLC status: **RUN**
 Cycle times: Actual 1 ms | Min 1 ms | Max 2 ms
 Reading **DB15**
 DB id: **ibaREQ-S7-M**
 DB version: **1.0.0.0**
 FB version: **1.0.0.0**
 DB length: **9120**
 Max. pointers: **512**
 Max. data bytes: **1466**
 HW version: **0**
 Total memory size: **1072432**
 DB memory size: **528384**
 DB used size: **15250**
 Code memory size: **544048**
 Code used size: **29416**
 No. inputs: **128**
 No. outputs: **128**
 No. markers: **4096**
 No. timers: **2048**
 No. counters: **2048**
 I/O space: **4096**
 Local datasize: **4096**

Tip



Error message "DB is not a valid request DB ..."

Check the following:

- The Request block has been loaded into the CPU.
- The right DB number has been configured on the Request block.
- The Request block is called in the program.
- Possibly, the DB is written from another source.

3.3.4.2 Connection mode PC/CP

This mode activates a connection over the interface cards of the computer that are configured using SIMATIC Net.

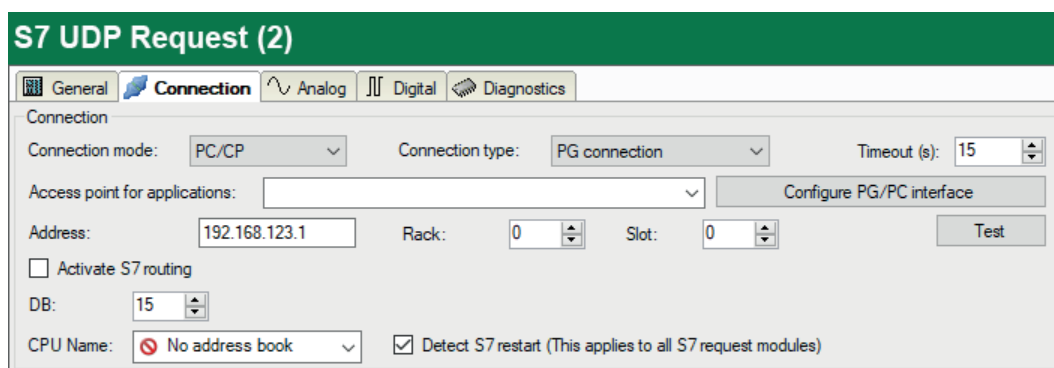
You can use the interfaces configured in SIMATIC Net, e.g.:

- MPI adapter (COM)
- MPI adapter (USB)
- PROFIBUS (CP5611, CP5622)
- TCP/IP (RFC1005)
- ...

Note



If you want to use this connection type, the Siemens software SIMATIC Net (e.g. SIMATIC Manager or Softnet) has to be installed. When using the modules CP55..., CP56... and the MPI adapter, the installation of the device drivers is sufficient.



Connection mode

Selection of the PC/CP connection mode

Connection type

Selection of the connection type PG, OP, or other connections (determines which type of connection resource is occupied on the CPU).

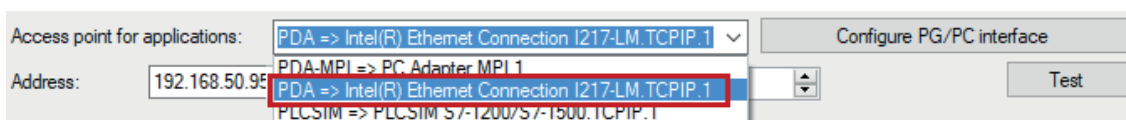
Timeout

Specify a value for the timeout in seconds for establishing the connection and for read access. If the time set here is exceeded, *ibaPDA* declares the controller as not accessible or not responsive.

Access point for applications

Selection of the access point to be used

For more information on creating and configuring an access point, see [➤ Setting PG/PC interface/defining new access point](#), page 117.



Note

Configure available access points in SIMATIC Net with the "PG/PC interface settings" tool by Siemens.

For the connection of *ibaPDA-Request-S7-UDP* to SIMATIC S7 via PC/CP connections, iba generally recommends setting a specific access point for *ibaPDA* when *ibaPDA-Request-S7-UDP* and SIMATIC Manager run on the same computer. With an own access point, there is no longer the risk that the access for *ibaPDA-Request-S7-UDP* will be disturbed in case the standard access point is changed in the SIMATIC Manager.

<Configure PG/PC interface>

This button opens the dialog box for setting the PG/PC interface of SIMATIC STEP 7.

Address

Address of the controller (MPI, PROFIBUS or IP address depending on the configured access point)

Rack

Rack number of the controller (default: 0)

Slot

Slot number of the controller in the rack
(Use "0" for S7-1500 CPUs.)

Activate S7 routing

Activate this option if the S7-CPU and the *ibaPDA* computer are not in the same network, but only communicate over a gateway that supports S7 routing. Such a gateway can be e.g. an IE/PB link, over which a S7-CPU can be reached without an Ethernet connection.

Two additional input fields appear:

- Address of device acting as gateway: Enter the IP address of the gateway.
- S7 subnet ID of target net: Enter subnet ID from STEP 7 NetPro or TIA Portal.

For more information on S7 routing, see ➤ *S7 routing*, page 120.

DB

Number of the data block used as *ibaPDA* communication interface (ibaREQ_DB)

CPU Name

Selection of the linked address book

Detect S7 restart

The current request configuration is stored in a data block on the CPU. In case the *Detect S7 restart* option is enabled, *ibaPDA* can detect if this data block has been deleted or overwritten, e.g. as a result of loading the offline program or due to a cold restart, and restarts the data acquisition. The configuration data are transferred again. This does not affect a warm restart of the CPU.

<Test>

ibaPDA tests the connection to the CPU and displays available diagnostic data.

S7 UDP Request (2)

General Connection Analog Digital Diagnostics

Connection

Connection mode: PC/CP Connection type: PG connection Timeout (s): 15

Access point for applications: ibaTCP => TCP/IP -> Intel(R) PRO/1000 PL N... Configure PG/PC interface

Address: 192.168.123.1 Rack: 0 Slot: 0 Test

☐ Activate S7 routing

DB: 15

CPU Name: No address book ☒ Detect S7 restart (This applies to all S7 request modules)

Connection established
 MLFBNr of PLC is: **6ES7 412-2EK06-0AB0**
 PLC status: **RUN**
 Cycle times: Actual 1 ms Min 1 ms Max 2 ms
 Reading **DB15**
 DB id: **ibaREQ-S7-M**
 DB version: **1.0.0.0**
 FB version: **1.0.0.0**
 DB length: **9120**
 Max. pointers: **512**
 Max. data bytes: **1466**
 HW version: **0**
 Total memory size: **1072432**
 DB memory size: **528384**
 DB used size: **15250**
 Code memory size: **544048**
 Code used size: **29416**
 No. inputs: **128**
 No. outputs: **128**
 No. markers: **4096**
 No. timers: **2048**
 No. counters: **2048**
 I/O space: **4096**
 Local datasize: **4096**

Tip

Error message "DB is not a valid request DB ..."

Check the following:

- The Request block has been loaded into the CPU.
- The right DB number has been configured on the Request block.
- The Request block is called in the program.
- Possibly, the DB is written from another source.

3.3.4.3 Connection mode TCP/IP S7-1x00

This mode activates a connection via the standard network interface of the computer. You can only use this mode in combination with S7-1500 CPUs.

S7 UDP Request (2)

General Connection Analog Digital Diagnostics

Connection

Connection mode: TCP/IP S7-1x00 Connection type: PG connection Timeout (s): 15

Address: 192.168.123.1 Test

Password: ☐ Use secure communication

DB: None Load address book from S7

CPU Name: No address book ☒ Detect S7 restart (This applies to all S7 request modules)

Connection mode

Selection of the TCP/IP S7-1x00 connection mode

Connection type

Selection of the connection type PG, OP, or other connections (determines which type of connection resource is occupied on the CPU).

Timeout

Specify a value for the timeout in seconds for establishing the connection and for read access. If the time set here is exceeded, *ibaPDA* declares the controller as not accessible or not responsive.

Address

IP address of the controller

Password

Depending on the configuration in the controller, access to the PLC may be protected by a password. In this case, enter this password here.

Use secure communication

The S7-1500 controller supports secure communication via TLS encryption with TIA Portal v17 or higher. In TIA Portal, you can set secure PG/PC and HMI communication.

If you have activated this option in the controller, you must also activate secure communication in *ibaPDA*.

DB

Number of the data block used as *ibaPDA* communication interface (ibaREQ_DB)

CPU Name

Selection of the linked address book (only TIA Portal address books available).

Note



The TCP/IP S7-1x00 connection mode does not support the use of absolute operands and addresses.

Note

For the communication with the CPU, port 102 must be allowed in the target system. If the data traffic runs via an external firewall, then you must also allow port 102 through this firewall.

Detect S7 restart

The current request configuration is stored in a data block on the CPU. In case the *Detect S7 restart* option is enabled, *ibaPDA* can detect if this data block has been deleted or overwritten, e.g. as a result of loading the offline program or due to a cold restart, and restarts the data acquisition. The configuration data are transferred again. This does not affect a warm restart of the CPU.

<Test>

ibaPDA tests the connection to the CPU and displays available diagnostic data.

S7 UDP Request (2)

General Connection Analog Digital Diagnostics

Connection

Connection mode: TCP/IP S7-1x00 Connection type: PG connection Timeout (s): 15

Address: 192.168.80.90 Test

Password: Use secure communication

DB: ibaREQ_DB_UDP (DB) Load address book from S7

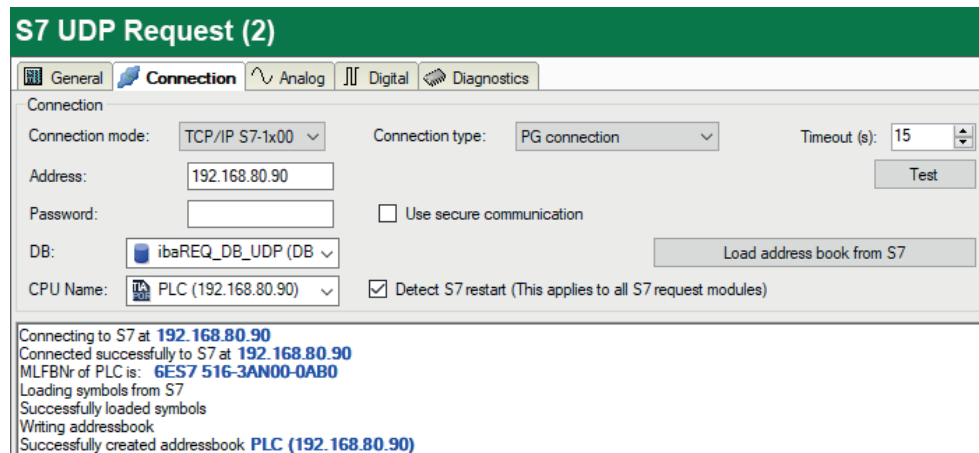
CPU Name: PLC (192.168.80.90) Detect S7 restart (This applies to all S7 request modules)

Connection established
 MLFBNr of PLC is: 6ES7 516-3AN00-0AB0
 Reading ibaREQ_DB_UDP (DB19)
 DB id: ibaREQ-S7-M
 DB version: 1.0.0.0
 FB version: 1.3:1.5
 DB length: 9120
 Max. pointers: 512
 Max. data bytes: 1466

HW version: 0
 Total memory size: 0
 DB memory size: 0
 DB used size: 0
 Code memory size: 0
 Code used size: 0
 No. inputs: 32768
 No. outputs: 32768
 No. markers: 16384
 No. timers: 2048
 No. counters: 2048
 I/O space: 0
 Local datasize: 0

<Load address book from S7>

By clicking on this button *ibaPDA* reads the list of symbols directly from the PLC and stores it in an address book for further use in the symbol browser.


Tip

Error message "DB is not a valid request DB ..."

Check the following:

- The Request block has been loaded into the CPU.
- The right DB number has been configured on the Request block.
- The Request block is called in the program.
- Possibly, the DB is written from another source.

In this context, also note the access protection of an S7-1500 CPU.

For more information, see ➤ *Device configuration*, page 57.

3.3.5 Signal configuration

In the I/O Manager, you select the signals to be acquired. There are 3 ways of selecting measured values:

- Selection using the absolute address of the S7 operands, see [↗ Selection via the absolute address of the operands](#), page 40
- Selection using the S7 symbol addresses (symbol table and symbols from DBs) using a symbol browser, see [↗ Selection via the symbolic operand addresses](#), page 41
- Selection using the CFC connectors (when programming the CPU with SIMATIC CFC), see [↗ Selection of CFC connectors](#), page 44

SIMATIC CPU	Access using absolute address	Access using symbol	Access using CFC connectors ²⁾
S7-300	X	X	X
S7-400	X	X	X
WinAC	X	X	X
S7-1500	X	X	-

Supported operand ranges:

Operand range	SIMATIC CPUs S7-300/400	SIMATIC CPUs S7-1500
Inputs (I)	X	X
Peripheral inputs (PI)	X	-
Outputs (O)	X	X
Markers (M)	X	X
Data blocks (DB)	X	X

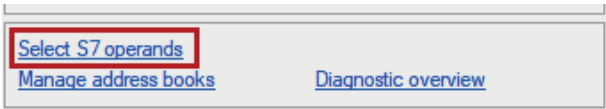
For S7-1500 you can use optimized data blocks. Signals within these data blocks can only be accessed using their symbolic names and not using the address or the operand. To access these data ranges, you must use the Request blocks from the ibaREQsym family, see [↗ ibaREQsym iba block family](#), page 79.

²⁾ The prerequisite for this is use of the optional S7-CFC SIMATIC STEP 7 package. For TIA portal, SIMATIC CFC is not supported.

3.3.5.1 Selection via the absolute address of the operands

You have two options to select the measurement values via the operand address:

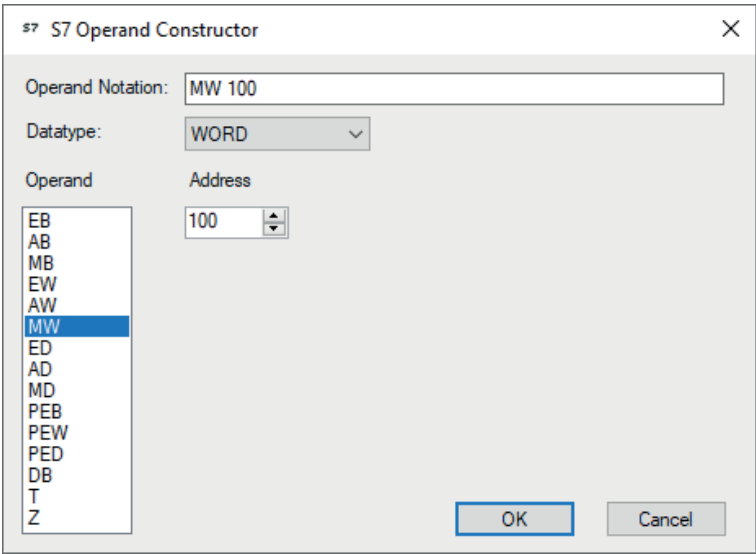
- In the module's *General* tab, click on the *Select S7 operands* link.



The S7 operand editor opens.

- In the *Analog* or *Digital* tab, click in a cell in the *S7 operand* column.

The button <...> appears. Click the button to open the S7 operand editor.



After you have set the desired operand address, click <OK> to exit the dialog.

Thereafter, you can enter the signal name in the *Name* column.

<div><div><div><div><div></div><div>S7</div></div></div><div>General</div><div>Connection</div><div>Analog</div><div>Digital</div><div>Diagnostics</div></div></div>								
	Name		Unit	Gain	Offset	S7 Operand	S7 DataType	Active
0	counter 16bit			1	0	MW 100	WORD	<input checked="" type="checkbox"/>
1	counter 32bit			1	0	MD 104	DWORD	<input checked="" type="checkbox"/>
2	sinus			1	0	MD 112	REAL	<input checked="" type="checkbox"/>
3	cosinus			1	0	MD 116	REAL	<input checked="" type="checkbox"/>
4				1	0		INT	<input type="checkbox"/>

You can enter the desired operand address also directly in the *S7 Operand* column without using the S7 operand editor.

3.3.5.2 Selection via the symbolic operand addresses

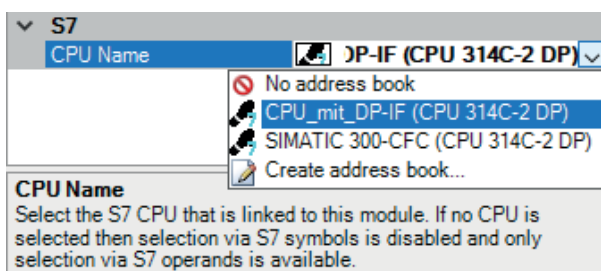
An advantage of this way of access is that the symbol addresses are applied automatically in *ibaPDA* as signal names.

Requirements for this method of access:

- The signals to be measured already have an entry in the S7 symbol table, the PLC variable list or in a data block.
- An address book has been created (see chapter [Address books](#), page 51).

Integrating an address book into a module

- In the module's *General* tab under *CPU Name*, select the S7-CPU you want to assign this module to.



→ In the *Analog* and *Digital* tabs, an additional column *S7 Symbol* is displayed.

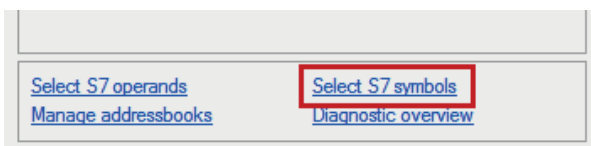
S7							
General Connection Analog Digital Diagnostics							
Name	Unit	Gain	Offset	S7 Operand	S7 DataType	Active	
0		1	0		INT	<input type="checkbox"/>	^
1		1	0		INT	<input type="checkbox"/>	
2		1	0		INT	<input type="checkbox"/>	
3		1	0		INT	<input type="checkbox"/>	
4		1	0		INT	<input type="checkbox"/>	

→ Now you can access symbol addresses using the S7 CFC- and Symbol Browser (in short: symbol browser).

Selecting signals via the symbol browser

You have two options to select the signals to be measured:

- In the module's *General* tab, click on the *Select S7 symbols* link.



The symbol browser opens.

In the symbol browser, you can select all symbols of the address book. *ibaPDA* enters the selected signals automatically in the appropriate table *Analog* or *Digital*. You can add several signals successively.

- On the *Analog* or *Digital* tab, click in a cell of the *S7 Symbol* column.

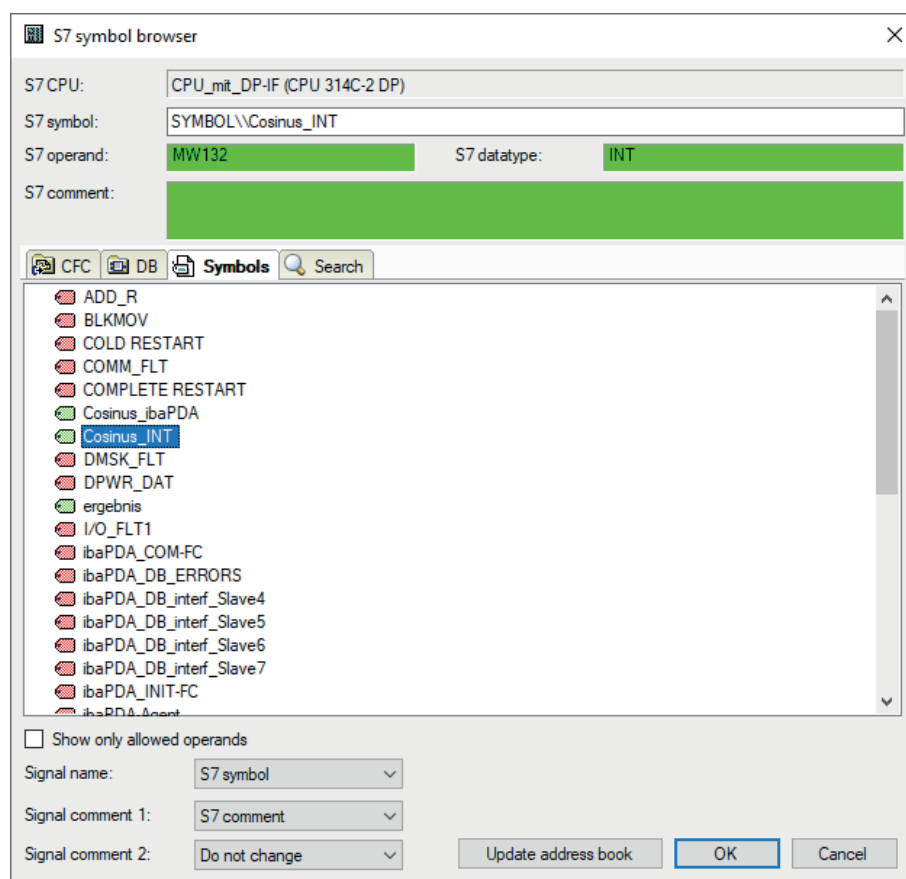
The button <...> appears. Click this button to open the symbol browser.

In the symbol browser, you can only select symbols with the data type matching the table. *ibaPDA* enters the symbol in the previously selected row of the signal table. After every selection, the symbol browser closes.

User interface of the symbol browser

In the symbol browser, you have the following options:

- **CFC variables:**
On the *CFC* tab, you can select the configured CFC variables consisting of the configured names of chart, block and connector.
- **DB variables:**
On the *DB* tab, you can select the individual data blocks and their variables.
- **Symbol table:**
On the *Symbols* tab, you can select the entries of the S7 symbol table.
- **Search tab:**
You can search the variables with a part of their name.



After selecting the variable, the symbol browser shows the operand address, the data type and a comment.

The variables have the following colors:

Green	The operand is valid. You can transfer it to the signal table with <Add> or <OK>.
Yellow	The operand has a data type that does not match the selected row or table, e. g. in case you have selected a Boolean variable as analog value or an integer value as digital value.
Red	The operand has a data type that is not supported by <i>ibaPDA</i> , or the operand is a constant.

Show only allowed operands

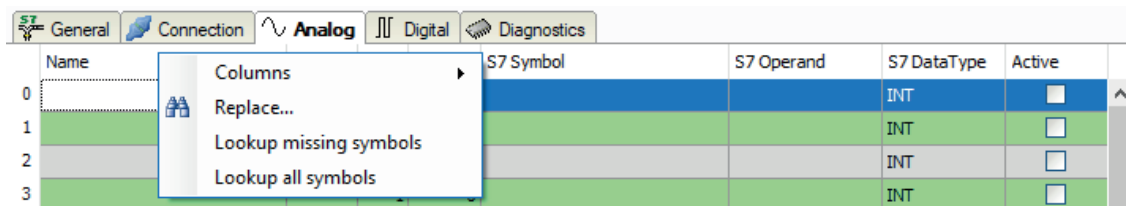
If you activate this option, the symbol browser shows only operands and symbols that *ibaPDA* supports or that match the signal table from which you opened the browser (i.e. no red and yellows ones).

Signal name, signal comment 1 and 2

Usually, *ibaPDA* adopts the symbolic signals name from STEP 7 as signal name in the I/O Manager. Using these three drop-down menus, you can change the signal name and both comments.

Please select an option from the alternatives offered. If a signal name or comment in the signal table should not be changed, select *Do not change*.

Looking up symbols in the signal table



ibaPDA can look up the symbol that corresponds with an operand.

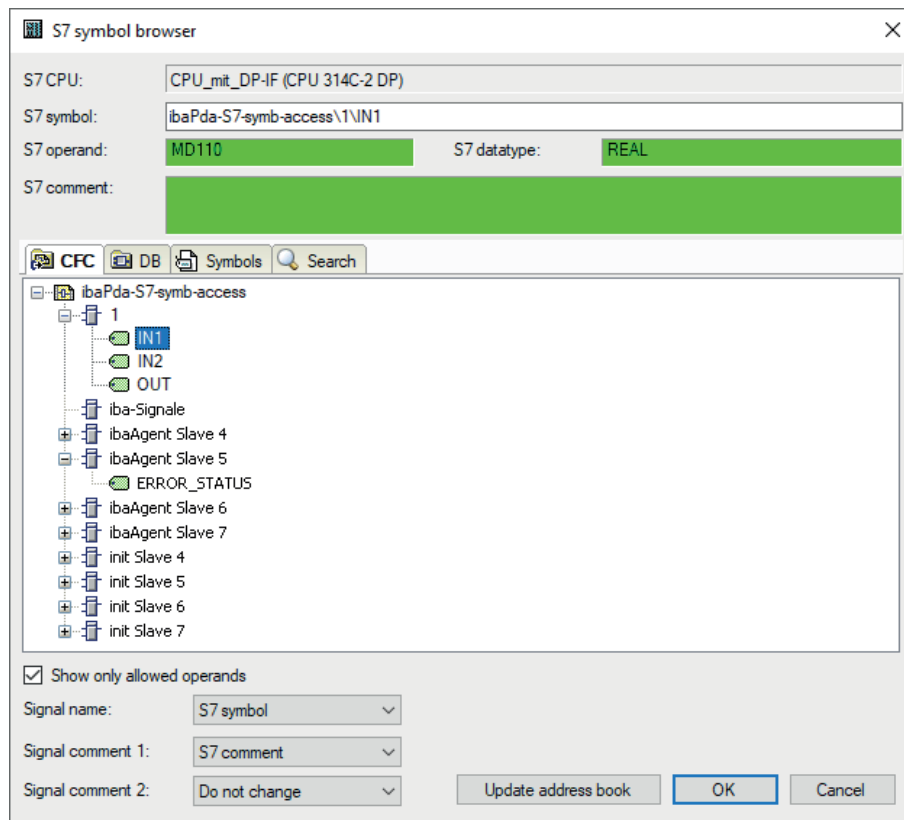
Right-click on the table header of the signal table.

- Select *Lookup missing symbols* to search only for the missing symbols.
- Select *Lookup all symbols* to search and replace all symbols. The command executes a backward resolution of the S7 symbols out of the S7 operands. *ibaPDA* searches the symbol table first, then CFC and finally the DBs for the operand.

3.3.5.3 Selection of CFC connectors

In order to select CFC connectors for the measurement, open the symbol browser first, see [➤ Selection via the symbolic operand addresses, page 41](#).

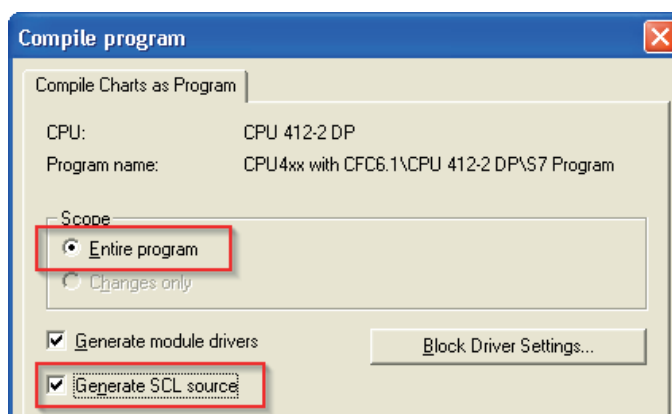
In the symbol browser, you open the *CFC* tab and select the signal. The connectors are listed hierarchically by chart name, module name and connector name:



Note



If no connectors are shown in the CFC tab, the SCL sources might not have been translated in the STEP 7 project. Activate the following options in the dialog for compiling the program in the SIMATIC software:



Afterwards, generate the address books again.

Note

When compiling a CFC program, DB addresses are assigned automatically in STEP 7 to the connectors. It may happen that other DB addresses are assigned to connectors, depending on the scope of the program changes that have been made between two compilation runs.

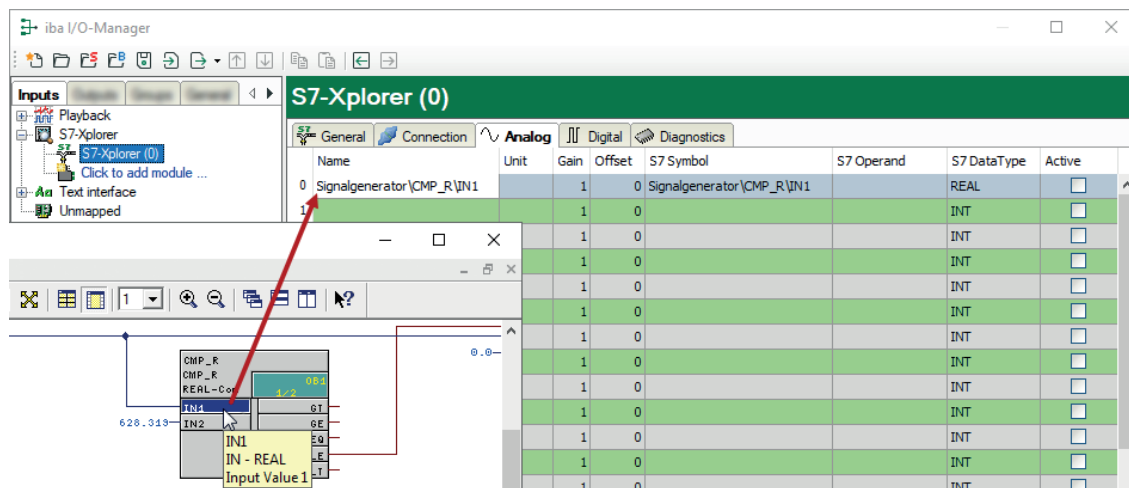
In this case, also the address book for *ibaPDA* has to be generated again. The symbolically configured signals are automatically checked in the I/O Manager whereas the related absolute S7 operands are updated.

Special function Drag & Drop

A convenient way to select signals is to drag & drop them from the CFC chart into the I/O Manager of *ibaPDA*.

1. Open the I/O Manager of *ibaPDA* and the signal table of the required module.
2. Start the CFC editor on the same computer as the *ibaPDA* client.
3. Drag the connector from the CFC editor into the desired line of the signal table in the I/O Manager of *ibaPDA*.

→ The CFC connector is now a measured signal in the signal table of the module.

**3.3.6 Module S7 Request**

With the *S7 Request* module, you can acquire up to 1024 analog signals and 1024 digital signals. A maximum of 1466 Byte is possible (max. length of the user data of an UDP telegram).

Configure a separate Request block call for each module.

For more information about the module settings, see ➤ *General module settings*, page 29.

3.3.7 Module S7 UDP Request Decoder

With the *S7 UDP Request Decoder* module, you can acquire up to 11728 digital signals, which are sent in form of a max. of 733 words (1466 Byte).

General tab

For more information on the module settings, see ➤ *General module settings*, page 29.

Module specific settings

Module layout – No. of decoders

Define the number of configurable decoders in the table of digital signals. The default value is 32. The maximum value is 733. The signal tables are adjusted accordingly.

Connection configuration

Configure the connection of the *S7 UDP Request Decoder* module in the same way as the connection for an S7-Request module, see ➤ *Connection settings*, page 30.

Digital tab

The declaration of the digital signals is done in two steps.

- First define the words (source signals), which are broken down for the digital signals (bits).
You can directly enter the words as basic signals for decoding via the absolute S7 operands. Only word operands (e. g. PIW, MW, DBW) are allowed.

You can also use S7 symbols by generating address books. For detailed information, see ➤ *Selection via the symbolic operand addresses*, page 41. The signals selected in the CFC and symbol browser are applied and the columns *Name*, *S7 symbol*, *S7 Operand* and *Data Type* are filled in automatically.

- You can open each word (source signal) via the <+> button to display the list of associated digital signals.

Then define the individual digital signals (bits) of the source signal.

S7 UDP Request Decoder (4)			
General Connection Digital Diagnostics			
Decoder		S7 Operand	Active
0	+ PEW 1	PIW 1	<input checked="" type="checkbox"/>
1	+ DB 3.DBW 2	DB 3.DBW 2	<input checked="" type="checkbox"/>
2	- DB 3.DBW 4	DB 3.DBW 4	<input checked="" type="checkbox"/>
Name			Active
Digital Signal 0			<input checked="" type="checkbox"/>
Digital Signal 1			<input checked="" type="checkbox"/>
Digital Signal 2			<input checked="" type="checkbox"/>
Digital Signal 3			<input checked="" type="checkbox"/>
Digital Signal 4			<input checked="" type="checkbox"/>
Digital Signal 5			<input checked="" type="checkbox"/>
Digital Signal 6			<input checked="" type="checkbox"/>
Digital Signal 7			<input checked="" type="checkbox"/>
Digital Signal 8			<input checked="" type="checkbox"/>
Digital Signal 9			<input checked="" type="checkbox"/>
Digital Signal 10			<input checked="" type="checkbox"/>
Digital Signal 11			<input checked="" type="checkbox"/>
Digital Signal 12			<input checked="" type="checkbox"/>
Digital Signal 13			<input checked="" type="checkbox"/>
Digital Signal 14			<input checked="" type="checkbox"/>
Digital Signal 15			<input checked="" type="checkbox"/>
3	+ DB 3.DBW 6	DB 3.DBW 6	<input checked="" type="checkbox"/>

The individual columns of the signal table have the following meanings:

Source signal

Decoder

Enter a name for the source signal.

S7 Operand/S7 Symbol

Enter the S7 operand to which the signal is assigned and, if applicable, the S7 symbol.

DataType

Enter the data type of the signal. The data type also determines the number of digital signals. *ibaPDA* automatically derives the possible data type from the S7 operand or S7 symbol.

Active

If you activate the source signal, it is acquired with all digital signals. You can deactivate individual digital signals.

Individual digital signals (bits)**Name**

Enter a name for the individual digital signals.

Active

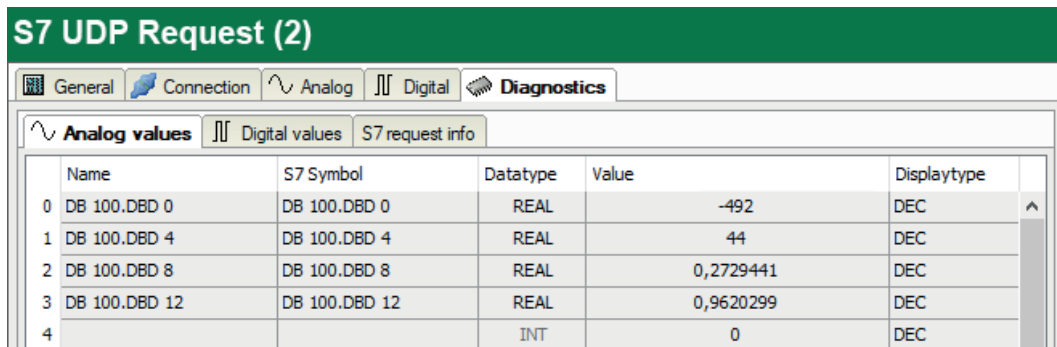
If you activate the signal, the signal is acquired and is also considered when checking the number of licensed signals.

Note

ibaPDA only takes the activated digital signals into account for the number of licensed signals, i.e. no additional signal for the source signal.

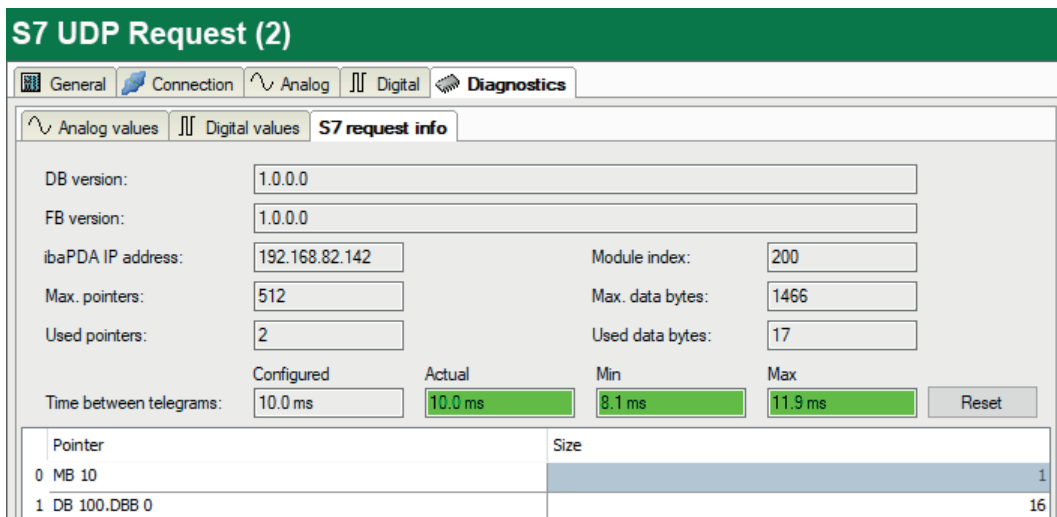
3.3.8 Module diagnostics

In the *Diagnostics* tab on the subtabs *Analog values* and *Digital values*, you can view all configured operands in tabular form with their data type and actual value.



	Name	S7 Symbol	Datatype	Value	Displaytype
0	DB 100.DB0 0	DB 100.DB0 0	REAL	-492	DEC
1	DB 100.DB0 4	DB 100.DB0 4	REAL	44	DEC
2	DB 100.DB0 8	DB 100.DB0 8	REAL	0,2729441	DEC
3	DB 100.DB0 12	DB 100.DB0 12	REAL	0,9620299	DEC
4			INT	0	DEC

On the *S7 request info* subtab, you can view the data sent to the S7-CPU, and the data that are reported back as well as the general diagnostics data.



S7 UDP Request (2)

General Connection Analog Digital **Diagnostics**

Analog values Digital values **S7 request info**

DB version: 1.0.0.0

FB version: 1.0.0.0

ibaPDA IP address: 192.168.82.142

Module index: 200

Max. pointers: 512

Max. data bytes: 1466

Used pointers: 2

Used data bytes: 17

Time between telegrams: Configured 10.0 ms Actual 10.0 ms Min 8.1 ms Max 11.9 ms [Reset]

Pointer	Size
0 MB 10	1
1 DB 100.DBB 0	16

DB version

Version of the data block used in the CPU

FB version

Version of the function block used in the CPU

ibaPDA IP address

IP address of the *ibaPDA* computer sent to the S7-CPU.

Module index

Module index sent to the S7-CPU (see also [General module settings](#), page 29)

Max. pointers

The max. number of used pointers (depends on the size of the data block *ibaREQ_DB*).

Used pointers

Currently used number of pointers.

Max. data bytes

Max. size of the user data in the data telegrams to *ibaPDA*

Used data bytes

Currently used bytes in the user data of the data telegrams.

Time between telegrams:

Configured: Corresponds to the setting *Timebase* on the *General* tab

Actual: Time between the two last received telegrams

Min: shortest time

Max: longest time

The background color of the values *Actual*, *Min* and *Max* provides additional information:

Color	Meaning
Green	The time span between two telegrams is shorter than the double of the defined timebase.
Orange	The time span between two telegrams is higher or equals the double of the defined timebase.

<Reset>

Resetting the Min and Max values

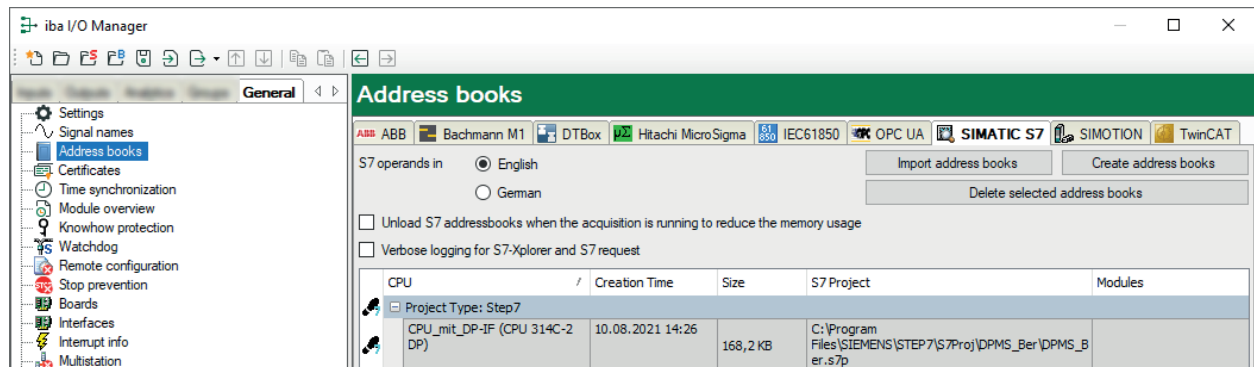
Pointer table

Currently required data pointer with address and length

For optimizing the communication performance, signals with consecutive addresses are requested and transferred as a block (pointer).

3.3.9 Address books

The address books for SIMATIC S7 controllers are created and managed across modules. You can use one address book in more than one module.



There are different types of address books for the different S7 project types:

- STEP 7: SIMATIC Manager project
(not for S7-Xplorer modules with connection mode TCP/IP S7-1x00)
- TIA Portal: TIA Portal Project

S7 operands in English/German

Here you can choose the language in which the S7 operands will later be available when browsing through the signal tables.

<Create address books>

This button opens the "S7 address book generator" dialog. You can select the source directory of an S7 project for creating the S7 address book. This can be a local or network drive.

<Import address books>

Import address books which are already available as ZIP files.

<Delete selected address books>

Delete address books from the *ibaPDA* server's directory.

Unload S7 address book when the acquisition is running to reduce the memory usage

By enabling this option, the address book is outsourced to the hard disk during the acquisition in order to free up the main memory for the acquisition.

Table

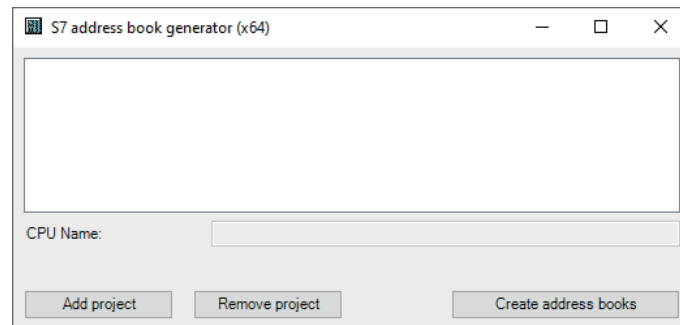
List of all address books that are currently available in the system with name, creation date, path of STEP 7 project or IP address of the CPU in case of online generated address books and location where the address book is used.

3.3.9.1 Creating address books offline from S7 project

For creating an address book, the S7 project has to be available. For the subsequent use, this is not necessary.

You create an address book using the S7 address book generator.

S7 address book generator



CPU Name

Name of the CPU

Step 7 HW Config export

A HW config export file can be selected as an option (useful when using an iba bus monitor in sniffer mode)

Comment language

Selection of the language that is to be imported for comment texts (only available for SIMATIC TIA portal projects)

<Add project>

Adds a new project to the list

Remove project>

Removes the marked project from the list

<Create address books>

Creates address books from the selected projects

Note

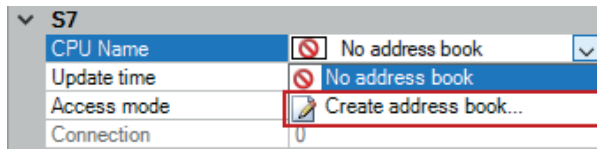


You can overwrite the entry in the *CPU Name* field. Thus, you can assign a unique name for the CPU that differs from that in the STEP 7 project. This is especially interesting when you use several STEP 7 projects in which the CPUs have the same name.

Creating address books via the S7 address book generator

1. Open the S7 address book generator by one of the following ways:

- On the *General* tab *Address books* node via the <Create address books> button
- In the module configuration on the *General* tab under *S7 – CPU Name*: Select *Create address book* in the drop-down menu.

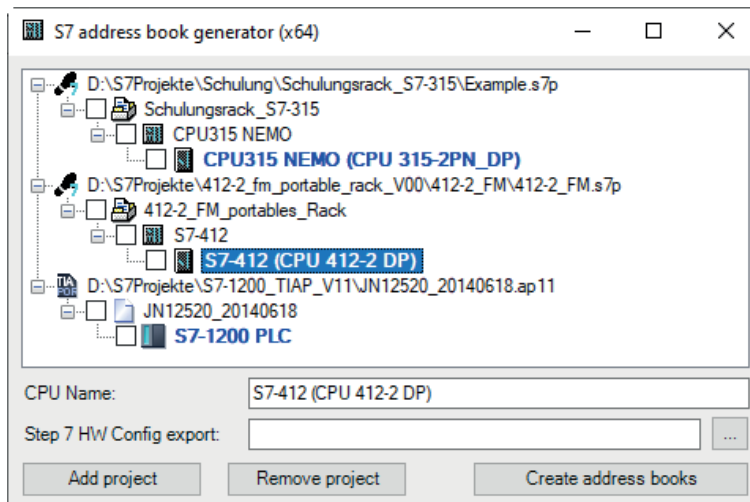


2. Click on the <Add project>.

3. Select a project file in the file browser

→ Now, the STEP 7 project with all configured CPUs is displayed.

4. Mark the CPUs you want to create the address books from and click on <Create address books>.



Note



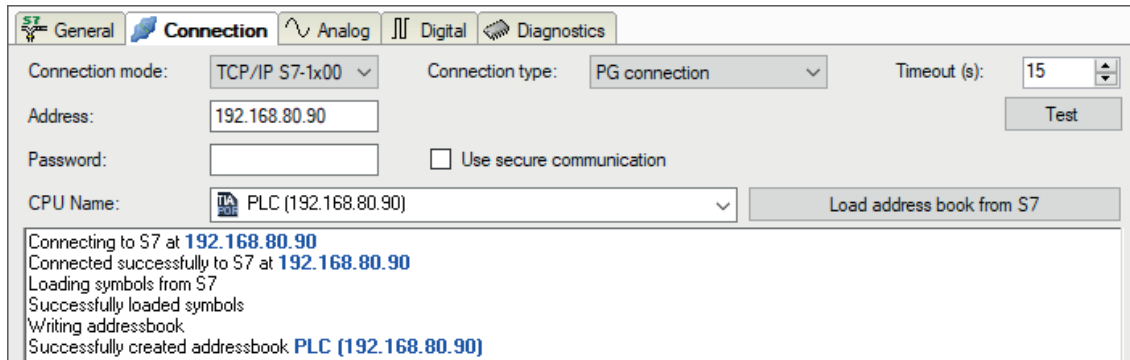
When address books of these projects are generated, TIA Portal projects must be compiled, saved and closed.

3.3.9.2 Creating address books online from S7-1200/1500 CPU

You can create online address books out of an S7-1200 or S7-1500 CPU if connection mode *TCP/IP S7-1x00* has been selected. The address data are read directly from the CPU. Accessing the S7 project is not necessary.

Click on <Load address book from S7> to load the address book.

The CPU name of the address book is given automatically.



Address books that have been created online also contain operand type address information and hence can be used in connection mode TCP/IP as well. Just change the connection mode after creating the address book.

4 Description of Request blocks

4.1 ibaREQ iba block family

These blocks initialize and control communication between *ibaPDA* and the *S7* controller.

The ibaREQ iba block family allows access exclusively to non-optimized data blocks. Addressing is carried out using the operand address.

One set of Request blocks has to be called for each Request module (connection). The used blocks are part of the iba *S7* library (see chapter [↗ iba S7 library](#), page 107).

Note



Only use Request blocks from the latest iba *S7* library!

Request blocks in application examples can be outdated and, thus, cause errors.

For S7-300/S7-400

Use different Request block combinations depending on the current system configuration:

Request block	CPU with integrated PN interface or WinAC RTX	S7-300 CPU + CP343-1	S7-400 CPU + CP443-1	recommended call level
ibaREQ_M (FB140)	X	X	X	OB1
ibaREQ_UDPact (FB145)	X	X	X	OB3x
ibaREQ_UDPint (FB146)	X	-	-	OB3x
ibaREQ_UDPext3 (FB147)	-	X	-	OB3x
ibaREQ_UDPext4 (FB148)	-	-	X	OB3x
ibaREQ_DB (DB15)	X	X	X	-
ibaUDT_UDPact (UDT145)	X	X	X	-

Use always the following blocks:

■ ibaREQ_M (Management)

The block realizes the communication with *ibaPDA*. Ideally, the block is called in OB1. This block has to be called separately in every system configuration for each module in ibaPDA.

- **ibaREQ_UDPact** (provision of current signal values)
The block provides the current signal values using the transmission cycle. The block is called internally in the blocks *ibaREQ_UDPint*, *ibaREQ_UDPext3*, or *ibaREQ_UDPext4*. The block must therefore always be present in the project, but does not have to be called separately.
- **ibaREQ_DB** (interface DB)
This DB acts as an interface to *ibaPDA* and between the different Request blocks.

Use different Request blocks depending on the current S7 system configuration:

- **ibaREQ_UDPint**
The block sends the current signal values provided via an integrated PN interface.
- **ibaREQ_UDPext3**
The block sends the current signal values provided via an external communication processor CP343-1.
- **ibaREQ_UDPext4**
The block sends the current signal values provided via an external communication processor CP443-1.

Always use the blocks *ibaREQ_UDPint*, *ibaREQ_UDPext3* and *ibaREQ_UDPext4* alternatively.

For S7-1500

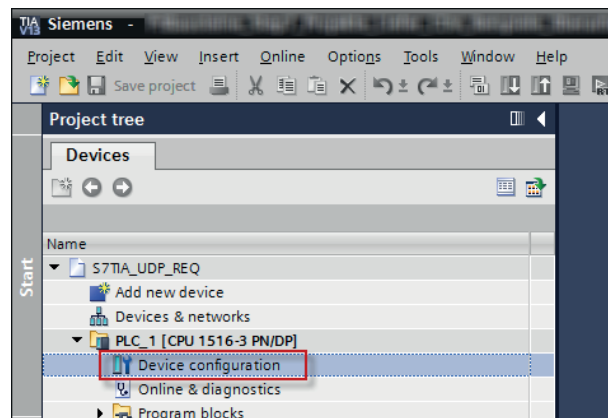
Use different Request block combinations depending on the current system configuration:

Request block	S7-1500 CPU with integrated PN interface	recommended call up level
<i>ibaREQ_M</i> (FB1400)	X	OB1
<i>ibaREQ_UDP2</i> (FB1406)	X	OB3x
<i>ibaREQ_UDPact</i> (FB 1410)	X	-
<i>ibaREQ_DB</i> (DB15)	X	-
<i>ibaREQ_DB-Interface</i>	X	-

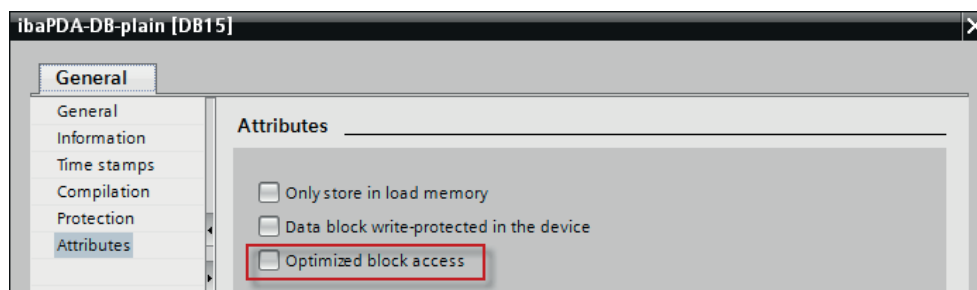
- **ibaREQ_M** (Management)
The block realizes the communication with *ibaPDA*. Ideally, the block is called in OB1.
- **ibaREQ_UDP2** (provision and sending of current signal values)
The block provides the current signal values using the transmission cycle. Ideally, the block is called in a cyclic interrupt OB.
- **ibaREQ_UDPact**
The block is used internally by *ibaREQ_UDP2*.
- **ibaREQ_DB** (interface DB)
This DB acts as an interface to *ibaPDA* and between the different Request blocks.

4.1.1 Device configuration

Make the following setting in CPU device configuration:



Under *ibaREQ_DB (DB15) block properties – Attributes* disable the *Optimized block access* option.



Note

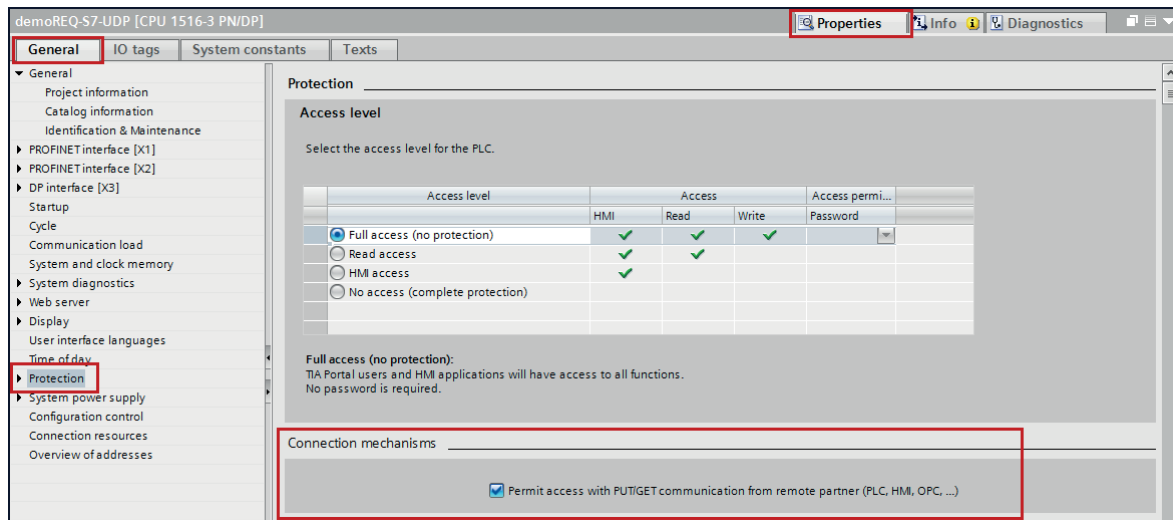


The configuration of the UDP connection is carried out program-controlled in the Request block. It is not allowed to configure manually a UDP connection and use it for this purpose.

S7-1500 CPUs with TCP/IP connection mode (not TCP/IP S7-1x00)

Up to TIA Portal V18, make the following settings in the TIA Portal:

In the CPU properties (*Properties – General – Protection – Connection mechanisms*), activate the access via PUT/GET communication.



For the S7-1200, this option is only available firmware V4.0 or higher.

As of TIA Portal V19 and CPU firmware V3.1 (V4.7 for S7-1200), activate the PUT/GET access for S7-1500 and S7-1200 CPUs as follows:

1. In the project navigation, navigate to *Security settings – Users and roles – Roles* tab.
2. Add a new role and enter a name, e.g. "Put/Get".
3. In the *Runtime rights* tab, select your PLC under *Function rights categories*.
4. Under *Function rights*, activate the access level *HMI access*.
5. Under *Users and Roles*, go to the *Users* tab.
6. Activate the user "Anonymous".
7. Confirm the following message with <OK>.
8. Under *Assigned roles*, assign the newly created role to the anonymous user.
9. In the *Device view*, open the properties of the CPU.
10. In the section navigation, navigate to *Protection & Security – Connection mechanisms*.
11. Activate the option *Permit access with PUT/GET communication from remote partner*.
12. Save and compile the configuration and load the changes into the CPU.

You can find further information in the SiePortal under

<https://support.industry.siemens.com/cs/ww/en/view/109925755>.

S7-1500 access protection

You can enable access protection on S7-1500 CPUs. There is the following dependency with *ibaPDA*:

Access level	CPU access	ibaPDA reads symbolic from CPU	S7 access
Full access (no protection)	HMI, read, write	OK	OK
Read access	HMI, read	OK	OK
HMI access	HMI	No	OK
No access (complete protection)	-	No	No

4.1.2 ibaREQ_M (FB140)

Description of the formal parameters

Name	Type	Data Type	Description
REQ_DB	IN	BLOCK_DB	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
RESET	IN	BOOL	FALSE: No reset (default) TRUE: Block reset
INP_RANGE	IN	INT	Number of input bytes (evaluation only during initialization), 0: automatic detection (recommended)
OUT_RANGE	IN	INT	Number of output bytes (evaluation only during initialization), 0: automatic detection (recommended)
MARKER_RANGE	IN	INT	Number of marker bytes (evaluation only during initialization), 0: automatic detection (recommended)
ERROR_STATUS	OUT	DWORD	Internal error code

The following SFCs are used internally:

- SFC 20 (BLKMOV)
- SFC 21 (FILL)
- SFC 24 (TEST_DB)
- SFC 51 (RDSYSST)

Detailed description

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

Length:

5280 Bytes: up to 128 Pointers (min.)

9120 Bytes: up to 512 Pointers

14240 Bytes: up to 1024 Pointers (max.)

Any other length within these limits is permitted. The number of usable pointers are evaluated according to the length.

RESET

Used to manually reset the Request blocks. All Request blocks of a combination are automatically reset together. Usually, the parameter does not have to be connected.

INP_RANGE

Limits the number of input bytes to be measured.

If INP_RANGE = 0, the size of the available processor image of the inputs is determined by the Request FB itself (recommended). Evaluation is done only during the initialization phase of the module.

OUT_RANGE

Limits the number of output bytes to be measured.

With OUT_RANGE = 0, the size of the available processor image of the outputs is determined by the Request FB itself (recommended). Evaluation is done only during the initialization phase of the module

MARKER_RANGE

Limits the number of marker bytes to be measured.

With MARKER_RANGE = 0, the number of available markers is determined by the Request FB itself (recommended). Evaluation is done only during the initialization phase of the module.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

Error codes

Decimal value ERROR_STATUS	Description
1	datablock ibaREQ_DB is write protected
2	datablock ibaREQ_DB invalid (DB = 0 or > limit of CPU)
3	datablock ibaREQ_DB does not exist
4	datablock ibaREQ_DB undefined error
5	datablock ibaREQ_DB too short
6	datablock ibaREQ_DB too short for ibaREQ_UDP
9	internal error (RD_SINFO)
10	no access to datablock ibaREQ_DB (read)
11	no access to datablock ibaREQ_DB (write)
20	initialization not finished

Decimal value ERROR_STATUS	Description
21	insufficient memory for SZL
22	wrong SZL_ID
23	wrong or invalid index of SZL
24	error while reading I&M data from CPU
25	error while reading PLC data
31	initialization canceled with error
32	initialization not completed
41	too many pointers (ibaREQ_DB too small)
42	too many pointers in one command (>128)
44	invalid command id
45	operand invalid (not defined)
46	operand invalid (datatype)
47	operand invalid (memory area)
200	no connection to PN device / DP slave
300	version of ibaREQ_UDPack does not match with ibaREQ_M (ID)
301	version of ibaREQ_UDPack does not match with ibaREQ_M (FB)
302	version of ibaREQ_UDPack does not match with ibaREQ_M (DB)
303	type of transmit agent does not match with configured request type in <i>ibaPDA</i>
305	PROFIBUS DP slave hardware configuration is invalid
306	configured peripheral address is invalid
310	no access to datablock ibaREQ_DB (read)
311	no access to datablock ibaREQ_DB (write)
315	error while masking of synchronous faults
316	error while demasking of synchronous faults
320	operand invalid (datatype)
321	operand invalid (pointer)
401	ADR_SLOT/ADR_SLOT_0 invalid hw-id
402	ADR_SLOT/ADR_SLOT_0 invalid hw-id, no IO-Device or DP-Slave
403	ADR_SLOT/ADR_SLOT_0 invalid hw-id, is no PROFIBUS or PROFINET
406	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
407	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
409	ADR_SLOT_1 invalid configuration slot 1
410	no connection to PN device/DP-Slave or error
411	ADR_SLOT_1 invalid hw-id
412	ADR_SLOT_1 invalid hw-id, no IO-Device or DP-Slave
413	ADR_SLOT_1 invalid hw-id, is no PROFIBUS
416	ADR_SLOT_1 invalid configuration slot 1
0x8yyy	error code of inner TUSEND/AG_SEND/AG_LSEND

4.1.3 ibaREQ_M (FB1400)

Description of formal parameters

Name	Type	Data Type	Description
REQ_DB	IN	DB_ANY	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
RESET	IN	BOOL	FALSE: No reset (default) TRUE: Block reset
CPU_HW_ID	IN	HW_IO	Hardware ID of local CPU
ERROR_STATUS	OUT	WORD	Error code

The following SIMATIC standard block is used internally:

■ GET_IM_DATA (FB801)

Detailed description

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

The length of the data block is fixed.

RESET

Used to manually reset the Request blocks. All Request blocks of a combination are automatically reset together. Usually, the parameter does not have to be connected.

CPU_HW_ID

TIA Portal system constant, which refers to the corresponding CPU.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

Error codes

Value ERROR_STATUS		Description
Hexadecimal	Decimal	
0x01		1 datablock ibaREQ_DB is write protected
0x02	2	datablock ibaREQ_DB invalid (DB = 0 or > limit of CPU)
0x03	3	datablock ibaREQ_DB does not exist
0x04	4	datablock ibaREQ_DB undefined error
0x05	5	datablock ibaREQ_DB too short
0x09	9	internal error (RD_SINFO)
0x01A	0	no access to datablock ibaREQ_DB (read)
0x0B	11	no access to datablock ibaREQ_DB (write)
0x14	20	initialization not finished
0x15	21	insufficient memory for SZL
0x16	22	wrong SZL_ID
0x17	23	wrong or invalid index of SZL
0x18	24	error while reading I&M data from CPU
0x19	25	error while reading plc data
0x1D	29	reset not finished
0x1F	31	initialization canceled with error
0x20	32	initialization not completed
0x29	41	too many pointers (>1024)
0x30	42	too many pointers in one command (>128)
0x31	44	invalid command id
0x32	45	operand invalid (not defined)
0x33	46	operand invalid (datatype)
0x34	47	operand invalid (memory area)

4.1.4 ibaREQ_UDPact (FB145)

This block is called internally by the blocks ibaREQ_UDPint, ibaREQ_UDPext3, and ibaREQ_UDPext4, so you do not need to include it manually.

Description of the formal parameters

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
Xchange	INOUT	UDT145	Interface for the calling block
ERROR_STATUS	OUT	WORD	Internal error code

The following SFCs are used internally

- SFC 20 (BLKMOV)
- SFC 21 (FILL)
- SFC 36 (MSK_FLT)
- SFC 37 (DMSK_FLT)

Detailed description

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

Xchange

Via the parameterized data range, the data are exchanged with the calling block.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

Error codes

Decimal value ERROR_STATUS	Description
1	datablock ibaREQ_DB is write protected
2	datablock ibaREQ_DB invalid (DB = 0 or > limit of CPU)
3	datablock ibaREQ_DB does not exist
4	datablock ibaREQ_DB undefined error
5	datablock ibaREQ_DB too short
6	datablock ibaREQ_DB too short for ibaREQ_UDP
9	internal error (RD_SINFO)
10	no access to datablock ibaREQ_DB (read)
11	no access to datablock ibaREQ_DB (write)
20	initialization not finished
21	insufficient memory for SZL

Decimal value ERROR_STATUS	Description
22	wrong SZL_ID
23	wrong or invalid index of SZL
24	error while reading I&M data from CPU
25	error while reading PLC data
31	initialization canceled with error
32	initialization not completed
41	too many pointers (ibaREQ_DB too small)
42	too many pointers in one command (>128)
44	invalid command id
45	operand invalid (not defined)
46	operand invalid (datatype)
47	operand invalid (memory area)
200	no connection to PN device / DP slave
300	version of ibaREQ_UDPack does not match with ibaREQ_M (ID)
301	version of ibaREQ_UDPack does not match with ibaREQ_M (FB)
302	version of ibaREQ_UDPack does not match with ibaREQ_M (DB)
303	type of transmit agent does not match with configured request type in <i>ibaPDA</i>
305	PROFIBUS DP slave hardware configuration is invalid
306	configured peripheral address is invalid
310	no access to datablock ibaREQ_DB (read)
311	no access to datablock ibaREQ_DB (write)
315	error while masking of synchronous faults
316	error while demasking of synchronous faults
320	operand invalid (datatype)
321	operand invalid (pointer)
401	ADR_SLOT/ADR_SLOT_0 invalid hw-id
402	ADR_SLOT/ADR_SLOT_0 invalid hw-id, no IO-Device or DP-Slave
403	ADR_SLOT/ADR_SLOT_0 invalid hw-id, is no PROFIBUS or PROFINET
406	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
407	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
409	ADR_SLOT_1 invalid configuration slot 1
410	no connection to PN device/DP-Slave or error
411	ADR_SLOT_1 invalid hw-id
412	ADR_SLOT_1 invalid hw-id, no IO-Device or DP-Slave
413	ADR_SLOT_1 invalid hw-id, is no PROFIBUS
416	ADR_SLOT_1 invalid configuration slot 1
0x8yyy	error code of inner TUSEND/AG_SEND/AG_LSEND

4.1.5 ibaREQ_UDPact (FB1410)

This block is called internally by the blocks ibaREQ_UDPint, ibaREQ_UDPext3, and ibaREQ_UDPext4, so you do not need to include it manually.

Description of the formal parameters

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
Xchange	INOUT	UDT145	Interface for the calling block
ERROR_STATUS	OUT	WORD	Internal error code

Detailed description

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

Xchange

Via the parameterized data range, the data are exchanged with the calling block.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

Error codes

Decimal value ERROR_STATUS	Description
1	datablock ibaREQ_DB is write protected
2	datablock ibaREQ_DB invalid (DB = 0 or > limit of CPU)
3	datablock ibaREQ_DB does not exist
4	datablock ibaREQ_DB undefined error
5	datablock ibaREQ_DB too short
6	datablock ibaREQ_DB too short for ibaREQ_UDP
9	internal error (RD_SINFO)
10	no access to datablock ibaREQ_DB (read)
11	no access to datablock ibaREQ_DB (write)
20	initialization not finished
21	insufficient memory for SZL
22	wrong SZL_ID
23	wrong or invalid index of SZL
24	error while reading I&M data from CPU
25	error while reading PLC data
31	initialization canceled with error
32	initialization not completed

Decimal value ERROR_STATUS	Description
41	too many pointers (ibaREQ_DB too small)
42	too many pointers in one command (>128)
44	invalid command id
45	operand invalid (not defined)
46	operand invalid (datatype)
47	operand invalid (memory area)
200	no connection to PN device / DP slave
300	version of ibaREQ_UDPack does not match with ibaREQ_M (ID)
301	version of ibaREQ_UDPack does not match with ibaREQ_M (FB)
302	version of ibaREQ_UDPack does not match with ibaREQ_M (DB)
303	type of transmit agent does not match with configured request type in <i>ibaPDA</i>
305	PROFIBUS DP slave hardware configuration is invalid
306	configured peripheral address is invalid
310	no access to datablock ibaREQ_DB (read)
311	no access to datablock ibaREQ_DB (write)
315	error while masking of synchronous faults
316	error while demasking of synchronous faults
320	operand invalid (datatype)
321	operand invalid (pointer)
401	ADR_SLOT/ADR_SLOT_0 invalid hw-id
402	ADR_SLOT/ADR_SLOT_0 invalid hw-id, no IO-Device or DP-Slave
403	ADR_SLOT/ADR_SLOT_0 invalid hw-id, is no PROFIBUS or PROFINET
406	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
407	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
409	ADR_SLOT_1 invalid configuration slot 1
410	no connection to PN device/DP-Slave or error
411	ADR_SLOT_1 invalid hw-id
412	ADR_SLOT_1 invalid hw-id, no IO-Device or DP-Slave
413	ADR_SLOT_1 invalid hw-id, is no PROFIBUS
416	ADR_SLOT_1 invalid configuration slot 1
0x8yyy	error code of inner TUSEND/AG_SEND/AG_LSEND

4.1.6 ibaREQ_UDPint (FB146)

Description of the formal parameters

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
CON_ID	IN	INT	Unique connection ID of the sending block (TUSEND)
LOCAL_DEVICE_ID	IN	BYTE	Device ID of the sending block (TUSEND)
LOCAL_PORT	IN	DINT	Used local port number
RESET_CON	IN	BOOL	FALSE: no reset (standard) TRUE: reset of the communication connection
ERROR_STATUS	OUT	WORD	Internal error code
ERROR_TCON	OUT	WORD	Error code connection setup of the TCON function block
COUNT_TCON	OUT	WORD	Counter attempts for connecting
ERROR_TSEND	OUT	WORD	Error code of the sending block TUSEND
COUNT_TSEND	OUT	WORD	Counter sent telegrams
LOST_SAMPLES	OUT	WORD	Counter for lost measurement values

The following SFCs are used internally:

- FB145 (ibaREQ_UDPact)
- FB 65 (TCON)
- FB 66 (TDISCON)
- FB 67 (TUSEND)
- SFB 4 (TON)
- UDT 65 (TCON_PAR)
- UDT 66 (TADDR_PAR)
- UDT 145 (ibaUDT_UDPact)

Detailed description

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

CON_ID

Unique reference to the connection to be established. Corresponds to the parameter ID of the Siemens standard block TCON.

LOCAL_DEVICE_ID

ID of the used interface. Corresponds to the parameter local_device_id in the CONNECT structure of the Siemens standard block TCON.

Hexadecimal value	Description
B#16#01	Communication via IE interface in interface slot 1 (IF1) with WinAC RTX or an IM 151-8 PN/DP CPU.
B#16#02	Communication through the integrated PROFINET interface of the CPU31x-2 PN/DP, CPU314C-2 PN/DP and IM154-8 CPU.
B#16#03	Communication through the integrated PROFINET interface of the CPU319-3 PN/DP, CPU315T-3 PN/DP, CPU317T-3 PN/DP, CPU317TF-3PN/DP.
B#16#04	Communication through SINUMERIK NCU7x0.2 PN with CPU319-3 PN/DP and SINUMERIK NCU7x0.3PN with CPU317-2 PN/DP.
B#16#05	Communication through the integrated PROFINET interface of the CPU412-2 PN, CPU414-3 PN/DP, CPU416-3 PN/DP, CPU412-5H PN/DP (Rack 0), CPU414-5H PN/DP (Rack 0), CPU416-5H PN/DP (Rack 0) and CPU417-5H PN/DP (Rack 0).
B#16#06	Communication via the IE interface in interface slot 2 (IF2) with WinAC RTX
B#16#08	Communication via the integrated PROFINET interface of the CPU410-5H (Rack 0)
B#16#0B	Communication via the IE interface in interface slot 3 (IF3) with WinAC RTX
B#16#0F	Communication via the IE interface in interface slot 4 (IF4) with WinAC RTX
B#16#15	Communication through the integrated PROFINET interface of the CPU412-5H PN/DP (Rack 1), CPU414-5H PN/DP (Rack 1), CPU416-5H PN/DP (Rack 1) and CPU417-5H PN/DP (Rack 1).
B#16#18	Communication via the integrated PROFINET interface of the CPU 410-5H (Rack 1)

Table 2: Valid values of the parameter LOCAL_DEVICE_ID

Other documentation

You find more information under the following link:

<https://support.industry.siemens.com/cs/ww/en/view/51339682>

LOCAL_PORT

Number of the locally used port.

RESET_CON

Used for manually resetting the communication connection.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

Decimal value ERROR_STATUS	Description
350	Parameter LOCAL_DEVICE_ID is invalid.
360	Parameter LOCAL_PORT is invalid.

ERROR_TCON

Error code of connection setup. The standard error codes for the TCON function block are output.

0X8yyy error code of TCON block is passed.

COUNT_TCON

Counter for the attempts of connection setup. An increasing value indicates problems when establishing the connection to the *ibaPDA* computer.

ERROR_TSEND

Error code when sending. The standard error codes of the TSEND block are output.

0X8yyy error code of TSEND block is passed.

COUNT_TSEND

Counter for the sent telegrams. Usually, the counter is incremented continuously.

LOST_SAMPLES

The counter is incremented with every call of the block if no new UDP telegram can be sent to *ibaPDA*, as the previous send order has not been finished, yet. A continuously rising value indicates a shortage in the communication performance.

Error codes

Hexadecimal value ERROR_TCON	Description
W#16#8086	The ID parameter is outside the valid range.
W#16#8087	Maximum number of connections reached; no additional connection possible.
W#16#8089	The CONNECT parameter does not point to a data block.
W#16#809A	The CONNECT parameter points to a field that does not match the length of the connection description (UDT65).
W#16#809B	The local_device_id in the connection description does not match the CPU.
W#16#80A0	Group error for error codes W#16#80A1 and W#16#80A2
W#16#80A1	Connection or port is already used by user.
W#16#80A2	Local or remote port is used by the system.
W#16#80A3	Attempt is being made to re-establish an existing connection.
W#16#80A4	IP address of the remote connection endpoint is invalid, or it matches the local IP address.

Hexadecimal value ERROR_TCON	Description
W#16#80A7	Communications error: You called TDISCON before TCON had completed. TDISCON must first complexly terminate the connection referenced by the ID.
W#16#80B2	The CONNECT parameter points to a data block that was generated with the keyword UNLINKED.
W#16#80B3	Inconsistent parameter assignment: Group error for the error codes W#16#80A0 to W#16#80A2, W#16#80A4, W#16#80B4 to W#16#80B9
W#16#80B5	Error in active_est parameter (UDT 65) in the UDP protocol variant
W#16#80B6	Parameter assignment error relating to the connection_type parameter (UDT 65)
W#16#80B7	Error in one of the following parameters of UDT 65: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len
W#16#80B8	Parameter ID in the local connection description (UDT 65) and parameter ID are different.
W#16#80C3	All connection resources are in use.
W#16#80C4	Temporary communications error: <ul style="list-style-type: none"> ■ The connection cannot be established at this time. ■ The interface is receiving new parameters. ■ The configured connection is being removed by a TDISCON instruction. ■ The H system is connecting and updating.

Hexadecimal value ERROR_TSEND	Description
W#16#8085	LEN parameter has the value "0" or is greater than the highest valid value.
W#16#8086	The ID parameter is outside the valid address range.
W#16#8088	LEN parameter is greater than the memory area specified in DATA.
W#16#8089	ADDR parameter does not point to a data block.
W#16#80A1	Communication error: <ul style="list-style-type: none"> ■ The specified connection between user program and communication layer of the operating system has not yet been established. ■ The specified connection between the user program and the communication layer of the operating system is currently being terminated. Transfer over this connection is not possible. ■ The interface is being re-initialized.

Hexadecimal value ERROR_TSEND	Description
W#16#80A4	IP address of the remote connection endpoint is invalid, or it matches the local IP address.
W#16#80B3	The protocol variant (connection_type parameter in the connection description) is not set to UDP. Please use FB 63 TSEND. ADDR parameter: Invalid settings for port number.
W#16#80C3	<ul style="list-style-type: none">■ A block with this ID is already being processed in a different priority class.■ Internal lack of resources.
W#16#80C4	Temporary communications error: <ul style="list-style-type: none">■ The connection between the user program and the communication layer of the operating system cannot be established at this time.■ The interface is receiving new parameters.

Other documentation



For more information, refer the Siemens documentation about the TCON and TSEND blocks.

4.1.7 ibaREQ_UDPext3 (FB147)

Description of the formal parameters

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
ID	IN	INT	Connection ID of the connection configured in NetPro
HW_LADDR	IN	WORD	Module starting address of the CP
ERROR_STATUS	OUT	WORD	Internal error code
ERROR_SEND	OUT	WORD	Error code of the send block AG_SEND
COUNT_SEND	OUT	WORD	Telegram counter of the send block
LOST_SAMPLES	OUT	WORD	Counter for lost measurement values

The following SFCs are used internally:

- FB145 (ibaREQ_UDPact)
- FC 5 (AG_SEND)
- UDT 145 (ibaUDT_UDPact)

Detailed description

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

ID

Unique reference to the connection to be established. Has to match the ID used in NetPro.

HW_LADDR

Module starting address of the used CP. Has to match the LADDR used in NetPro.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

ERROR_SEND

Error code when sending The standard error codes of the AG_SEND block are issued.

0X8yyy error code of AG_SEND block is passed.

COUNT_SEND

Counter for the sent telegrams. Usually, the counter is incremented continuously.

LOST_SAMPLES

The counter is incremented with every call of the block if no new UDP telegram can be sent to *ibaPDA*, as the previous send order has not been finished, yet. A continuously rising value indicates a shortage in the communication performance.

4.1.8 ibaREQ_UDPext4 (FB148)

Description of the formal parameters

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
ID	IN	INT	Connection ID of the connection configured in NetPro
HW_LADDR	IN	WORD	Module starting address of the CP
ERROR_STATUS	OUT	WORD	Internal error code
ERROR_SEND	OUT	WORD	Error code of the send block AG_LSEND
COUNT_SEND	OUT	WORD	Telegram counter of the send block
LOST_SAMPLES	OUT	WORD	Counter for lost measurement values

The following SFCs are used internally:

- FB145 (ibaREQ_UDPact)
- FC 50 (AG_LSEND)
- UDT 145 (ibaUDT_UDPact)

Detailed description

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

ID

Unique reference to the connection to be established. Has to match the ID used in NetPro.

HW_LADDR

Module starting address of the used CP. Has to match the LADDR used in NetPro.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

ERROR_SEND

Error code when sending. The standard error codes of the AG_SEND block are issued.

The following error codes can be displayed:

0X8yyy error code of AG_LSEND block will be passed

COUNT_SEND

Counter for the sent telegrams. Usually, the counter is incremented continuously.

LOST_SAMPLES

The counter is incremented with every call of the block if no new UDP telegram can be sent to *ibaPDA*, as the previous send order has not been finished, yet. A continuously rising value indicates a shortage in the communication performance.

4.1.9 ibaREQ_UDP2 (FB1406)

Description of the formal parameters

Name	Type	Data type	Description
INTERFACE_ID	IN	HW_ANY	Hardware identifier of the used interface
CON_ID	IN	CONN_OUC	Unique connection ID of the send block (TSEND_C)
LOCAL_PORT	IN	UINT	Local port number
RESET_CON	IN	BOOL	FALSE: no reset (default) TRUE: reset of the communication connection
REQ_DB	INOUT	DB_ANY	DB of the <i>ibaPDA</i> communication interface ibaREQ_DB
ERROR_STATUS	OUT	WORD	Internal error code
LOST_SAMPLES	OUT	UNIT	Counter for lost measurement values

The following SIMATIC standard blocks are used internally:

- TCON
- TUSEND
- TDISCON

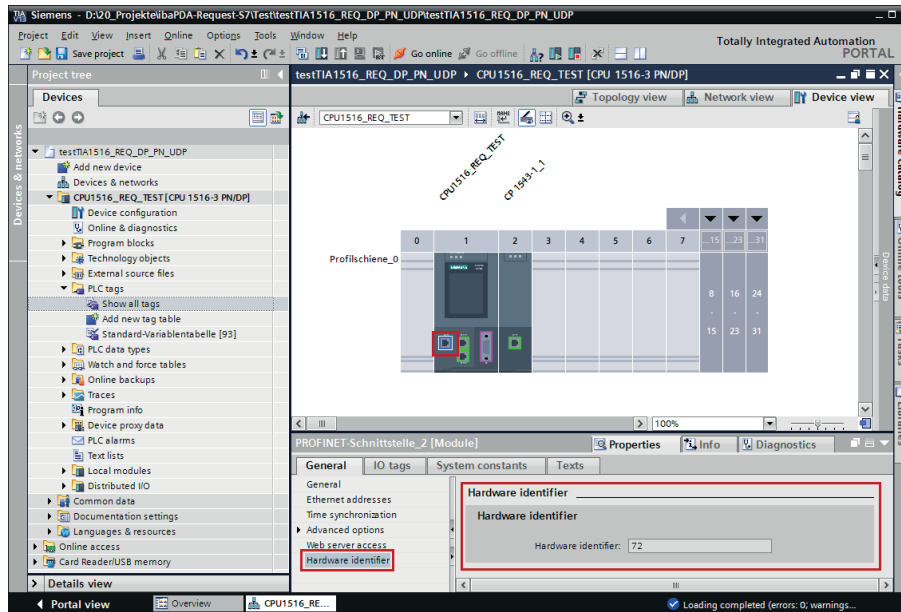
Detailed description

INTERFACE_ID

Hardware identifier of the used interface.

Tip

You find the hardware identifier of the marked interface under *Properties – General – Hardware identifier*.



The hardware identifier could be configured as a numerical value or as a system constant of the type *Hw_Interface*. You find the system constant under *Properties – System constants*. Always use the system constant of the interface and not of a port, or of the IO system.

PROFINET-Schnittstelle_2 [Module]				
Properties				
System constants				
Name	Type	Hardware identi.	Comment	
Local-PROFINET-Schnittstelle_2	Hw_Interface	72		
Local-PROFINET-Schnittstelle_2-Port_1	Hw_Interface	73		

CON_ID

Unique reference to the connection to be set up, value range: 1 to 4095.

LOCAL_PORT

Number of the locally used port

RESET_CON

Used for manually resetting the communication connection.

REQ_DB

The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

ERROR_STATUS

Internal error of the block. If there is no error, the value 0 is output.

LOST_SAMPLES

The counter is incremented with every call of the block if no new UDP telegram can be sent to *ibaPDA*, as the previous send order has not been finished, yet. A continuously rising value indicates a shortage in the communication performance.

Error codes

Hexadecimal value ERROR_TSEND	Description
W#16#80A1	The specified connection or the port is already used by user. Communication error: <ul style="list-style-type: none"> ■ The specified connection has not yet been established. ■ The specified connection is being terminated. ■ Transfer via this connection is not possible. ■ The interface is being re-initialized.
W#16#80A3	The nested T_DIAG instruction has reported that the connection has closed.
W#16#80A4	IP address of the remote connection endpoint is invalid, or it matches the IP address of the local partner.
W#16#80A7	Communication error: You called the instruction with COM_RST = 1 before the send job was complete.
W#16#80AA	A connection is currently being established with the same connection ID by another block. Repeat the job with a new rising edge at the REQ parameter.
W#16#80B6	Parameter assignment error in the connection_type parameter of the data block for connection description.
W#16#80B7	Error in one of the following parameters of the data block for connection description: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len.
W#16#8085	The LEN parameter is greater than the highest valid value.
W#16#8086	The ID parameter within the CONNECT parameter is outside the valid range.
W#16#8087	Maximum number of connections reached; no additional connection possible.
W#16#8088	The value at the LEN parameter does not correspond to the receive area set at the DATA parameter.
W#16#8091	Maximum nesting depth exceeded.
W#16#809A	The CONNECT parameter points to a field that does not match the length of the connection description.
W#16#809B	InterfaceID is invalid. It does not point to a local CPU interface or a CP or has the value "0".

³⁾ You must use the blocks at the same call level.

Hexadecimal value ERROR_TSEND	Description
W#16#80C3	<ul style="list-style-type: none">■ All connection resources are in use.■ A block with this ID is already being processed in a different priority group.
W#16#80C4	Temporary communication error: <ul style="list-style-type: none">■ The connection cannot be established at this time.■ The interface is receiving new parameters or the connection is being established.■ The configured connection is being removed by a TDISCON instruction.■ The used connection is being terminated by a call with COM_RST = 1.
W#16#80C6	Remote network error: The remote partner cannot be reached.

Other documentation



For more information, refer the Siemens documentation about the TSEND_C block.

4.2 ibaREQsym iba block family

These blocks initialize and control communication between *ibaPDA* and the S7 controller.

The ibaREQsym block family allows access to both optimized and non-optimized data blocks. Addressing is purely using the symbol names.

One set of Request blocks has to be called for each Request module (connection) in *ibaPDA*. The blocks used are part of the iba S7 library, see ↗ *iba S7 library*, page 107.

Note



Only use Request blocks from the latest iba S7 library!

Request blocks in application examples can be outdated and, thus, cause errors.

Use different Request block combinations depending on the current system configuration:

Request block	S7-1500 CPU with integrated PN interface	Recommended call level
ibaREQsym_M	X	OB1 or OB3x ³⁾
ibaREQsym_UDP	X	
ibaREQsym_DB_PDA	X	-
ibaREQsym-Interface	X	-

- ibaREQsym_M (Management)
The block realizes the communication with *ibaPDA*. Ideally, the block is called in OB1.
- ibaREQsym_UDP (sending of current signal values)
The block sends the current signal values to *ibaPDA*.
- ibaREQsym_DB_PDA (DB interfaces)
This DB acts as an interface to *ibaPDA* and between the different Request blocks.

4.2.1 ibaREQsym_M

Description of formal parameters

Name	Type	Data Type	Description
reset	IN	BOOL	FALSE: No reset (default) TRUE: Block reset
DB_PDA	INOUT	UDT	DB for the <i>ibaPDA</i> communication interface ibaREQ_DB_PDA
state	OUT	STRING[16]	Block status
errorStatusRun	OUT	WORD	Internal error code
errorStatus1	OUT	WORD	Internal error code
errorStatus2	OUT	WORD	Error code for internally called blocks

Detailed description

reset

Used to manually reset the block

DB_PDA

Pointer to the communication data range. This range is used for the data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

state

Block status in plain text

errorStatusRun

Internal error code for the block. If there is no error, the value 0 is output.

errorStatus1

Internal error code for the block. If there is no error, the value 0 is output.

errorStatus2

Internal error code for the block. If there is no error, the value 0 is output.

Error codes

Hexadecimal value errorStatus	Description
01x8001	invalid command id
01x800F	internal error (error moving data - pdainfo)
01x8010	internal error (error moving data - command)
01x8011	internal error (error moving data - response reset)
01x8012	internal error (error moving data - response diag)
01x8014	internal error (error moving data - response request)
01x801F	internal error (error moving data - response unknown)
01x8101	internal error (error moving data - response invalid command ID)
01x8202	internal error (error moving data - diag response data)

Hexadecimal value errorStatus	Description
01x8401	internal error (error moving data - request command data)
01x8402	internal error (error moving data - request response data)
01x8410	too many requested symbols in total (>2000)
01x8411	too many requested symbols in one command (>100)
01x8412	requested data length is too long (<max)
01x8413	requested data length is too short (<0)
01x8422	error moving symbol offsets
01x8423	error moving symbol names
01x8425	error resolving requested symbols
01x8426	error moving resolved symbols data
01x8512	no active symbols
01x8FFF	undefined internal error

4.2.2 ibaREQsym_UDP

Description of formal parameters

Name	Type	Data Type	Description
interfaceld	IN	HW_ANY	HW ID of the used interface
connectionId	IN	CONN_OUC	Unique connection ID of the send block (TSEND_C)
localPort	IN	UINT	Local port number
reset_com	IN	BOOL	FALSE: No reset (default) TRUE: Communication connection reset
DB_PDA	INOUT	DB_ANY	DB for the <i>ibaPDA</i> communication interface ibaREQsym_DB_PDA
state	OUT	STRING[16]	Block status
errorTsend	OUT	WORD	Collective error code for the internally called Tsend blocks
errorTcon	OUT	WORD	Error code for the internally called Tcon block
errorTusend1	OUT	WORD	Error code for the internally called Tusend1 block
errorTusend2	OUT	WORD	Error code for the internally called Tusend2 block
lostSamples	OUT	UNIT	Counter for lost measured values

The following SIMATIC standard blocks are used internally:

- TCON
- TUSEND
- TDISCON

Detailed description

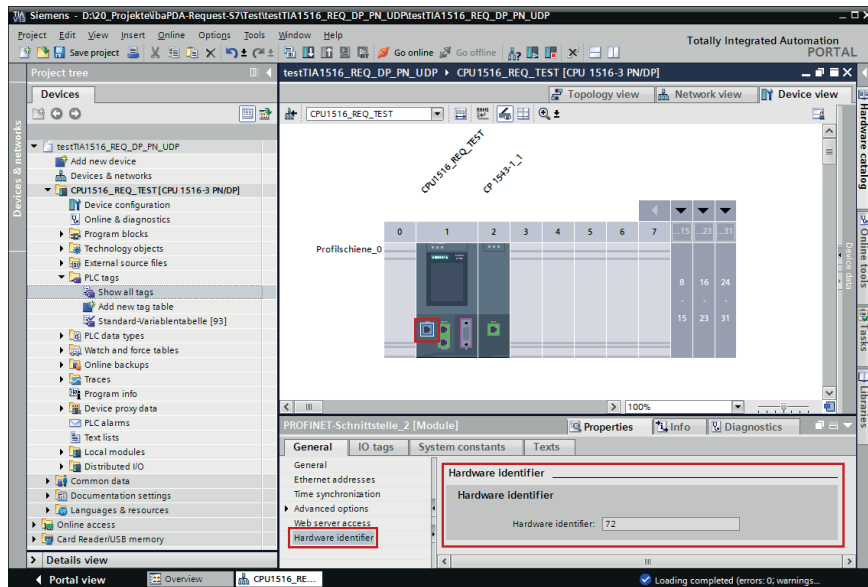
interfaceld

The HW ID of the used interface is a constant in TIA Portal and dependent on the configured hardware.

Tip



You find the hardware identifier of the marked interface under *Properties – General – Hardware identifier*



You can configure the hardware identifier as a numerical value or as a system constant of the type `Hw_Interface`. You can find the system constant under *Properties – System constants*. Always use the system constant of the interface and not of a port, or of the IO system.

PROFINET-Schnittstelle_2 [Module]				
Properties				
System constants				
Name	Type	Hardware identi.	Comment	
Local-PROFINET-Schnittstelle_2	Hw_Interface	72		
Local-PROFINET-Schnittstelle_2-Port_1	Hw_Interface	73		

connectionId

Unambiguous reference to the connection to be established, value range: 1 to 4095.

localPort

Number of the local port used

reset_com

Used to manually reset the communication connection.

DB_PDA

Pointer to the communication data range. This range is used for the data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

state

Block status in plain text

errorTsend

Collective error code for the internally called Tsend blocks

errorTcon

Error code for the internally called Tcon block

errorTusend1

Error code for the internally called 1st Tusend block

errorTusend2

Error code for the internally called 2nd Tusend block

A list of all possible error codes for the Tsend, Tusend, Tcon system blocks can be found in the Siemens documentation.

lostSamples

The counter is incremented each time a block is called if a new UDP telegram cannot be sent to *ibaPDA* because the preceding send job has not yet been completed. A constantly increasing value indicates a bottleneck in communication performance.

Error codes**errorTcon**

Hexadecimal value Status info	Description
W#16#0000	Connection successfully established.
W#16#7000	No job processing is active.
W#16#7001	Start job execution, establish connection.
W#16#7002	Connection is being established (REQ irrelevant).

Hexadecimal value Error	Description
W#16#8085	Connection ID (ID parameter) is already used by a configured connection.
W#16#8086	The ID parameter is outside the valid range.
W#16#8087	Maximum number of connections reached; no additional connection possible.
W#16#8089	The CONNECT parameter does not point to a connection description or the connection description was created manually.
W#16#809A	The structure at the CONNECT parameter is not supported on an integrated interface or the length is invalid.
W#16#809B	The element interfacedId within the TCON_xxx structure does not reference a hardware identifier of a CPU or CM/CP interface or has the value "0".
W#16#80A1	The specified connection or the port is already used.

Hexadecimal value Error	Description
W#16#80A2	Local or remote port is used by the system. The following ports are reserved locally: 20, 21, 80, 102, 135, 161, 162, 443, 34962, 34963, 34964 as well as the area 49152 to 65535.
W#16#80A3	ID is used by a connection created by the user program, which uses the same connection description at the CONNECT parameter.
W#16#80A4	IP address of the remote connection endpoint is invalid, or it matches the IP address of the local partner.
W#16#80A7	Communication error: You executed TDISCON before TCON had completed.
W#16#80B4	Only with TCON_IP_RFC: The local T selector was not specified, or the first byte does not contain the value 0x0E (only with a length of T selector = 2) or the local T selector starts with "SIMATIC-".
W#16#80B5	Only passive connection establishment is permitted for connection type 13 = UDP (parameter ActiveEstablished of the structure TCON_IP_v4/TCON_PARAM has the value TRUE).
W#16#80B6	Parameter assignment error in the ConnectionType parameter of the data block for connection description. <ul style="list-style-type: none"> ■ Only valid with TCON_IP_v4: 0x11, 0x0B and 0x13 ■ Only valid with TCON_IP_RFC: 0x0C and 0x12
W#16#80B7	With TCON_IP_v4: <ul style="list-style-type: none"> ■ TCP (active connection establishment): Remote port is "0". ■ CP (passive connection establishment): Local port is "0". ■ UDP: Local port is "0". ■ IP address of the partner endpoint is set to 0.0.0.0. With TCON_IP_RFC: <ul style="list-style-type: none"> ■ For local (LocalTSelector) or remote (RemoteTSelector) T selector, a length greater than 32 bytes is specified. ■ For TSelLength of the T selector (local or remote), a length greater than 32 is specified. ■ Error in the length of the IP address of the specific connection partner. ■ IP address of the partner endpoint is set to 0.0.0.0.
W#16#80B8	Parameter ID in the local connection description (structure at CONNECT parameter) and parameter ID of the instruction are different.
W#16#80C3	All connection resources are assigned, or ports are dynamically used by other applications or connections.

Hexadecimal value Error	Description
W#16#80C4	Temporary communication error: <ul style="list-style-type: none"> ■ The connection cannot be established at this time. ■ The connection cannot be established because the firewalls on the connection path are not open for the required ports. ■ The interface is receiving new parameters. ■ The configured connection is being removed by a TDISCON instruction.
W#16#80C5	The connection partner refuses to establish the connection, has terminated the connection or actively ended it.
W#16#80C6	Network error: The connection partner cannot be reached.
W#16#80C7	Execution timeout.
W#16#80C8	Value at the ID parameter is already used by a connection that was created by the user program. The connection uses the identical ID, but different connection settings at the CONNECT parameter.
W#16#80C9	Validation of the connection partner failed. The connection partner that wants to establish the connection does not match the defined partner at the CONNECT parameter.
W#16#80CE	The IP address of the local interface is 0.0.0.0.
W#16#80D0	With TCP and active connection endpoint: The remote_qdn parameter is an empty string. A connection cannot be established.
W#16#80D1	The remote_qdn parameter is not a fully qualified domain name. The period at the end might be missing.
W#16#80D2	No DNS server address is configured.
W#16#80D3	The fully qualified domain name cannot be resolved. Possible causes: <ul style="list-style-type: none"> ■ The DNS server is not reachable, for example, because it has been shut down or the remote port is not reachable. ■ An error occurred during communication with the DNS server. ■ The DNS server returned a valid DNS answer, but the answer contained no IPv4 address.
W#16#80E0	Unsuitable or poor message was received.
W#16#80E1	Error during handshake. Possible causes: <ul style="list-style-type: none"> ■ Aborted by the user ■ Security not high enough ■ Renewed negotiation is not supported. ■ SSL/TLS version is not supported. ■ Validation of the host name failed.

Hexadecimal value Error	Description
W#16#80E2	<p>Certificate is not supported or invalid. Possible cause: The time-of-day of the module concerned is not set or the module is not synchronized.</p> <p>Example: The default setting for the date of the module is 1/1/2012 and it was not set during commissioning. The validity period of the certificate starts on 20 August 2016 and ends on 20 August 2024. In this case, the date of the module is outside the valid period of the certificate; the certificate is invalid for the module.</p>
W#16#80E3	Certificate was discarded.
W#16#80E4	No valid certification authority found.
W#16#80E5	Certificate expired.
W#16#80E6	Integrity errors in the Transport Layer Security Protocol.
W#16#80E7	Not supported extension in X.509-V3 certificate.
W#16#80E9	TLS server without server certificate is not supported.
W#16#80EA	DTLS (UDP) protocol is not supported.
W#16#80EB	A client cannot request a client certificate.
W#16#80EC	The server cannot perform validation based on subjectAlternateName (only clients can do this).
W#16#80ED	TLSServerCertRef_m-ID is invalid.

errorTusend

Hexadecimal value Status info	Description
W#16#0000	Send job completed without error.
W#16#7000	No job processing is active.
W#16#7001	Start of job processing, data being sent. Note: During this processing phase, the operating system accesses the data in the DATA send area.
W#16#7002	Intermediate call (REQ irrelevant), job is being processed. Note: During this processing phase, the operating system accesses the data in the DATA send area.

Hexadecimal value Error	Description
W#16#8085	The LEN parameter has the value "0" or is greater than the highest valid value.
W#16#8086	The ID parameter is outside the valid range.
W#16#8088	The LEN parameter is greater than the memory area specified in DATA.
W#16#8089	The ADDR parameter does not point to a data block with the structure TADDR_Param or TADDR_SEND_QDN.
W#16#80A1	Communication error: <ul style="list-style-type: none"> ■ The specified connection between user program and communication layer of the operating system has not yet been established. ■ The specified connection between the user program and the communication layer of the operating system is currently being terminated. Transfer over this connection is not possible. ■ The interface is being re-initialized.
W#16#80B1	You changed the DATA parameter before the current job finished.
W#16#80A4	IP address (at the ADDR parameter) of the remote connection endpoint is invalid, or it matches the IP address of the local partner.
W#16#80B3	<ul style="list-style-type: none"> ■ The protocol variant (connection_type parameter in the connection description) is not set to UDP. Please use TSEND. ■ Parameter ADDR: Invalid settings for port number.
W#16#80B7	The length of the structure referenced by the ADDR parameter is not 8 bytes.
W#16#80C3	<ul style="list-style-type: none"> ■ A block with this ID is already being processed in a different priority class. ■ Internal lack of resources.

Other documentation

For more information, refer the Siemens documentation about the TCON and TSEND blocks.

4.3 ibaREQ3sym iba block family

These blocks initialize and control communication between *ibaPDA* and the S7-1500 controllers.

The ibaREQ3sym block family allows access to both optimized and non-optimized data blocks. Addressing is done by using only the symbol names.

The ibaREQ3sym module family is the recommended access variant as of TIA Portal V19. With this version, runtime optimizations are possible, such as the outsourcing of the cyclical reading of the actual values from the ibaREQ3sym_M block. Therefore, the individual blocks can be specifically distributed across different call levels.

One set of Request blocks has to be called for each Request module (connection) in *ibaPDA*. The blocks used are part of the iba S7 library, see [iba S7 library](#), page 107.

Note



Only use Request blocks from the latest iba S7 library!

Request blocks in application examples can be outdated and, thus, cause errors.

Use different Request block combinations depending on the current system configuration:

Request block	Recommended call level
ibaREQ3sym_M	OB 1 or lower-priority OB
ibaREQ3sym_UP	OB where the data is updated
ibaREQ3sym_DB_PDA	-
ibaREQ3sym-Interface	-

- ibaREQ3sym_M (Management)
The block realizes the communication with *ibaPDA*. Ideally, the block is called in OB1 or lower-priority OB.
- ibaREQ3sym_UDP (Provision and buffering of current signal values)
The block provides the current signal values and buffers them. Ideally, the block is called in the OB in which the data is updated.
- ibaREQ3sym_DB_PDA (DB interfaces)
This DB acts as an interface to *ibaPDA* and between the different Request blocks.

4.3.1 ibaREQ3sym_M

Description of formal parameters

Name	Type	Data Type	Description
reset	IN	BOOL	FALSE: No reset (default) TRUE: Block reset
DB_PDA	INOUT	UDT	DB for the <i>ibaPDA</i> communication interface ibaREQ3sym_DB_PDA
state	OUT	STRING[16]	Block status
errorStatusRun	OUT	WORD	Internal error code
errorStatus1	OUT	WORD	Internal error code
errorStatus2	OUT	WORD	Error code for internally called blocks

Detailed description

reset

Used to manually reset the block

DB_PDA

Pointer to the communication data range. This range is used for the data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

state

Block status in plain text

errorStatusRun

Internal error code for the block. If there is no error, the value 0 is output.

errorStatus1

Internal error code for the block. If there is no error, the value 0 is output.

errorStatus2

Internal error code for the block. If there is no error, the value 0 is output.

Error codes

Hexadecimal value errorStatus	Description
01x8001	invalid command id
01x800F	internal error (error moving data - pdainfo)
01x8010	internal error (error moving data - command)
01x8011	internal error (error moving data - response reset)
01x8012	internal error (error moving data - response diag)
01x8014	internal error (error moving data - response request)
01x801F	internal error (error moving data - response unknown)
01x8101	internal error (error moving data - response invalid command ID)
01x8202	internal error (error moving data - diag response data)

Hexadecimal value errorStatus	Description
01x8401	internal error (error moving data - request command data)
01x8402	internal error (error moving data - request response data)
01x8410	too many requested symbols in total (>2000)
01x8411	too many requested symbols in one command (>100)
01x8412	requested data length is too long (<max)
01x8413	requested data length is too short (<0)
01x8422	error moving symbol offsets
01x8423	error moving symbol names
01x8425	error resolving requested symbols
01x8426	error moving resolved symbols data
01x8512	no active symbols
01x8FFF	undefined internal error

4.3.2 ibaREQ3sym_UDP

Description of formal parameters

Name	Type	Data Type	Description
interfaceld	IN	HW_ANY	HW ID of the used interface
connectionId	IN	CONN_OUC	Unique connection ID of the send block (TSEND_C)
localPort	IN	UINT	Local port number
reset_com	IN	BOOL	FALSE: No reset (default) TRUE: Communication connection reset
DB_PDA	INOUT	DB_ANY	DB for the <i>ibaPDA</i> communication interface ibaREQ3sym_DB_PDA
state	OUT	STRING[16]	Block status
errorTsend	OUT	WORD	Collective error code for the internally called Tsend blocks
errorTcon	OUT	WORD	Error code for the internally called Tcon block
errorTusend1	OUT	WORD	Error code for the internally called Tusend1 block
errorTusend2	OUT	WORD	Error code for the internally called Tusend2 block
lostSamples	OUT	UNIT	Counter for lost measured values

The following SIMATIC standard blocks are used internally:

- TCON
- TUSEND
- TDISCON

Detailed description

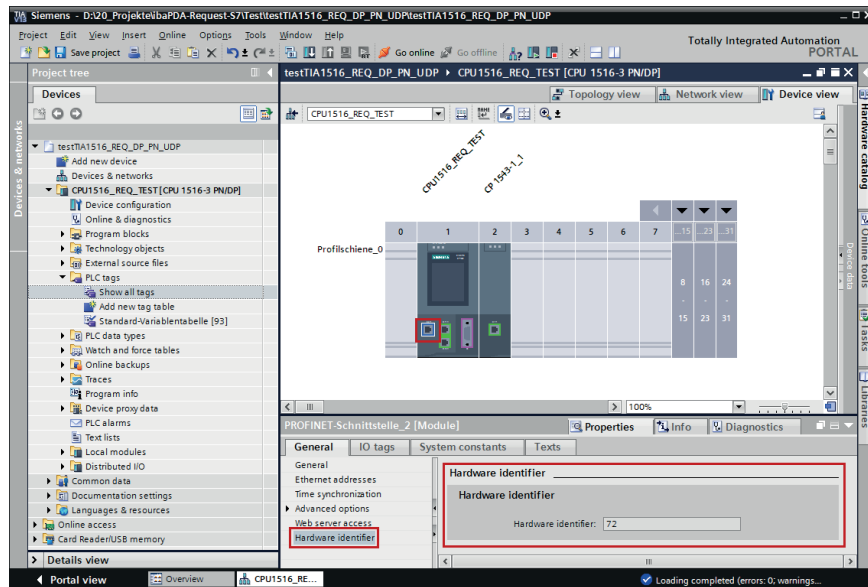
interfaceld

The HW ID of the used interface is a constant in TIA Portal and dependent on the configured hardware.

Tip



You find the hardware identifier of the marked interface under *Properties – General – Hardware identifier*



You can configure the hardware identifier as a numerical value or as a system constant of the type Hw_Interface. You can find the system constant under *Properties – System constants*. Always use the system constant of the interface and not of a port, or of the IO system.

PROFINET-Schnittstelle_2 [Module]				
Properties				
System constants				
Name	Type	Hardware identi.	Comment	
Local-PROFINET-Schnittstelle_2	Hw_Interface	72		
Local-PROFINET-Schnittstelle_2~Port_1	Hw_Interface	73		

connectionId

Unambiguous reference to the connection to be established, value range: 1 to 4095.

localPort

Number of the local port used

reset_com

Used to manually reset the communication connection.

DB_PDA

Pointer to the communication data range. This range is used for the data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

state

Block status in plain text

errorTsend

Collective error code for the internally called Tsend blocks

errorTcon

Error code for the internally called Tcon block

errorTusend1

Error code for the internally called 1st Tusend block

errorTusend2

Error code for the internally called 2nd Tusend block

A list of all possible error codes for the Tsend, Tusend, Tcon system blocks can be found in the Siemens documentation.

lostSamples

The counter is incremented each time a block is called if a new UDP telegram cannot be sent to *ibaPDA* because the preceding send job has not yet been completed. A constantly increasing value indicates a bottleneck in communication performance.

Error codes**errorTcon**

Hexadecimal value Status info	Description
W#16#0000	Connection successfully established.
W#16#7000	No job processing is active.
W#16#7001	Start job execution, establish connection.
W#16#7002	Connection is being established (REQ irrelevant).

Hexadecimal value Error	Description
W#16#8085	Connection ID (ID parameter) is already used by a configured connection.
W#16#8086	The ID parameter is outside the valid range.
W#16#8087	Maximum number of connections reached; no additional connection possible.
W#16#8089	The CONNECT parameter does not point to a connection description or the connection description was created manually.
W#16#809A	The structure at the CONNECT parameter is not supported on an integrated interface or the length is invalid.
W#16#809B	The element interfacedId within the TCON_xxx structure does not reference a hardware identifier of a CPU or CM/CP interface or has the value "0".
W#16#80A1	The specified connection or the port is already used.

Hexadecimal value Error	Description
W#16#80A2	Local or remote port is used by the system. The following ports are reserved locally: 20, 21, 80, 102, 135, 161, 162, 443, 34962, 34963, 34964 as well as the area 49152 to 65535.
W#16#80A3	ID is used by a connection created by the user program, which uses the same connection description at the CONNECT parameter.
W#16#80A4	IP address of the remote connection endpoint is invalid, or it matches the IP address of the local partner.
W#16#80A7	Communication error: You executed TDISCON before TCON had completed.
W#16#80B4	Only with TCON_IP_RFC: The local T selector was not specified, or the first byte does not contain the value 0x0E (only with a length of T selector = 2) or the local T selector starts with "SIMATIC-".
W#16#80B5	Only passive connection establishment is permitted for connection type 13 = UDP (parameter ActiveEstablished of the structure TCON_IP_v4/TCON_PARAM has the value TRUE).
W#16#80B6	Parameter assignment error in the ConnectionType parameter of the data block for connection description. <ul style="list-style-type: none"> ■ Only valid with TCON_IP_v4: 0x11, 0x0B and 0x13 ■ Only valid with TCON_IP_RFC: 0x0C and 0x12
W#16#80B7	With TCON_IP_v4: <ul style="list-style-type: none"> ■ TCP (active connection establishment): Remote port is "0". ■ CP (passive connection establishment): Local port is "0". ■ UDP: Local port is "0". ■ IP address of the partner endpoint is set to 0.0.0.0. With TCON_IP_RFC: <ul style="list-style-type: none"> ■ For local (LocalTSelector) or remote (RemoteTSelector) T selector, a length greater than 32 bytes is specified. ■ For TSelLength of the T selector (local or remote), a length greater than 32 is specified. ■ Error in the length of the IP address of the specific connection partner. ■ IP address of the partner endpoint is set to 0.0.0.0.
W#16#80B8	Parameter ID in the local connection description (structure at CONNECT parameter) and parameter ID of the instruction are different.
W#16#80C3	All connection resources are assigned, or ports are dynamically used by other applications or connections.

Hexadecimal value Error	Description
W#16#80C4	Temporary communication error: <ul style="list-style-type: none"> ■ The connection cannot be established at this time. ■ The connection cannot be established because the firewalls on the connection path are not open for the required ports. ■ The interface is receiving new parameters. ■ The configured connection is being removed by a TDISCON instruction.
W#16#80C5	The connection partner refuses to establish the connection, has terminated the connection or actively ended it.
W#16#80C6	Network error: The connection partner cannot be reached.
W#16#80C7	Execution timeout.
W#16#80C8	Value at the ID parameter is already used by a connection that was created by the user program. The connection uses the identical ID, but different connection settings at the CONNECT parameter.
W#16#80C9	Validation of the connection partner failed. The connection partner that wants to establish the connection does not match the defined partner at the CONNECT parameter.
W#16#80CE	The IP address of the local interface is 0.0.0.0.
W#16#80D0	With TCP and active connection endpoint: The remote_qdn parameter is an empty string. A connection cannot be established.
W#16#80D1	The remote_qdn parameter is not a fully qualified domain name. The period at the end might be missing.
W#16#80D2	No DNS server address is configured.
W#16#80D3	The fully qualified domain name cannot be resolved. Possible causes: <ul style="list-style-type: none"> ■ The DNS server is not reachable, for example, because it has been shut down or the remote port is not reachable. ■ An error occurred during communication with the DNS server. ■ The DNS server returned a valid DNS answer, but the answer contained no IPv4 address.
W#16#80E0	Unsuitable or poor message was received.
W#16#80E1	Error during handshake. Possible causes: <ul style="list-style-type: none"> ■ Aborted by the user ■ Security not high enough ■ Renewed negotiation is not supported. ■ SSL/TLS version is not supported. ■ Validation of the host name failed.

Hexadecimal value Error	Description
W#16#80E2	Certificate is not supported or invalid. Possible cause: The time-of-day of the module concerned is not set or the module is not synchronized. Example: The default setting for the date of the module is 1/1/2012 and it was not set during commissioning. The validity period of the certificate starts on 20 August 2016 and ends on 20 August 2024. In this case, the date of the module is outside the valid period of the certificate; the certificate is invalid for the module.
W#16#80E3	Certificate was discarded.
W#16#80E4	No valid certification authority found.
W#16#80E5	Certificate expired.
W#16#80E6	Integrity errors in the Transport Layer Security Protocol.
W#16#80E7	Not supported extension in X.509-V3 certificate.
W#16#80E9	TLS server without server certificate is not supported.
W#16#80EA	DTLS (UDP) protocol is not supported.
W#16#80EB	A client cannot request a client certificate.
W#16#80EC	The server cannot perform validation based on subjectAlternateName (only clients can do this).
W#16#80ED	TLSServerCertRef_m-ID is invalid.

errorTusend

Hexadecimal value Status info	Description
W#16#0000	Send job completed without error.
W#16#7000	No job processing is active.
W#16#7001	Start of job processing, data being sent. Note: During this processing phase, the operating system accesses the data in the DATA send area.
W#16#7002	Intermediate call (REQ irrelevant), job is being processed. Note: During this processing phase, the operating system accesses the data in the DATA send area.

Hexadecimal value Error	Description
W#16#8085	The LEN parameter has the value "0" or is greater than the highest valid value.
W#16#8086	The ID parameter is outside the valid range.
W#16#8088	The LEN parameter is greater than the memory area specified in DATA.
W#16#8089	The ADDR parameter does not point to a data block with the structure TADDR_Param or TADDR_SEND_QDN.
W#16#80A1	Communication error: <ul style="list-style-type: none"> ■ The specified connection between user program and communication layer of the operating system has not yet been established. ■ The specified connection between the user program and the communication layer of the operating system is currently being terminated. Transfer over this connection is not possible. ■ The interface is being re-initialized.
W#16#80B1	You changed the DATA parameter before the current job finished.
W#16#80A4	IP address (at the ADDR parameter) of the remote connection endpoint is invalid, or it matches the IP address of the local partner.
W#16#80B3	<ul style="list-style-type: none"> ■ The protocol variant (connection_type parameter in the connection description) is not set to UDP. Please use TSEND. ■ Parameter ADDR: Invalid settings for port number.
W#16#80B7	The length of the structure referenced by the ADDR parameter is not 8 bytes.
W#16#80C3	<ul style="list-style-type: none"> ■ A block with this ID is already being processed in a different priority class. ■ Internal lack of resources.

Other documentation



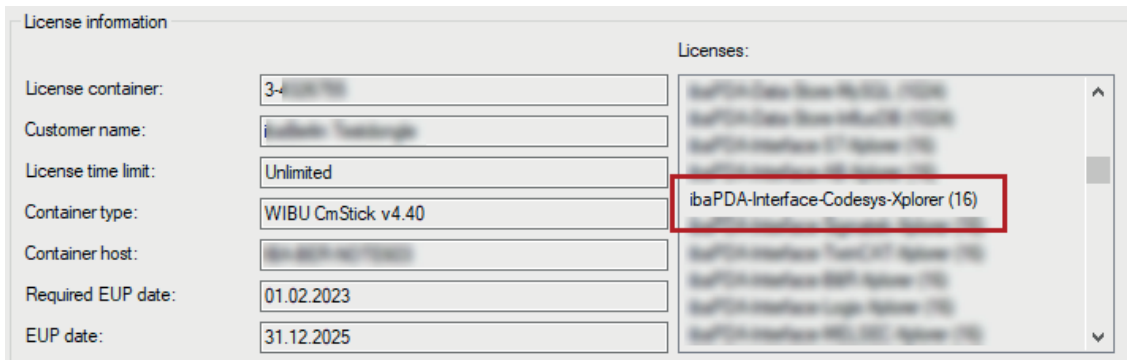
For more information, refer the Siemens documentation about the TCON, TDISCON and TSEND blocks.

5 Diagnostics

5.1 License

If the interface is not displayed in the signal tree, you can either check in *ibaPDA* in the I/O Manager under *General – Settings* or in the *ibaPDA* service status application whether your license for the interface *ibaPDA-Request-S7-UDP* has been properly recognized. The number of licensed connections is shown in brackets.

The figure below shows the license for the *Codesys Xplorer* interface as an example.



5.2 Visibility of the interface

If the interface is not visible despite a valid license, it may be hidden.

Check the settings in the *General* tab in the *Interfaces* node.

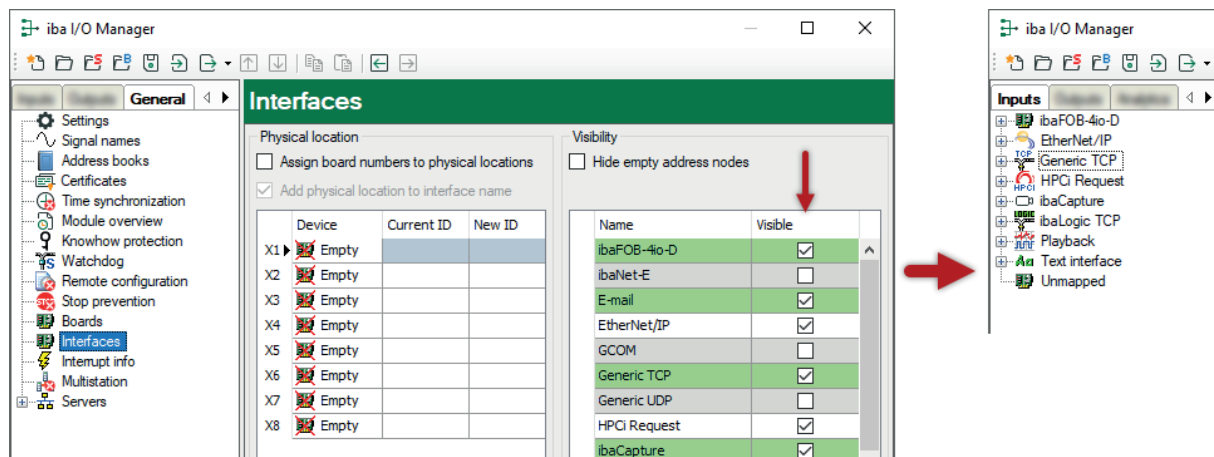
Visibility

The table *Visibility* lists all the interfaces that are available either through licenses or installed cards. These interfaces can also be viewed in the interface tree.

You can hide or display the interfaces not required in the interface tree by using the checkbox in the *Visible* column.

Interfaces with configured modules are highlighted in green and cannot be hidden.

Selected interfaces are visible, the others are hidden:



5.3 Log files

If connections to target platforms or clients have been established, all connection-specific actions are logged in a text file. You can open this (current) file and, e.g., scan it for indications of possible connection problems.

You can open the log file via the button <Open log file>. The button is available in the I/O Manager:

- for many interfaces in the respective interface overview
- for integrated servers (e.g. OPC UA server) in the *Diagnostics* tab.

In the file system on the hard drive, you can find the log files of the *ibaPDA* server (...\\ProgramData\\iba\\ibaPDA\\Log). The file names of the log files include the name or abbreviation of the interface type.

Files named `interface.txt` are always the current log files. Files named `Interface_yyyy_mm_dd_hh_mm_ss.txt` are archived log files.

Examples:

- `ethernetipLog.txt` (log of EtherNet/IP connections)
- `AbEthLog.txt` (log of Allen-Bradley Ethernet connections)
- `OpcUAServerLog.txt` (log of OPC UA server connections)

5.4 Connection diagnostics with PING

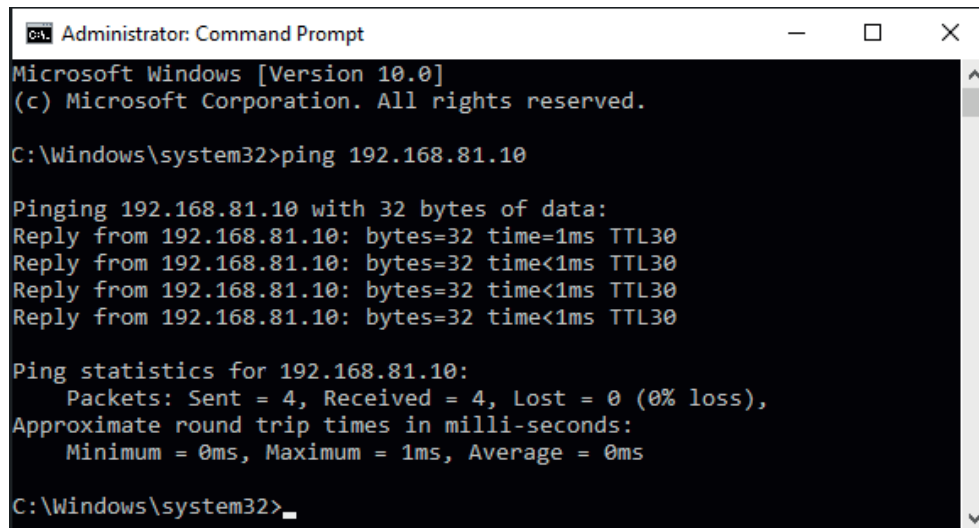
PING is a system command with which you can check if a certain communication partner can be reached in an IP network.

1. Open a Windows command prompt.



2. Enter the command "ping" followed by the IP address of the communication partner and press <ENTER>.

→ With an existing connection you receive several replies.

A screenshot of a Windows Command Prompt window titled 'Administrator: Command Prompt'. The window shows the output of the 'ping 192.168.81.10' command. The output indicates that the connection is successful, with four replies received from the destination IP address. The ping statistics show that all packets were sent and received, with a 0% loss rate.

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0]
(c) Microsoft Corporation. All rights reserved.

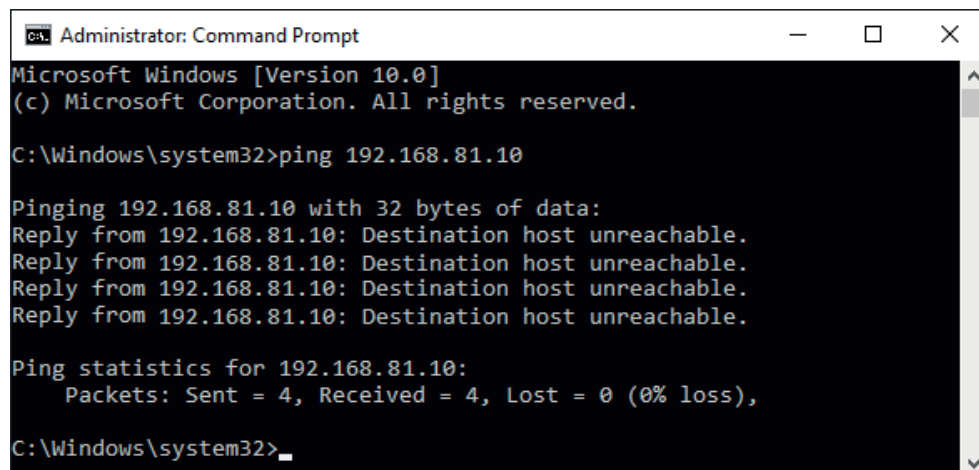
C:\Windows\system32>ping 192.168.81.10

Pinging 192.168.81.10 with 32 bytes of data:
Reply from 192.168.81.10: bytes=32 time=1ms TTL=30
Reply from 192.168.81.10: bytes=32 time<1ms TTL=30
Reply from 192.168.81.10: bytes=32 time<1ms TTL=30
Reply from 192.168.81.10: bytes=32 time<1ms TTL=30

Ping statistics for 192.168.81.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Windows\system32>
```

→ With no existing connection you receive error messages.

A screenshot of a Windows Command Prompt window titled 'Administrator: Command Prompt'. The window shows the output of the 'ping 192.168.81.10' command. The output indicates that the connection is failed, with four replies received from the destination IP address, all of which are 'Destination host unreachable'. The ping statistics show that all packets were sent and received, with a 0% loss rate.

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>ping 192.168.81.10

Pinging 192.168.81.10 with 32 bytes of data:
Reply from 192.168.81.10: Destination host unreachable.
Reply from 192.168.81.10: Destination host unreachable.
Reply from 192.168.81.10: Destination host unreachable.
Reply from 192.168.81.10: Destination host unreachable.

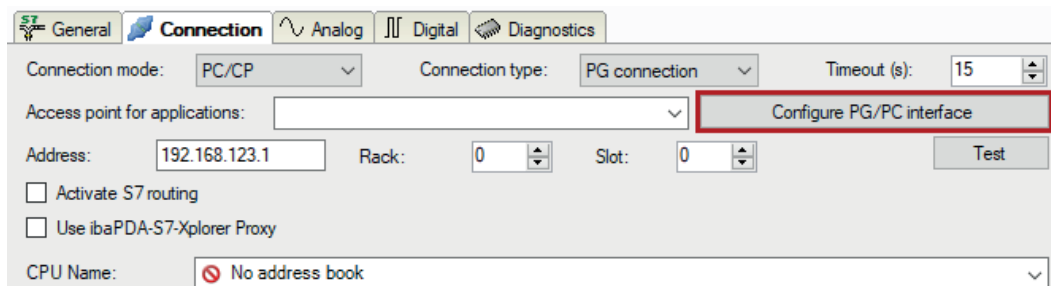
Ping statistics for 192.168.81.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

C:\Windows\system32>
```

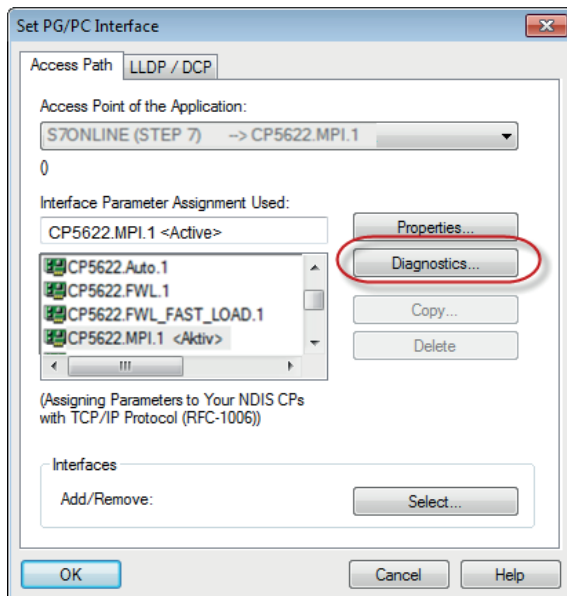
5.5 Connection diagnostics with PG/PC interface

Use the diagnostic function of the PG/PC interface to the functionality and connection configuration.

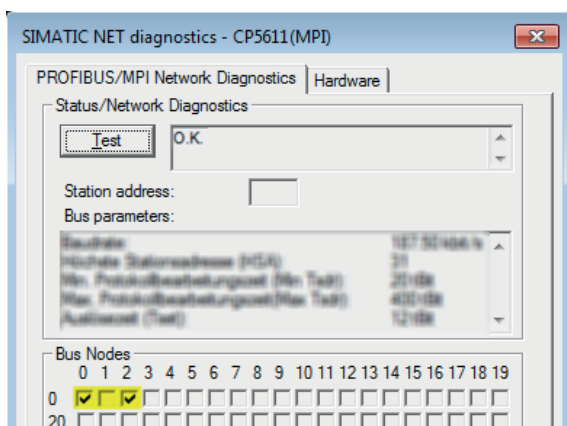
1. Open the dialog for configuring the PG/PC interface with the <Configure PG/PC interface> button.



2. Open the diagnostics dialog with the <Diagnostics> button.

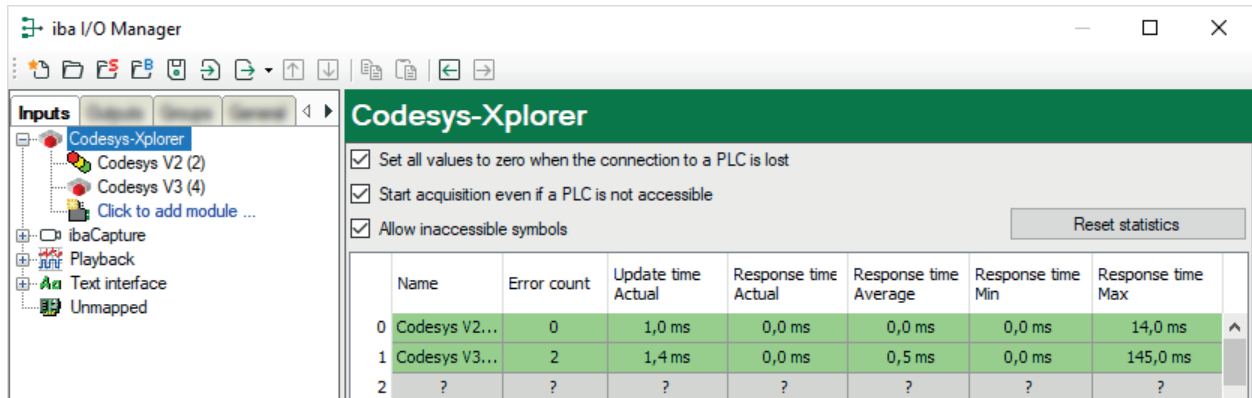


3. Start the network diagnostics with the <Test> button.
Check the availability of the bus devices with the <Read> button.
- The following figure shows an example of a diagnostics of a SIMATIC Net CP5622 (PROFIBUS-US).
There is one active station at each of the addresses 0 and 2.



5.6 Connection table

For every Ethernet-based interface, there is a table available in the I/O Manager which shows the status of each connection. Each line represents one connection. The following figure shows, as an example, the connection table of the Codesys-Xplorer interface:



The connected target systems (controllers) are identified by their name or IP address in the first (left) column.

Depending on the interface type the table shows error counters, read counters and/or data sizes, as well as the cycle times, refresh times and/or update times of the different connections during the data acquisition.

Click the <Reset statistics> button to reset the error counters and the calculation of the response times.

Additional information is provided by the background color of the table rows:

Color	Meaning
Green	The connection is OK and the data are read.
Yellow	The connection is OK, however the data update is slower than the configured update time.
Red	The connection has failed.
Gray	No connection configured.

5.7 Diagnostic modules

Diagnostic modules are available for most Ethernet based interfaces and Xplorer interfaces. Using a diagnostic module, information from the diagnostic displays (e.g. diagnostic tabs and connection tables of an interface) can be acquired as signals.

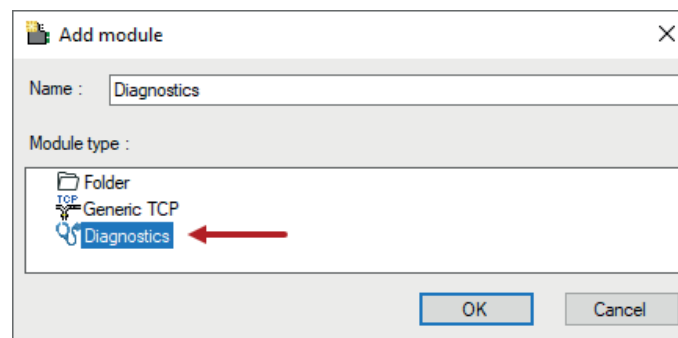
A diagnostic module is always assigned to a data acquisition module of the same interface and supplies its connection information. By using a diagnostic module, you can record and analyze the diagnostic information continuously in the *ibaPDA* system.

Diagnostic modules do not consume any license connections because they do not establish their own connection but refer to another module.

Example for the use of diagnostic modules:

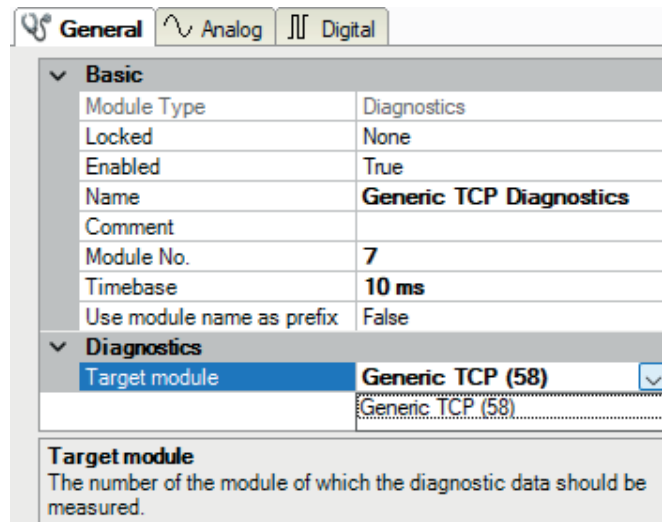
- A notification can be generated, whenever the error counter of a communication connection exceeds a certain value or the connection gets lost.
- In case of a disturbance, the current response times in the telegram traffic may be documented in an incident report.
- The connection status can be visualized in *ibaQPanel*.
- You can forward diagnostic information via the SNMP server integrated in *ibaPDA* or via OPC DA/UA server to superordinate monitoring systems like network management tools.

In case the diagnostic module is available for an interface, a "Diagnostics" module type is shown in the *Add module* dialog (example: Generic TCP).



Module settings diagnostic module

For a diagnostic module, you can make the following settings (example: Generic TCP):



General Analog Digital

Basic

Module Type	Diagnostics
Locked	None
Enabled	True
Name	Generic TCP Diagnostics
Comment	
Module No.	7
Timebase	10 ms
Use module name as prefix	False

Diagnostics

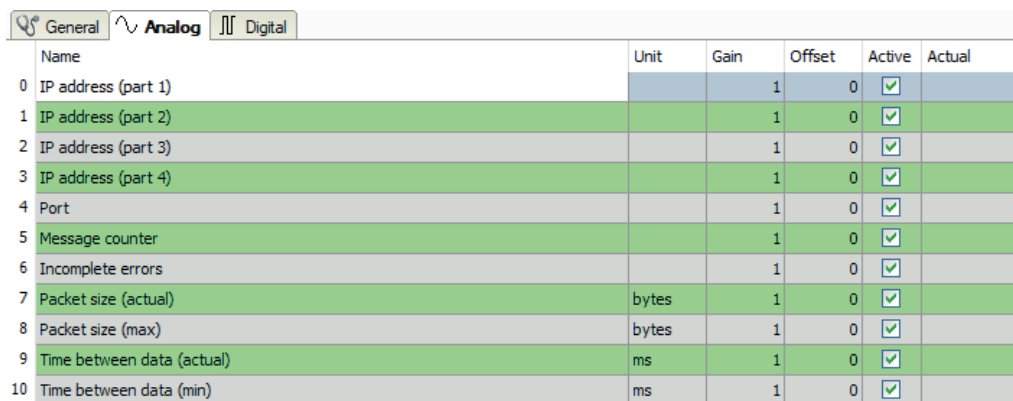
Target module	Generic TCP (58)
---------------	------------------

Target module
The number of the module of which the diagnostic data should be measured.

The basic settings of a diagnostic module equal those of other modules.

There is only one setting which is specific for the diagnostic module: the target module.

By selecting the target module, you assign the diagnostic module to the module on which you want to acquire information about the connection. You can select the supported modules of this interface in the drop-down list of the setting. You can assign exactly one data acquisition module to each diagnostic module. When having selected a module, the available diagnostic signals are immediately added to the *Analog* and *Digital* tabs. It depends on the type of interface, which signals exactly are added. The following example lists the analog values of a diagnostic module for a Generic TCP module.



	Name	Unit	Gain	Offset	Active	Actual
0	IP address (part 1)		1	0	<input checked="" type="checkbox"/>	
1	IP address (part 2)		1	0	<input checked="" type="checkbox"/>	
2	IP address (part 3)		1	0	<input checked="" type="checkbox"/>	
3	IP address (part 4)		1	0	<input checked="" type="checkbox"/>	
4	Port		1	0	<input checked="" type="checkbox"/>	
5	Message counter		1	0	<input checked="" type="checkbox"/>	
6	Incomplete errors		1	0	<input checked="" type="checkbox"/>	
7	Packet size (actual)	bytes	1	0	<input checked="" type="checkbox"/>	
8	Packet size (max)	bytes	1	0	<input checked="" type="checkbox"/>	
9	Time between data (actual)	ms	1	0	<input checked="" type="checkbox"/>	
10	Time between data (min)	ms	1	0	<input checked="" type="checkbox"/>	

For example, the IP (v4) address of a Generic TCP module (see fig. above) will always be split into 4 parts derived from the dot-decimal notation, for better reading. Also other values are being determined, as there are port number, counters for telegrams and errors, data sizes and telegram cycle times. The following example lists the digital values of a diagnostic module for a Generic TCP module.



	Name	Active	Actual
0	Active connection mode	<input checked="" type="checkbox"/>	
1	Invalid packet	<input checked="" type="checkbox"/>	
2	Connecting	<input checked="" type="checkbox"/>	
3	Connected	<input checked="" type="checkbox"/>	

Diagnostic signals

Depending on the interface type, the following signals are available:

Signal name	Description
Active	Only relevant for redundant connections. Active means that the connection is used to measure data, i.e. for redundant standby connections the value is 0. For normal/non-redundant connections, the value is always 1.
Buffer file size (actual/avg/max)	Size of the file for buffering statements
Buffer memory size (actual/avg/max)	Size of the memory used by buffered statements
Buffered statements	Number of unprocessed statements in the buffer
Buffered statements lost	Number of buffered but unprocessed and lost statements
Connected	Connection is established
Connected (in)	A valid data connection for the reception (in) is available
Connected (out)	A valid data connection for sending (out) is available
Connecting	Connection being established
Connection attempts (in)	Number of attempts to establish the receive connection (in)
Connection attempts (out)	Number of attempts to establish the send connection (out)
Connection ID O->T	ID of the connection for output data (from the target system to <i>ibaPDA</i>). Corresponds to the assembly instance number
Connection ID T->O	ID of the connection for input data (from <i>ibaPDA</i> to target system). Corresponds to the assembly instance number
Connection phase (in)	Status of the ibaNet-E data connection for reception (in)
Connection phase (out)	Status of the ibaNet-E data connection for sending (out)
Connections established (in)	Number of currently valid data connections for reception (in)
Connections established (out)	Number of currently valid data connections for sending (out)
Data length	Length of the data message in bytes
Data length O->T	Size of the output message in byte
Data length T->O	Size of the input message in byte
Destination IP address (part 1-4) O->T	4 octets of the IP address of the target system Output data (from target system to <i>ibaPDA</i>)
Destination IP address (part 1-4) T->O	4 octets of the IP address of the target system Input data (from <i>ibaPDA</i> to target system)
Disconnects (in)	Number of currently interrupted data connections for reception (in)
Disconnects (out)	Number of currently interrupted data connections for sending (out)
Error counter	Communication error counter
Exchange ID	ID of the data exchange
Incomplete errors	Number of incomplete messages

Signal name	Description
Incorrect message type	Number of received messages with wrong message type
Input data length	Length of data messages with input signals in bytes (<i>ibaPDA</i> receives)
Invalid data points	Number of received data points with missing configuration
Invalid packet	Invalid data packet detected
IP address (part 1-4)	4 octets of the IP address of the target system
Keepalive counter	Number of Keepalive messages received by the OPC UA Server
Lost images	Number of lost images (in) that were not received even after a retransmission
Lost Profiles	Number of incomplete/incorrect profiles
Message counter	Number of messages received
Messages per cycle	Number of messages in the cycle of the update time
Messages received since configuration	Number of received data telegrams (in) since start of acquisition
Messages received since connection start	Number of received data telegrams (in) since the start of the last connection setup. Reset with each connection loss.
Messages sent since configuration	Number of sent data telegrams (out) since start of acquisition
Messages sent since connection start	Number of sent data telegrams (out) since the start of the last connection setup. Reset with each connection loss.
Multicast join error	Number of multicast login errors
Number of request commands	Counter for request messages from <i>ibaPDA</i> to the PLC/CPU
Output data length	Length of the data messages with output signals in bytes (<i>ibaPDA</i> sends)
Packet size (actual)	Size of the currently received message
Packet size (max)	Size of the largest received message
Ping time (actual)	Response time for a ping telegram
Port	Port number for communication
Producer ID (part 1-4)	Producer ID as 4-byte unsigned integer
Profile Count	Number of completely recorded profiles
Read counter	Number of read accesses/data requests
Receive counter	Number of messages received
Response time (actual/average/max/min)	<p>Response time is the time between measured value request from <i>ibaPDA</i> and response from the PLC or reception of the data.</p> <p>Actual: current value</p> <p>Average/max/min: static values of the update time since the last start of the acquisition or reset of the counters.</p>
Retransmission requests	Number of data messages requested again if lost or delayed

Signal name	Description
Rows (last)	Number of resulting rows by the last SQL query (within the configured range of result rows)
Rows (maximum)	Maximum number of resulting rows by any SQL query since the last start of acquisition (possible maximum equals the configured number of result rows)
Send counter	Number of send messages
Sequence errors	Number of sequence errors
Source IP address (part 1-4) O->T	4 octets of the IP address of the target system Output data (from target system to <i>ibaPDA</i>)
Source IP address (part 1-4) T->O	4 octets of the IP address of the target system Input data (from <i>ibaPDA</i> to target system)
Statements processed	Number of executed statements since last start of acquisition
Synchronization	Device is synchronized for isochronous acquisition
Time between data (actual/ max/min)	Time between two correctly received messages Actual: between the last two messages Max/min: statistical values since start of acquisition or reset of counters
Time offset (actual)	Measured time difference of synchronicity between <i>ibaPDA</i> and the <i>ibaNet-E</i> device
Topics Defined	Number of defined topics
Topics Updated	Number of updated topics
Unknown sensor	Number of unknown sensors
Update time (actual/average/ configured/max/min)	Specifies the update time in which the data is to be retrieved from the PLC, the CPU or from the server (configured). Default is equal to the parameter "Timebase". During the measurement, the real actual update time can be higher than the set value if the PLC needs more time to transfer the data. How fast the data is really updated, you can check in the connection table. The minimum achievable update time is influenced by the number of signals. The more signals are acquired, the greater the update time becomes. Average/max/min: static values of the update time since the last start of the acquisition or reset of the counters.
Write counter	Number of successful write accesses
Write lost counter	Number of failed write accesses

6 Appendix

6.1 iba S7 library

The iba S7 library is available in two versions:

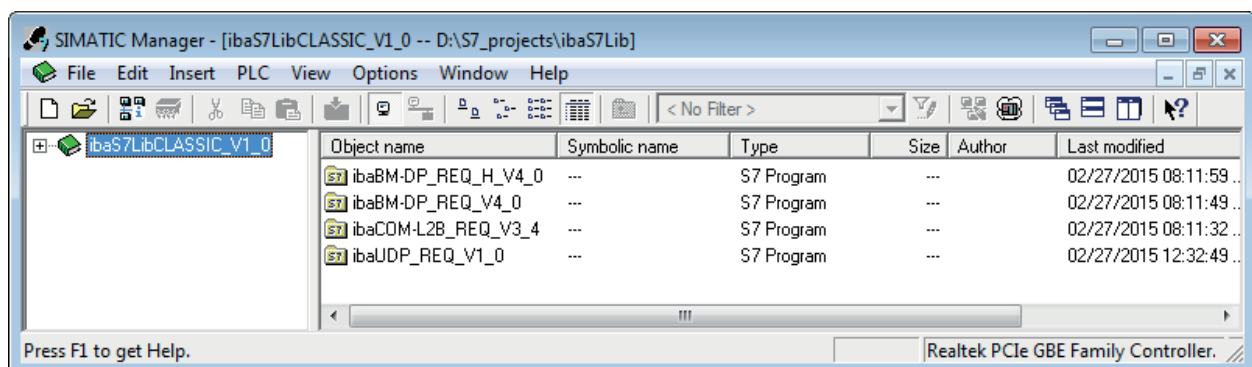
- SIMATIC Manager : STEP 7 ≥ V5.5
- SIMATIC TIA Portal STEP 7 ≥ V18

6.1.1 iba S7 library for SIMATIC Manager

The iba S7 library for SIMATIC Manager ("ibaS7LibCLASSIC_Vx_y") is suitable for SIMATIC Manager V5.5 or higher. It contains the Request blocks described in the manual, which are required for the use of *ibaPDA-Request-S7-UDP*.

You find the iba S7 library as an archived file on the data storage medium "iba Software & Manuals" in the following directory:

`\04_Libraries_and_Examples\10_Libraries\01_SIMATIC_S7\`



The following components are included:

iba connection	Block name	Block no.	Note
ibaBM-DP ibaBM-DPM-S	ibaDP_Req	FC122	
	ibaDP_DB_PDA	DB10	
	ibaDP_DB_work	DB25	
ibaBM-DP ibaBM-DPM-S Redundancy mode	ibaDP_Req_H	FC123	For S7-400H
	ibaDP_DB_PDA	DB10	
	ibaDP_DB_work	DB25	
ibaBM-PN	ibaREQ_M	FB140	
	ibaREQ_PN	FB141	
	ibaREQ_PNdev	FB150	
	ibaREQ_DB	DB15	
	ibaUDT_UDPact	UDT145	

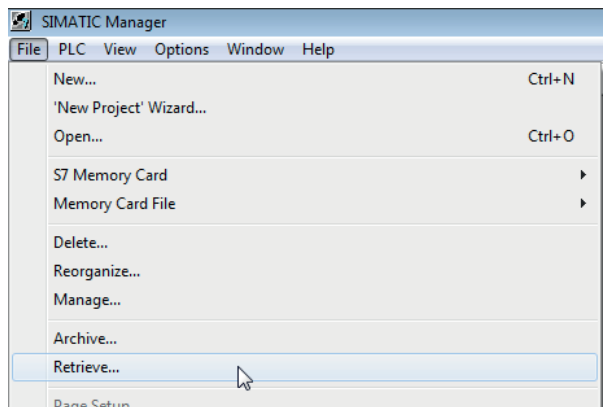
iba connection	Block name	Block no.	Note
ibaCom-L2B	ibaL2B_Init	FC111	formerly FC23 and FC101
	ibaL2B_Req	FC112	formerly FC22 and FC100
	ibaL2B_Req_CP	FC113	formerly FC26 and FC102 only necessary when using a CP342-5 instead of the FC112
	ibaL2B_DB_work	DB22	
	ibaL2B_DB_Struct	UDT22	
	ibaL2B_CP_SNDRCV	DB10	only necessary when using a CP342-5
ibaPDA-Interface-S7-TCP/UDP	ibaREQ_M	FB140	
	ibaREQ_UDPact	FB145	
	ibaREQ_UDPint	FB146	
	ibaREQ_UDPext3	FB147	
	ibaREQ_UDPext4	FB148	
	ibaREQ_DB	DB15	
	ibaUDT_UDPact	UDT145	

Table 3: ibaS7LibCLASSIC block overview

6.1.1.1 Integrating the library into SIMATIC Manager

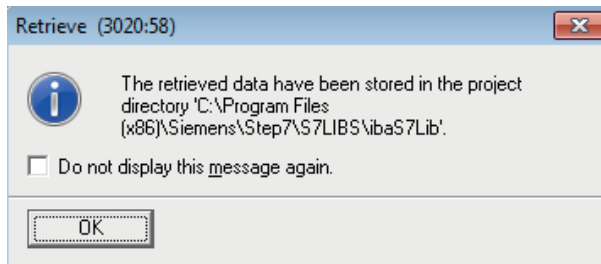
For integrating the library, it has to be retrieved in the SIMATIC Manager. Copy the iba S7 library to a local directory of your computer, on which the SIMATIC Manager is executed.

1. Select the menu *File – Retrieve*.

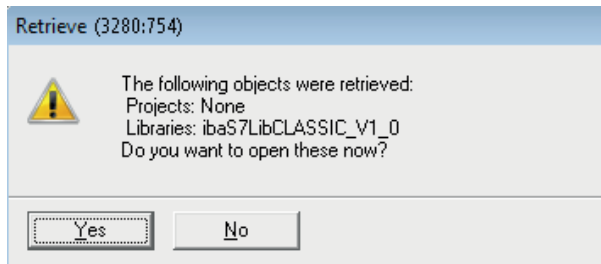


2. Select the archive file of the iba S7 library and select a storage location for the extracted library in a next step.

3. Confirm the message for a successful extraction.



4. Open the library by confirming the following dialog with <YES>.

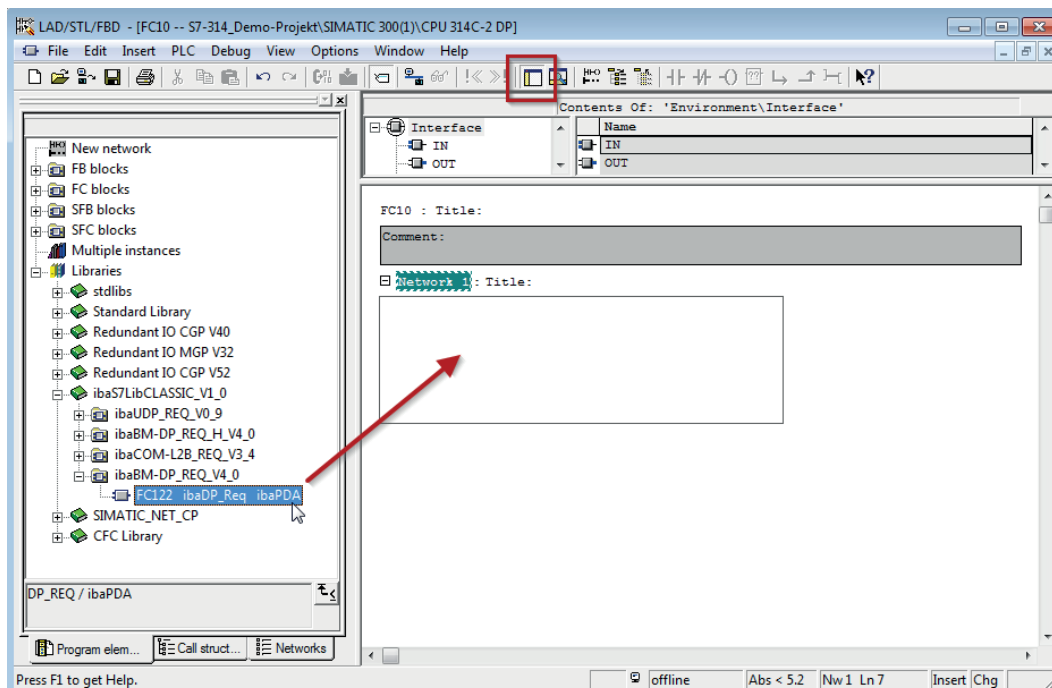


→ Now, the library is integrated and can be closed again.

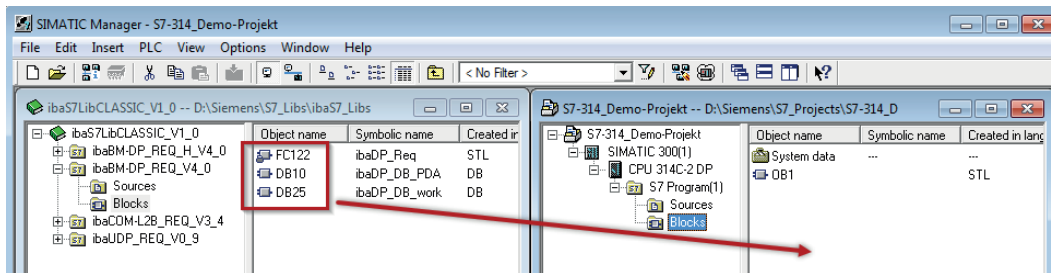
6.1.1.2 Using the blocks in SIMATIC Manager

There are two options for using the blocks from the library.

- Display the block library and drag the required blocks to the opened destination block.



- Open the library via *File – Open – Libraries*, and the required destination project in parallel. With the <Tile horizontal> button, both projects can be displayed side by side. You can either drag & drop the blocks or copy & paste them.



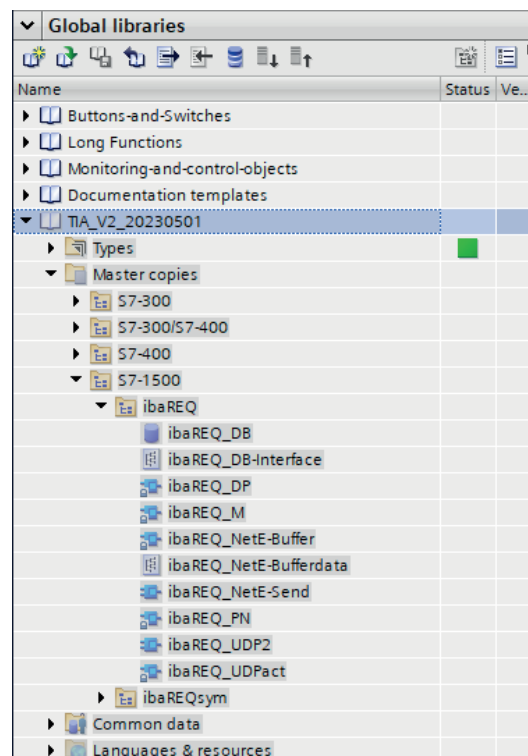
→ You can now use the blocks in the destination project.

6.1.2 iba S7 library for SIMATIC TIA portal

The iba S7 library for SIMATIC TIA Portal ("ibaS7LibTIA_Vx_y") is suitable for SIMATIC TIA Portal. It contains the Request blocks described in the manual, which are required for the use of *ibaPDA-Request-S7-UDP*.

You find the iba S7 library as an archived file on the data storage medium "iba Software & Manuals" in the following directory:

`\04_Libraries_and_Examples\10_Libraries\01_SIMATIC_S7\`



Note

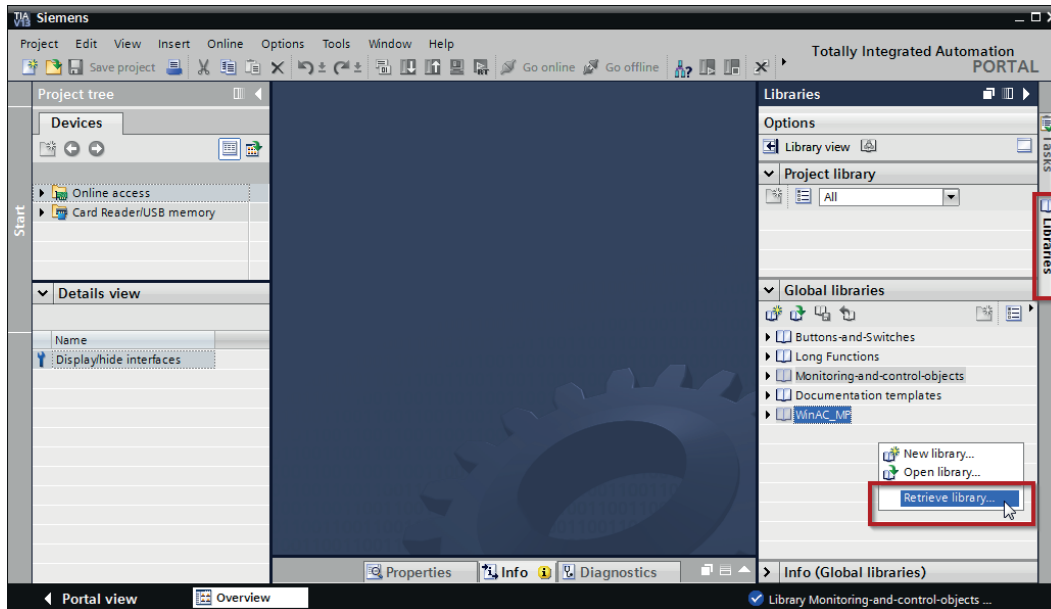


TIA portal libraries are version-dependent. There may be upward compatibility depending on the TIA portal version.

6.1.2.1 Integrating the library into TIA Portal

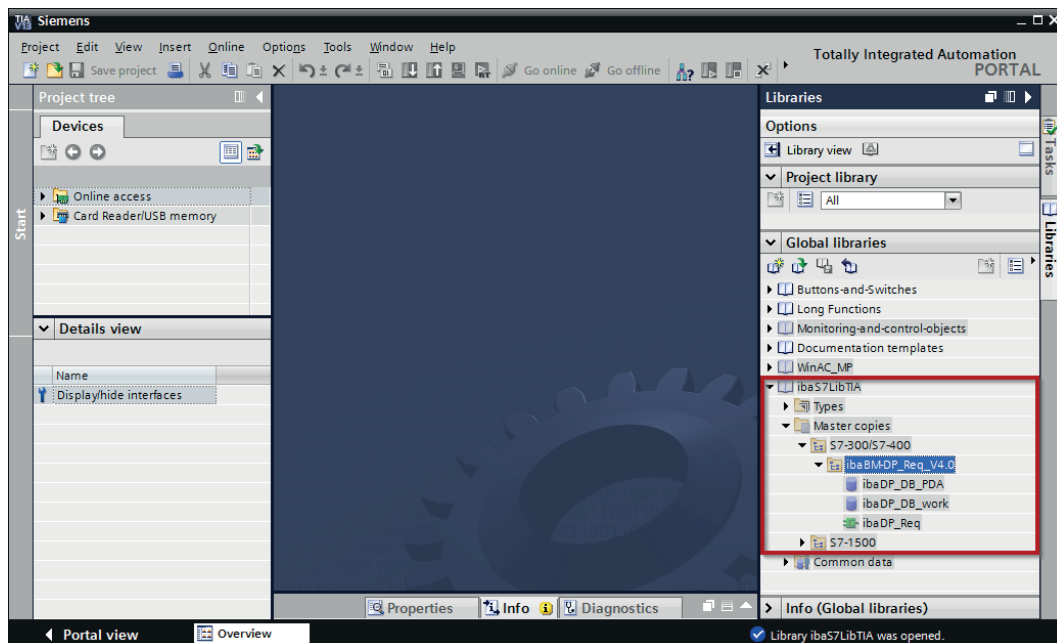
For integrating the library, you have to retrieve it in the TIA portal. Copy the iba S7 library to a local directory of your computer, where the TIA Portal is executed.

1. On the *Libraries* tab, select *Retrieve library* from the context menu.



2. Select the archive file of the iba S7 library and select a storage location for the extracted library in a next step.

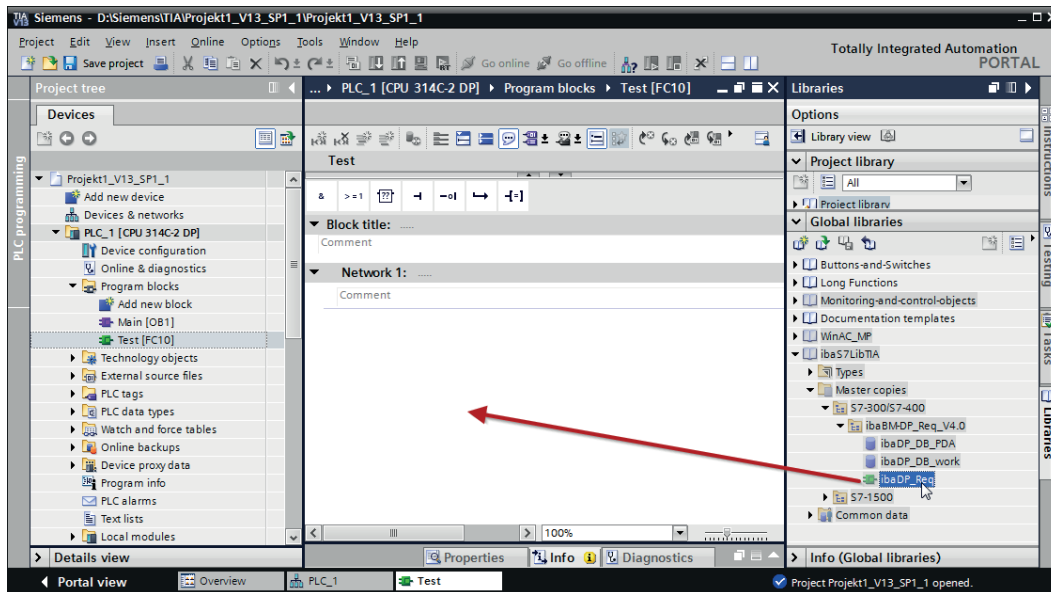
→ Now the library is integrated.



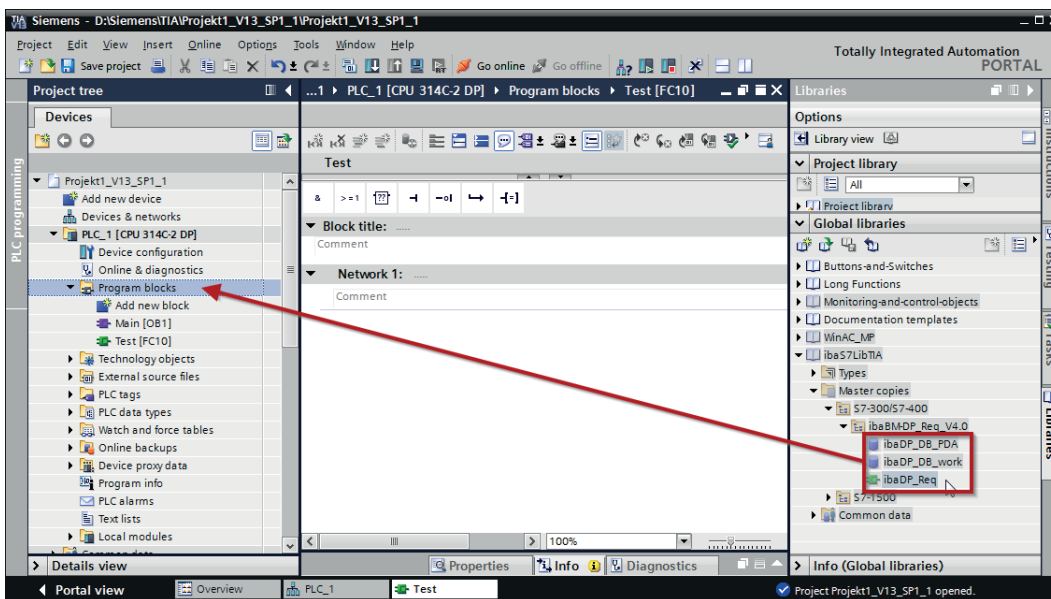
6.1.2.2 Using the blocks in TIA Portal

There are several options for using the blocks from the library.

- Display the block library and drag the required blocks to the opened destination block.

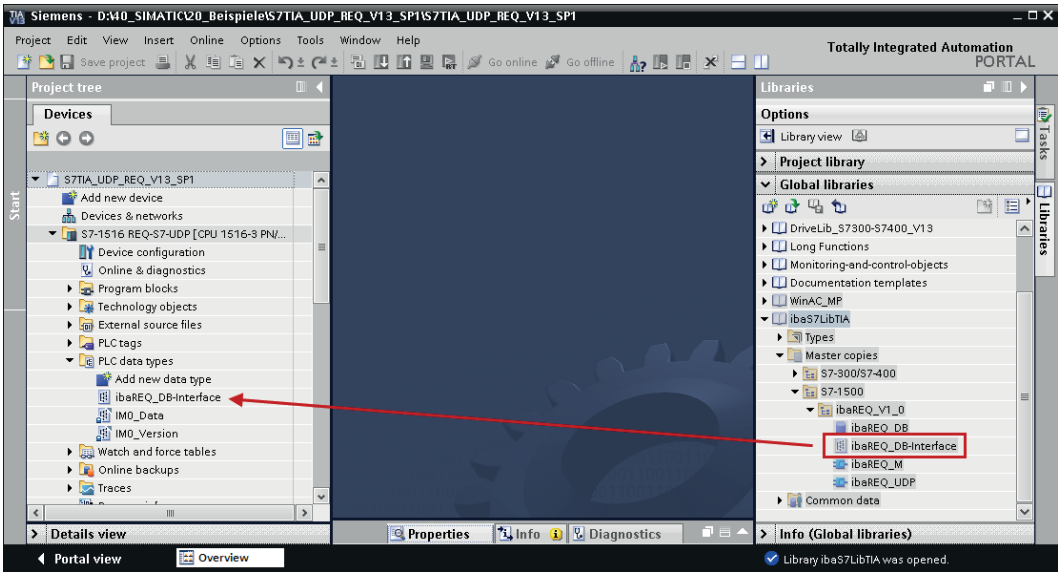


- Drag or copy the blocks to the program blocks directory in the project navigation.



→ You can call the blocks within a project block.

- Drag or copy the PLC data type to the PLC data type directory in the project navigation.



6.2 Application examples

You can find application examples for several different configurations on the storage medium "iba Software & Manuals".

\04_Libraries_and_Examples\50_ibaPDA-Interface-S7-TCP_UDP\Request-S7\

iba	S7-CPU	S7 project	ibaPDA project
ibaPDA-... ...Interface-S7-TCP/UDP + ...Request-S7-UDP	S7-300 PN	S7CLASSIC_ UDP_REQ_Vxx.zip	ibaPDA_S7CLASSIC_ UDP_REQ_Vxx.zip
	S7-300 + CP343-1 LEAN		
	S7-400 + CP443-1		
	S7-1500	S7TIA_UDP_REQ_ Vx_SPx_Vyy.zip	ibaPDA_S7TIA_UDP_ REQ_Vyy.zip
		ibaS7TIA_UDP_REQsym_ Vx_SPx_Vyy.zip	ibaPDA_S7TIA_UDP_ REQsym_Vyy.zip
		ibaS7TIA_UDP_REQ3sym_ Vx_SPx_Vyy.zip	

Table 4: Application examples on the storage medium

6.3 S7 cycle time measurements

The following tables show which code run times are needed by the Request blocks. The measurement values have been determined in a test environment and only serve as reference points. The values may deviate in other system environments.

SIMATIC S7-CPU	Number of signals	Data amount	ibaREQ_M FB140	ibaREQ_UDPint FB146
CPU412-2 PN 6ES7 412-2EK06-0AB0	1 INT + 0 BOOL (1 Pointer)	2 Byte	128 µs	510 µs
	59 REAL + 64 BOOL (2 Pointer)	244 Byte	132 µs	595 µs
	59 REAL + 64 BOOL (123 Pointer)	244 Byte	132 µs	1100 µs
	122 INT + 0 BOOL (1 Pointer)	244 Byte	132 µs	560 µs
	122 INT + 0 BOOL (122 Pointer)	244 Byte	132 µs	1112 µs
	512 INT + 512 BOOL (2 Pointer)	1088 Byte	132 µs	684 µs
	512 INT + 512 BOOL (1024 Pointer)	1088 Byte	132 µs	5502 µs
	366 REAL + 0 BOOL (1 Pointer)	1464 Byte	132 µs	700 µs
	366 REAL + 0 BOOL (366 Pointer)	1464 Byte	132 µs	2434 µs

SIMATIC S7-CPU	Number of signals	Data amount	ibaREQ_M FB1400	ibaREQ_UDP FB1405
CPU1516-3 PN/DP 6ES7 516-3AN00-0AB0	1 INT + 0 BOOL (1 Pointer)	2 Byte	195 µs	402 µs
	59 REAL + 64 BOOL (2 Pointer)	244 Byte	189 µs	421 µs
	59 REAL + 64 BOOL (123 Pointer)	244 Byte	195 µs	792 µs
	122 INT + 0 BOOL (1 Pointer)	244 Byte	189 µs	413 µs
	122 INT + 0 BOOL (122 Pointer)	244 Byte	195 µs	795 µs
	512 INT + 512 BOOL (2 Pointer)	1088 Byte	189 µs	431 µs
	512 INT + 512 BOOL (1024 Pointer)	1088 Byte	192 µs	2028 µs
	366 REAL + 0 BOOL (1 Pointer)	1464 Byte	189 µs	431 µs
	366 REAL + 0 BOOL (366 Pointer)	1464 Byte	196 µs	1586 µs

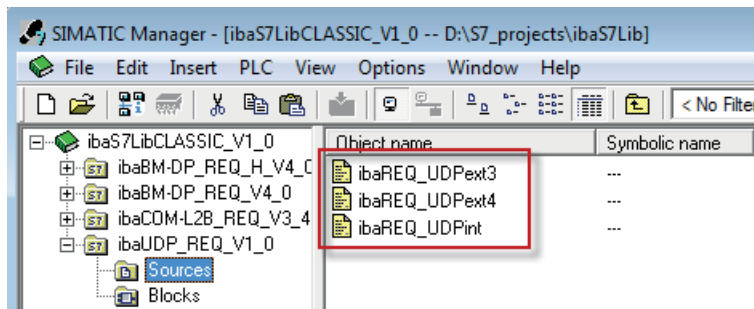
6.4 Adaptation to the renumbered system functions

This procedure is only necessary when using the SIMATIC Manager (STEP 7 ≤ V5), if a different block number was assigned to one of the following used embedded block numbers.

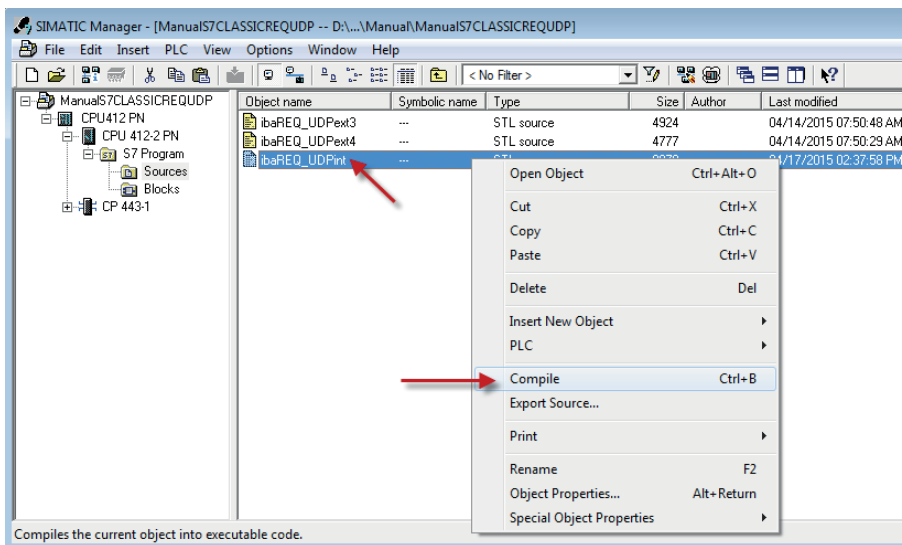
Symbolic name	Standard numbering	Origin
ibaREQ_UDPact	FB145	iba AG, ibaS7Lib
TCON	FB65	Siemens, Standard Library
TDISCON	FB66	Siemens, Standard Library
TUSEND	FB67	Siemens, Standard Library
AG_SEND	FC5	Siemens, SIMATIC_NET_CP
AG_LSEND	FC50	Siemens, SIMATIC_NET_CP
ibaUDT_UDPact	UDT145	iba AG, ibaS7Lib

Table 5: Subordinate blocks

1. Copy the following block sources from the iba S7 library to the source folder of your STEP 7 project.



2. Make a new translation for all sources of the function blocks you use.



Note



It is absolutely essential that the symbolic designation of the subordinate blocks is not modified (see table above).

6.5 Setting PG/PC interface/defining new access point

ibaPDA-Request-S7-UDP cannot establish a connection to a S7-CPU if the parametrization "AUTO" for an access point (MPI-adapter or CP) has been set in the SIMATIC Manager.

There are 2 possible remedies:

Changing the interface with remaining access point name

Change interface in the SIMATIC Manager e.g. from "CP5622 (AUTO)" to "CP5622 (MPI)" or "CP5622 (PROFIBUS)".

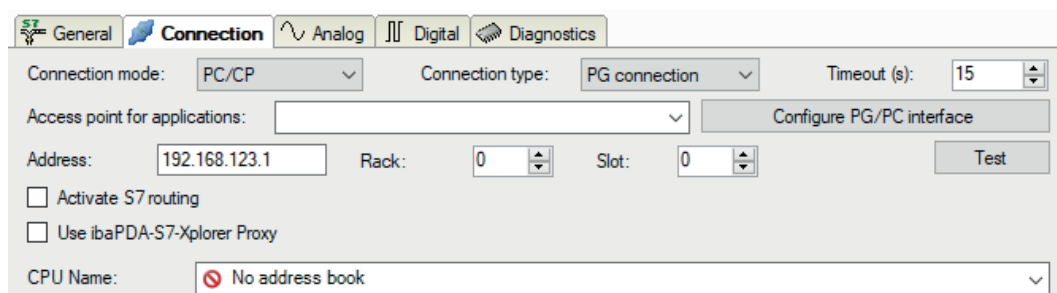
Disadvantage of this method: If the setting of the access point is changed again in the SIMATIC Manager, the measurement does no longer work because *ibaPDA* has no longer access.

Adding a special access point for ibaPDA

To avoid conflicts with the setting of SIMATIC Manager and *ibaPDA* when both programs run on the same computer, you can define a new access point.

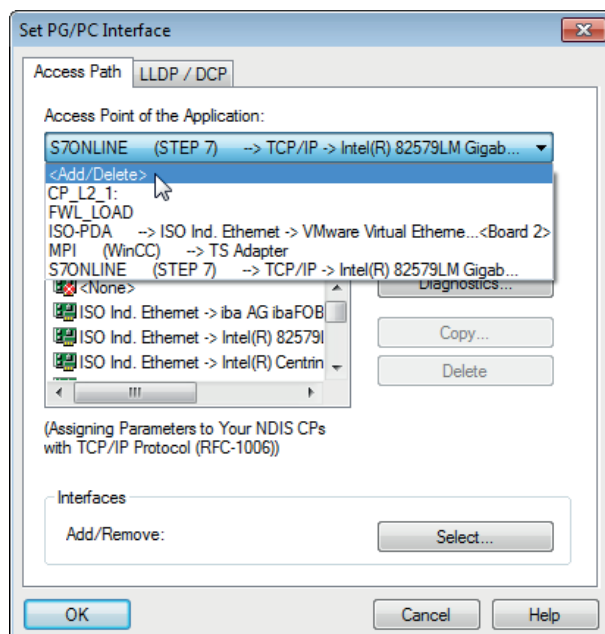
There is the <Configure PG/PC interface> button in the dialog window of the module. With this button, you can open the dialog for configuring the PG/PC interface.

The setting for the SIMATIC Manager is also changed.

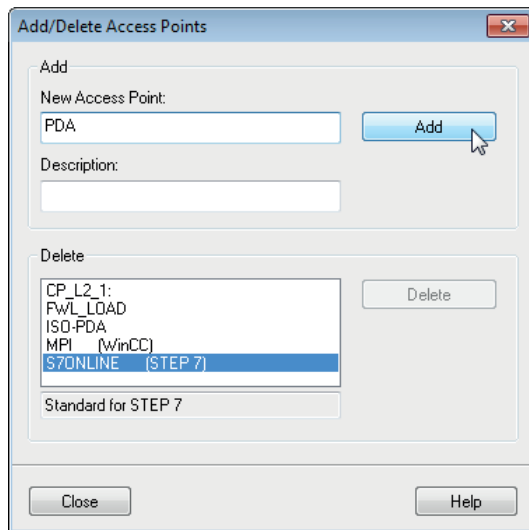


Procedure

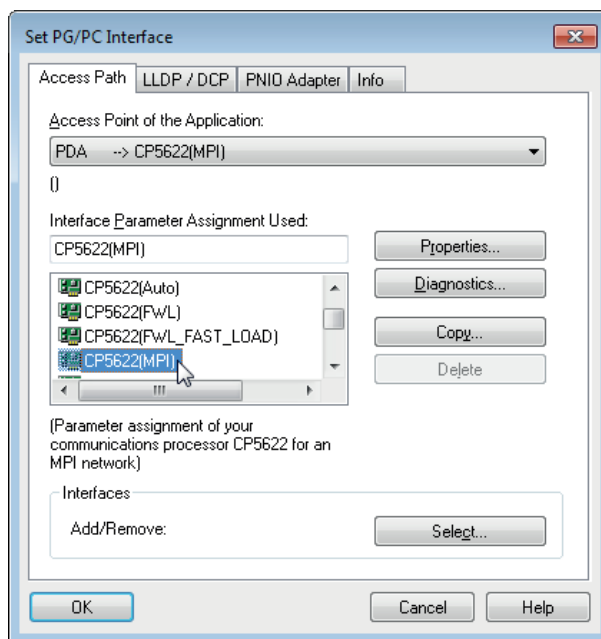
1. Open the dialog for configuring the PG/PC interface with the <Configure PG/PC interface> button.
2. Under *Access Point of the Application* select the row <Add/Delete>.



- Define a new access point: Enter a name, e.g. "PDA", and optionally a description for a better understanding.
Confirm your entries with <Add> and <Close>.

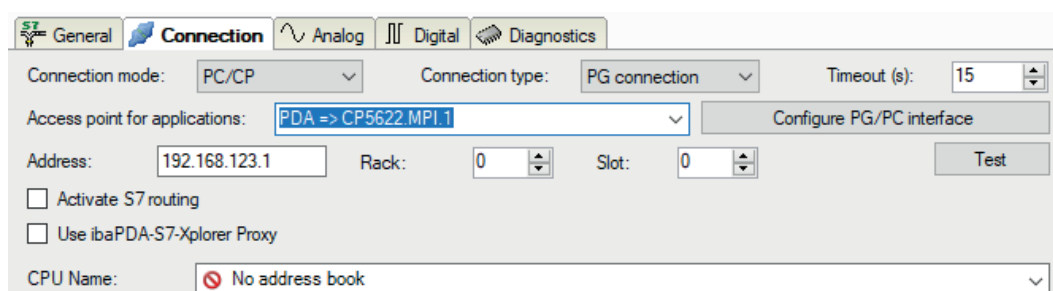


- Add an interface to the access point, e.g. "CP5622 (MPI)".



- Exit the configuration with <OK>.

→ *ibaPDA* subsequently displays the newly defined access (e.g. "PDA --> CP5622.MPI.1") in the connection dialog under *Access points for applications*.



Notes on the different access points

Depending on which access points have been configured in the engineering computer, there are different access points available for selection in the *ibaPDA* system.

Basically, there are 3 types of access points:

- TCP/IP
- ISO
- Bus system PROFIBUS or MPI

TCP/IP

If you select an access point using TCP/IP, you need to enter the IP address, rack number and slot number of the CP in the module configuration dialog. If you do not know the rack number or slot number, enter "0" for slot and click on the <Test> button.

ISO

If you select an access point using an ISO interface, you need to enter the MAC address, rack number and slot number. If you do not know the rack number or slot number, enter "0" for slot and click on the <Test> button.

Bus system PROFIBUS or MPI

If you select an access point using a bus interface, e.g. PROFIBUS or MPI, you need to enter the bus address, the rack number and slot number. You can also use the <Test> button and then click on one of the CPU links found to test the connection.

6.6 S7 routing

S7 routing is defined as the possibility to use S7 controls as router to access secondary target systems, i.e. controls or drives, which are in different subnets. This also includes changing the bus system (Ethernet, PROFIBUS, MPI).

Reference



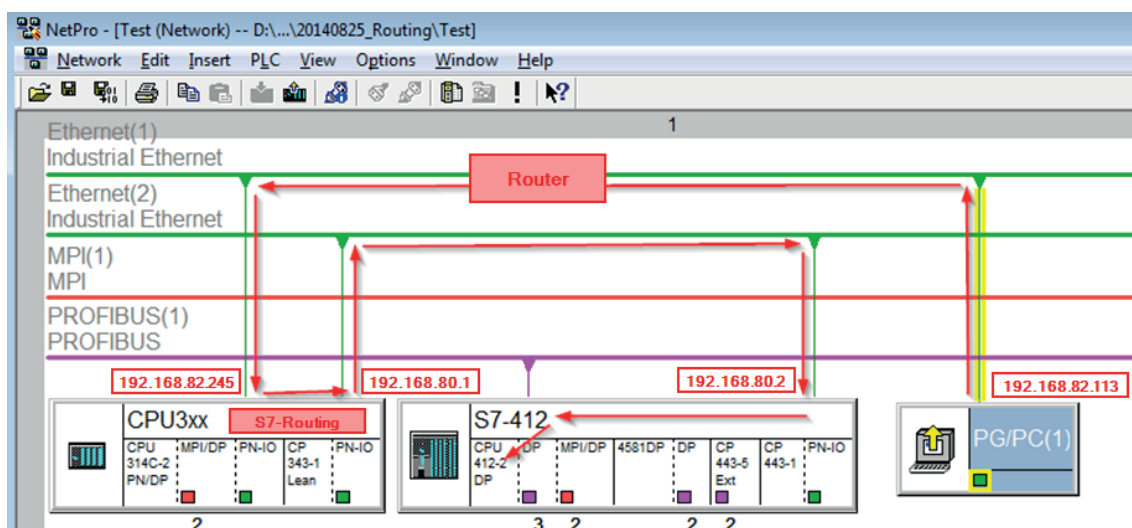
For more information about the S7 Routing, refer to:

- Which modules support the "S7 Routing" function in S7 subnets?
<https://support.automation.siemens.com/ww/view/en/584459>
- Which requirements must be fulfilled and what do I have to observe if I want to execute routing?
<https://support.industry.siemens.com/cs/ww/en/view/2383206>
- How do you enable cross-project S7 Routing in the TIA Portal and in STEP 7 V5.x?
<https://support.industry.siemens.com/cs/ww/en/view/109474569>

6.6.1 Routing from Ethernet to Ethernet

Do not mix up the *S7 Routing* function with IP routing.

The example shows how to implement the following way of access via S7 Routing in NetPro.



The engineering computer (also with *ibaPDA*) is to access the CPU412 controller. The computer and the controller are not directly connected via a common network/bus. The connection has to run over the CPU314C controller.

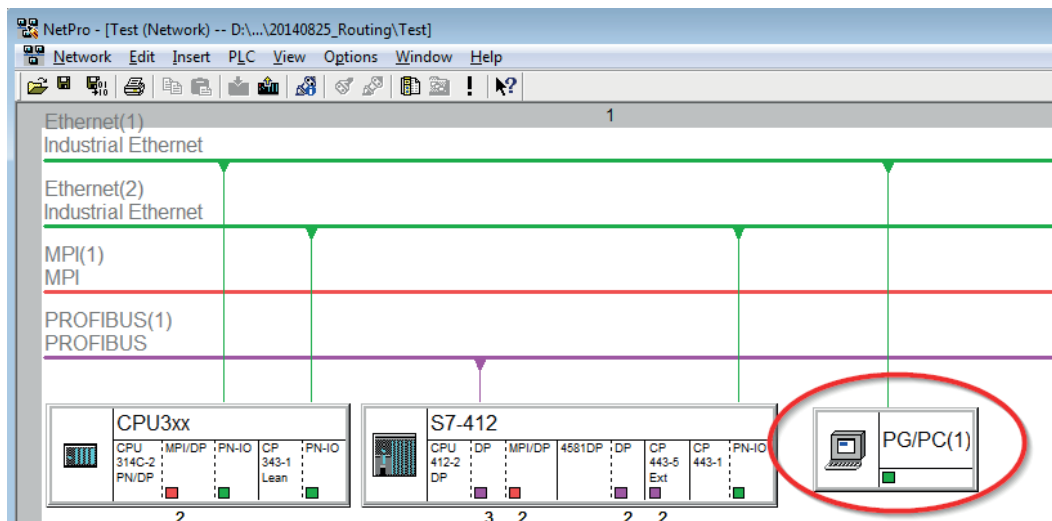
"Passing" the communication through this controller is called *S7 Routing*.

In our example, engineering computer and CPU314C are also located in two different (logic) subnets. You need an (IP) router for establishing a communication connection. This is completely independent of the S7 Routing function and should not be mixed up with it.

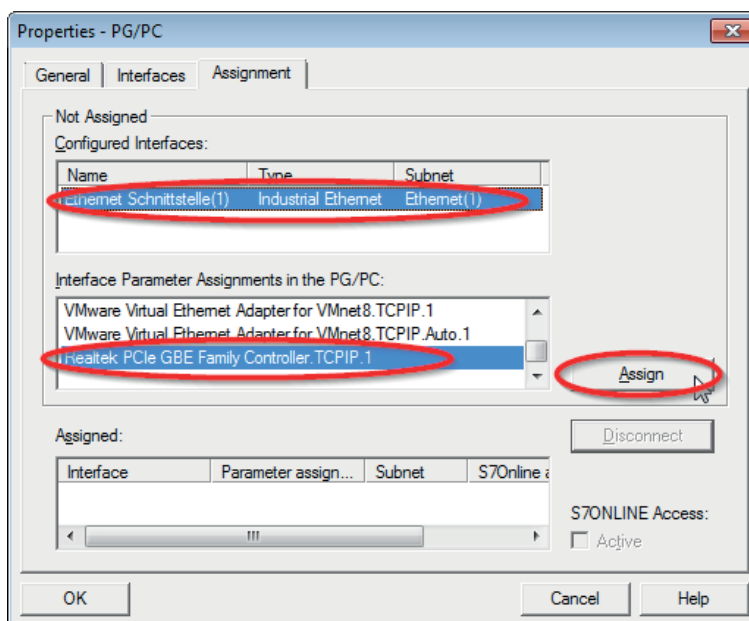
6.6.1.1 Configuration of STEP 7/NetPro

The following configuration steps are required to be able to access the secondary CPU412 control with the SIMATIC STEP 7 programming software. For SINUMERIK, SINAMICS, or SIMOTION, you can apply similar steps. For using *ibaPDA*, these configuration steps are not required.

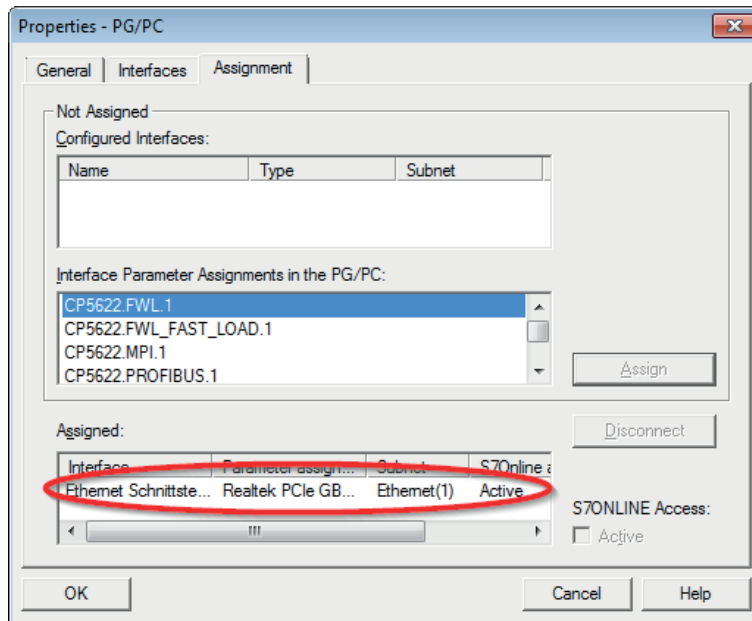
1. Add a PG/PC station and configure it.



2. Assign an interface (network card).

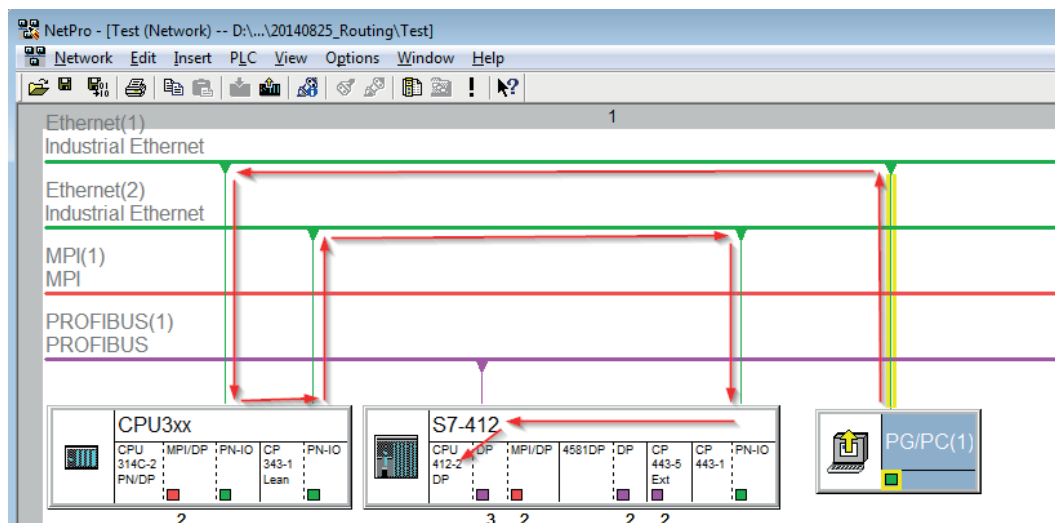


→ Result:



Now, the connection line from PG/PC to the network has to be marked in yellow.

In the following figure, the communication path is shown using arrows (these are not displayed in SIMATIC NetPro).

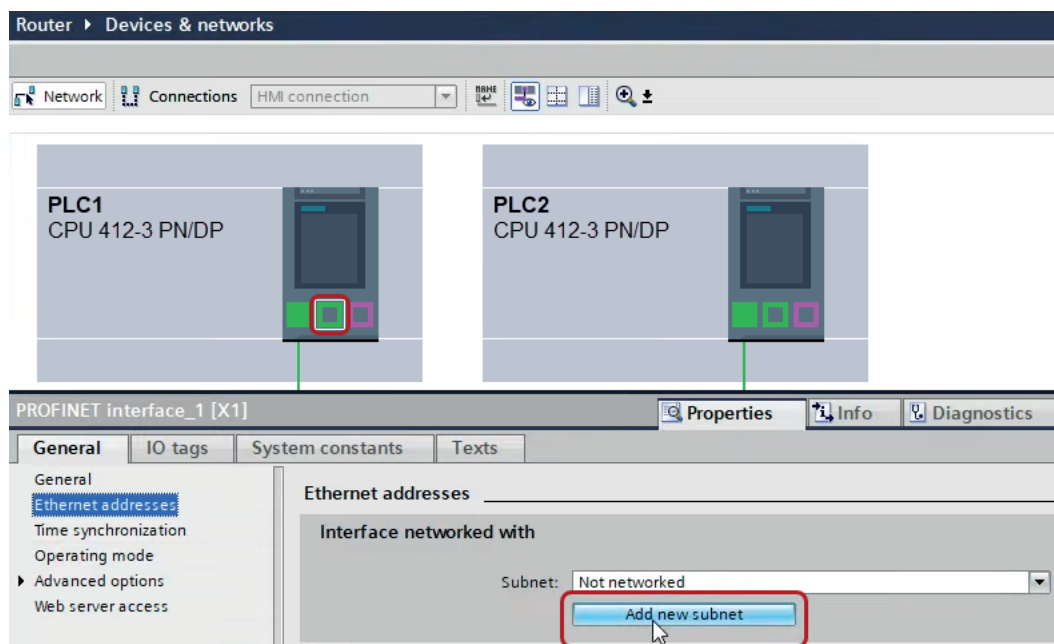


3. Finally, load all hardware configurations and connection data from NetPro.

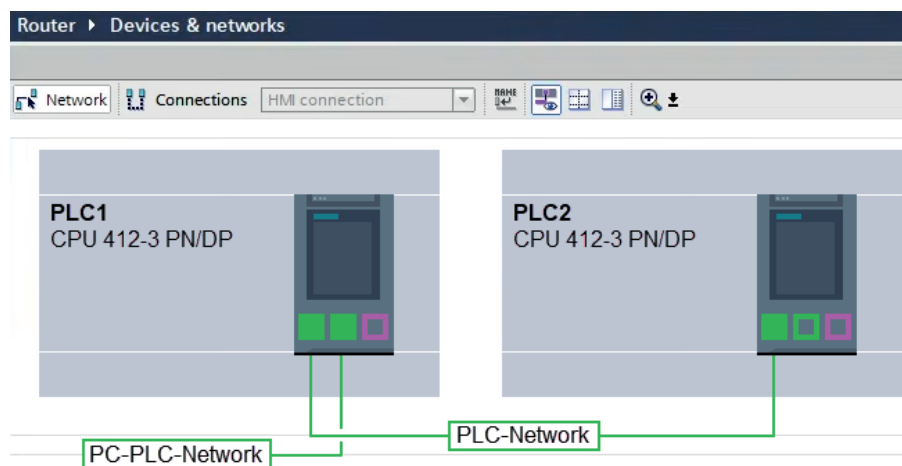
6.6.1.2 Configuration of TIA Portal

The following configuration steps are required exclusively for accessing the subordinate control "PLC2" by the programming software TIA 7. For SINUMERIK, SINAMICS, or SIMOTION, you can apply similar steps. For using *ibaPDA*, these steps are not required.

1. Connect both controllers in TIA Portal via the Ethernet ports.
2. Establish a connection with your computer and the first controller "PLC1" by adding a subnet.

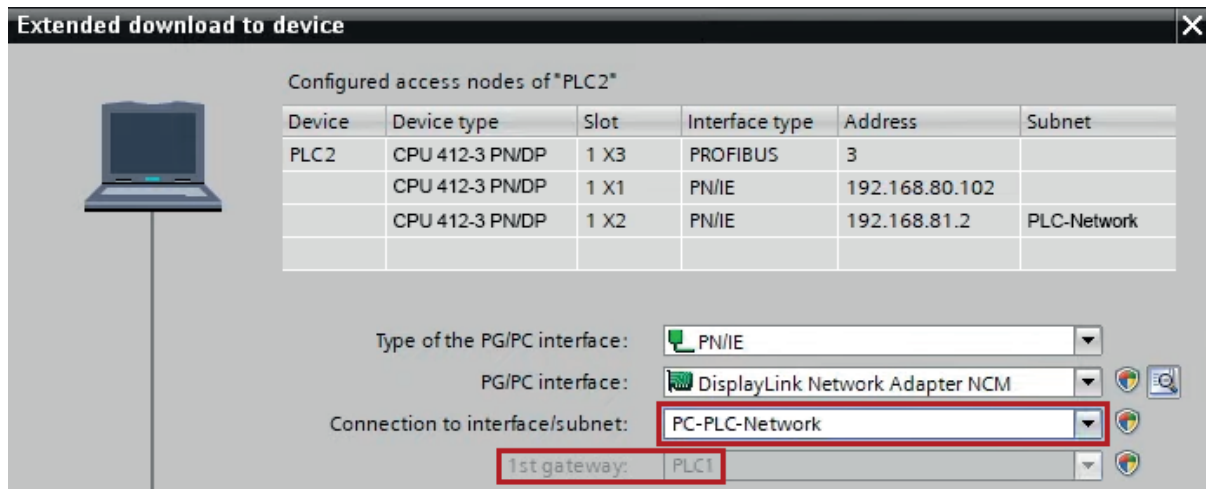


→ The connections between the controllers and to the computer are shown in TIA Portal.



3. Download the programming of the controller "PLC1" and then the programming of the controller "PLC2".

4. In the controller "PLC2", set the connection from "PLC1" to the computer as the connection to the interface/subnet.

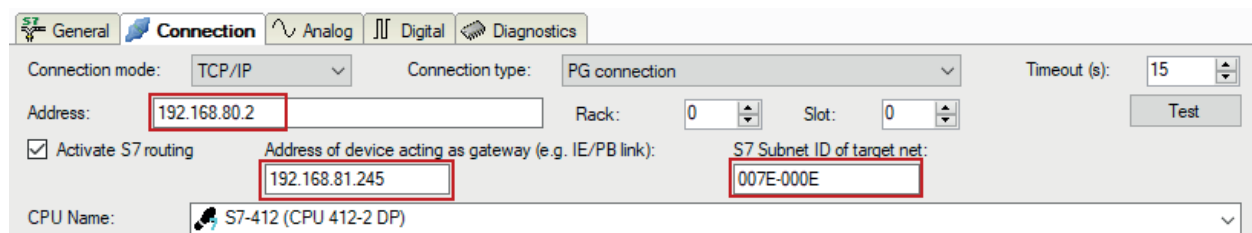


→ "PLC1" appears as the first gateway.

→ "PLC2" is now connected to the computer via S7 routing via "PLC1".

6.6.1.3 Configuration of ibaPDA

Configure the following settings.



Activate S7 routing

Enable this option to use S7 routing.

Address

Enter the address of the target control (here CPU412)

Address of device acting as gateway

Enter the address of the gateway (here CPU314C).

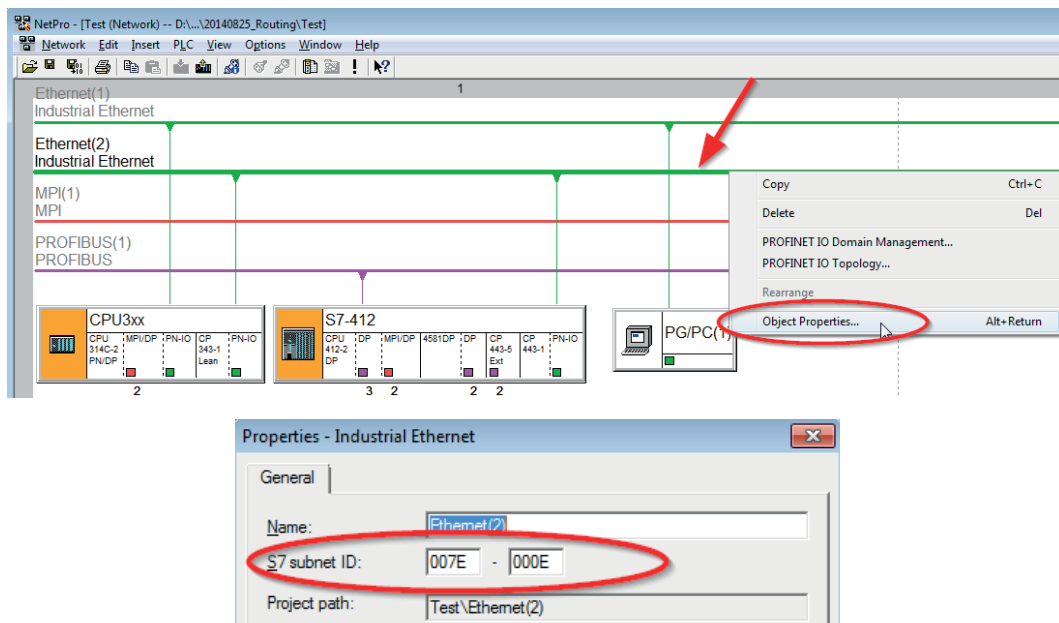
S7 subnet ID of target net

Enter the subnet ID from STEP 7 NetPro or TIA Portal.

Identifying the S7 subnet ID in NetPro

You can identify the S7 subnet ID in NetPro.

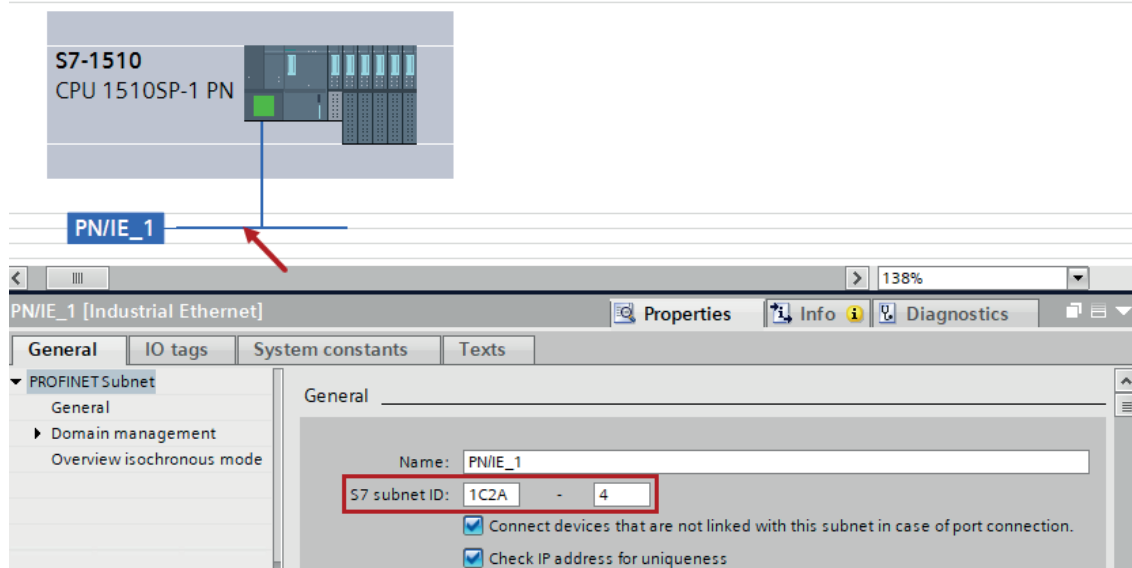
Right-click on the secondary bus system and open the *Object Properties*.



Identifying the S7 subnet ID in TIA Portal

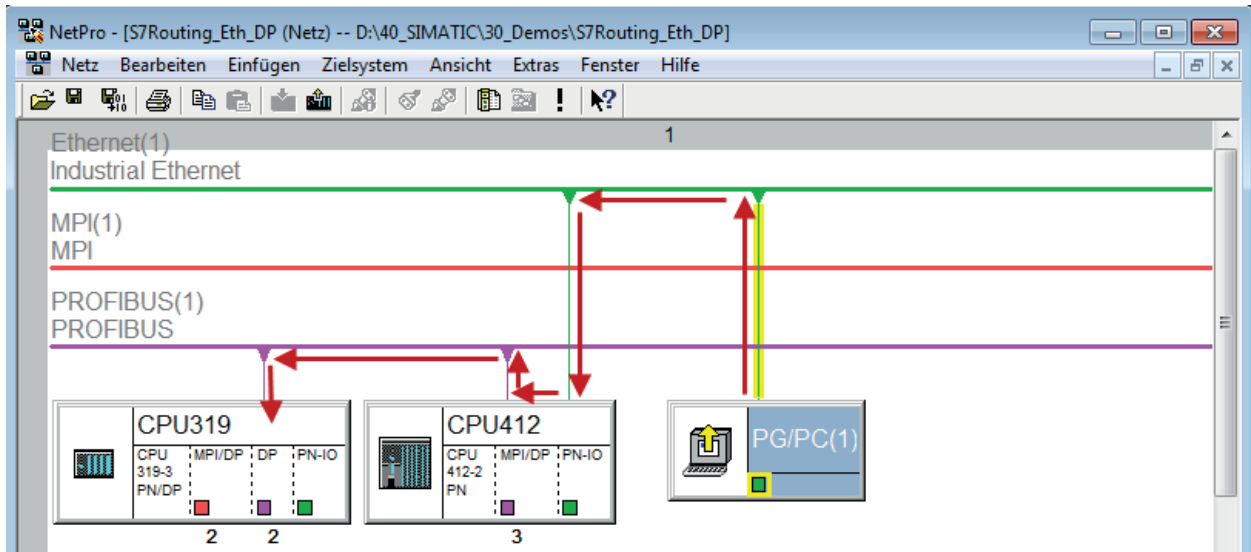
You can identify the S7 subnet ID in TIA Portal.

Click on the bus system and go to *Properties – General – General*.



6.6.2 Routing from Ethernet to PROFIBUS

The example shows how to implement the following way of access via S7 Routing and an example system topology for Ethernet PROFIBUS in NetPro.



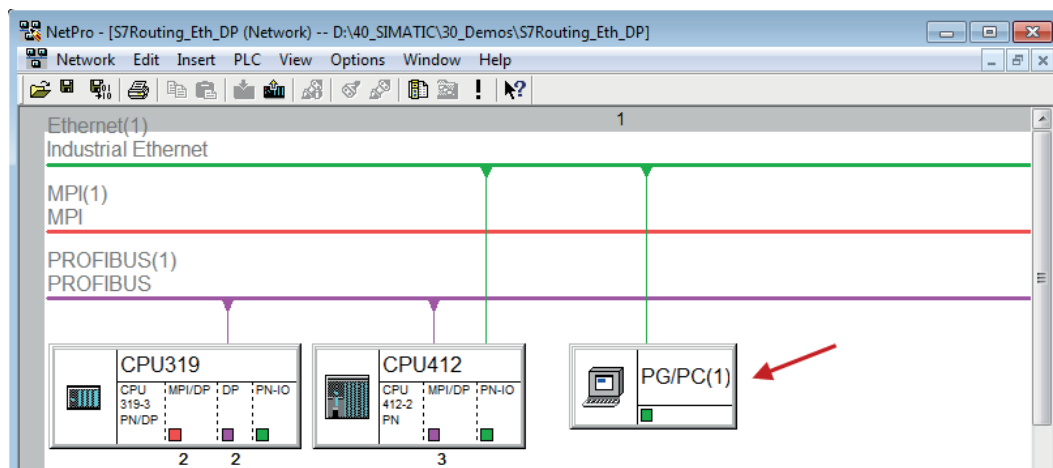
The engineering computer (also with *ibaPDA*) is to access the CPU319 controller. The computer and the controller are not directly connected via a common network/bus. The connection has to run over the CPU412 controller.

"Passing" the communication through this controller is called *S7 Routing*.

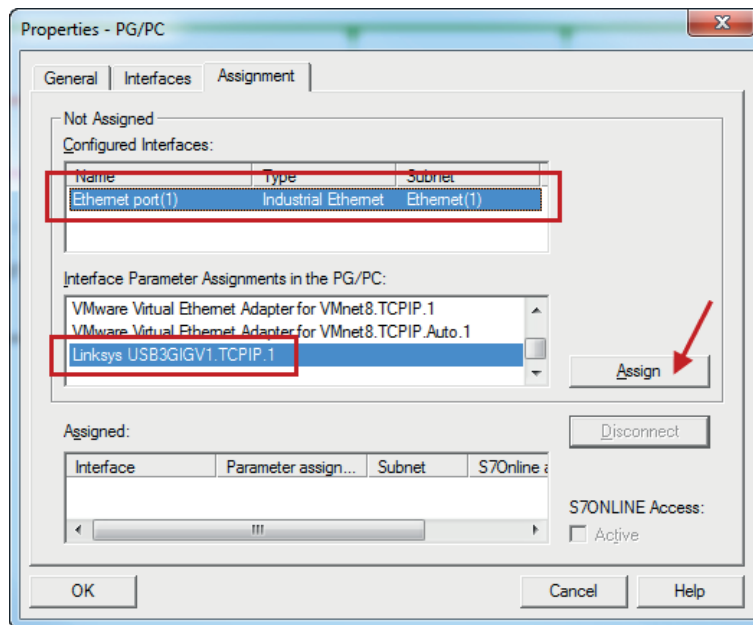
6.6.2.1 Configuration of STEP 7/NetPro

The following configuration steps are exclusively required for accessing the subordinate controller CPU319 via the SIMATIC STEP 7 programming software. For SINUMERIK, SINAMICS, or SIMOTION, you can apply similar steps. For using *ibaPDA*, these configuration steps are not required.

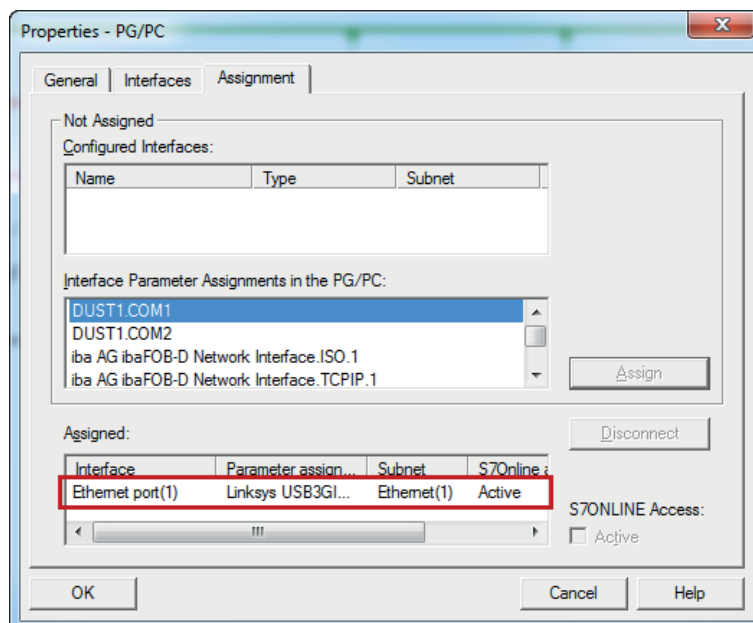
1. Add a PG/PC station and configure it.



2. Assign an interface (network card).

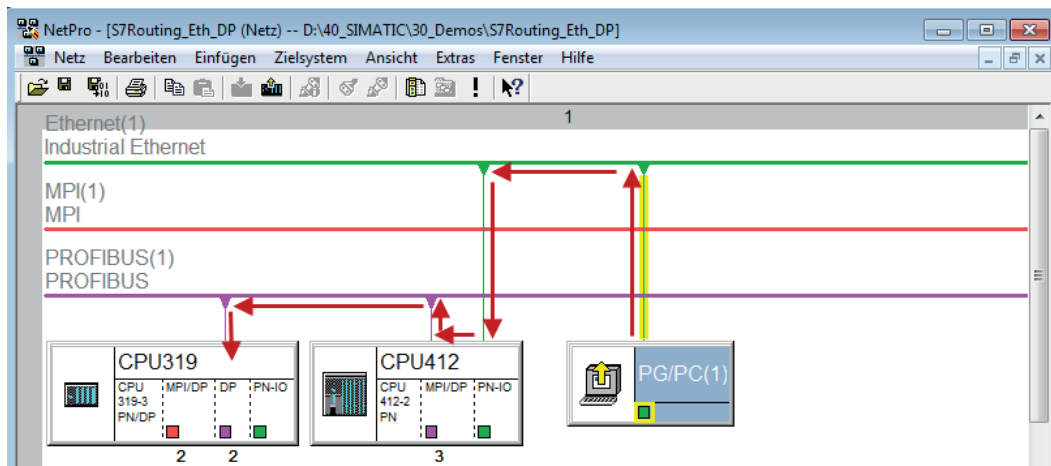


→ Result:



Now, the connection line from PG/PC to the network has to be marked in yellow.

In the following figure, the communication path is shown using arrows (these are not displayed in SIMATIC NetPro).



3. Finally, load all hardware configurations and connection data from NetPro.

6.6.2.2 Configuration of TIA Portal

The configuration steps are required exclusively for accessing the subordinate control "PLC2" by the programming software TIA Portal. For SINUMERIK, SINAMICS, or SIMOTION, you can apply similar steps.

For the configuration of PROFIBUS proceed as described for Ethernet, see ➤ *Configuration of TIA Portal*, page 123.

6.6.2.3 Configuration of ibaPDA

Configure the following settings.

Connection mode: TCP/IP Connection type: PG connection Timeout (s): 15

Address: 2 Rack: 0 Slot: 2 Test

☒ Activate S7 routing Address of device acting as gateway (e.g. IE/PB link): 192.168.50.95 S7 Subnet ID of target net: 02D6-000B

CPU Name: No address book

Activate S7 routing

Enable this option to use S7 routing.

Address

Enter the address of the target control (here CPU319).

Address of device acting as gateway

Enter the address of the gateway (here CPU412).

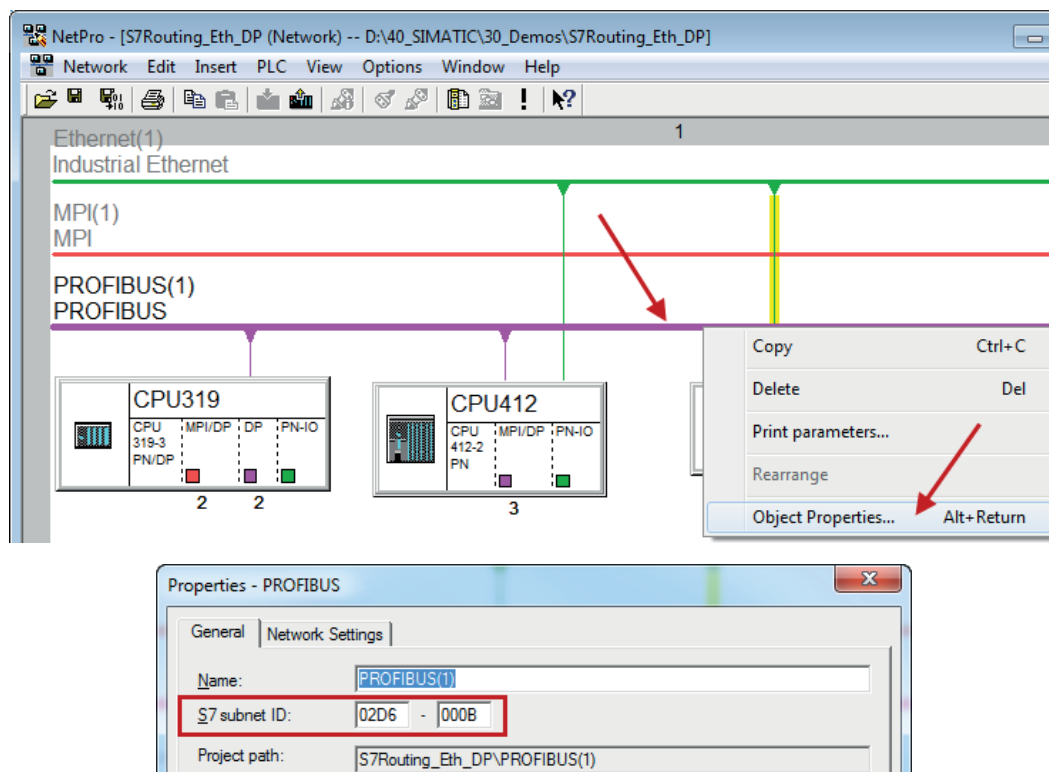
S7 subnet ID of target net

Enter the subnet ID from STEP 7 NetPro.

Identifying the S7 subnet ID in NetPro

You can identify the S7 subnet ID in NetPro.

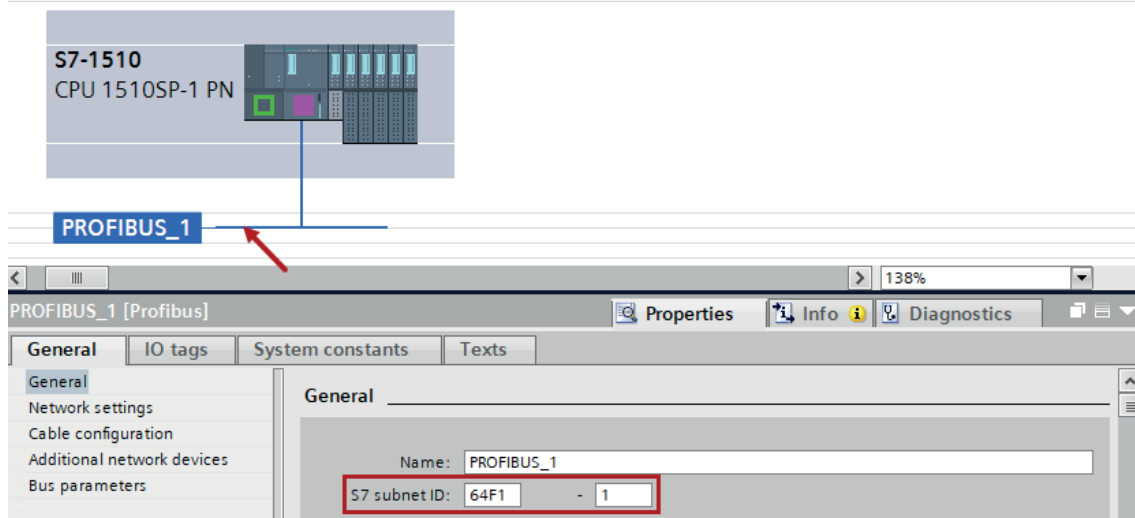
Right-click on the secondary bus system and open the *Object Properties*.



Identifying the S7 subnet ID in TIA Portal

You can identify the S7 subnet ID in TIA Portal.

Click on the bus system and go to *Properties – General – General*.



7 Support and contact

Support

Phone: +49 911 97282-14
Email: support@iba-ag.com

Note



If you need support for software products, please state the number of the license container. For hardware products, please have the serial number of the device ready.

Contact

Headquarters

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany

Phone: +49 911 97282-0
Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and Worldwide

For contact data of your regional iba office or representative please refer to our web site:

www.iba-ag.com