ibaPDA-Request-S7-UDP

Request Data Interface to SIMATIC S7 via UDP

Manual

Issue 1.3

**Measurement Systems
for Industry and Energy**

The content of this publication has been checked for compliance with the described hardware and software. Nevertheless, deviations cannot be excluded completely so that the full compliance is not guaranteed. However, the information in this publication is updated regularly. Required corrections are contained in the following issues or can be downloaded from the Internet.

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

Issue	Date	Revision	Author	Version SW
1.3	Aug 2018	div. adaptions (e.g. Port 102 notice, Request blocks etc.)	RM	6.39

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

SIMATIC S7-200®, S7-300®, S7-400®, S7-1200®, S7-1500® and LOGO!® are labels and registered trademarks of Siemens AG.

Table of Contents

1 About this manual

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

The product *ibaPDA-Request-S7-UDP* is an extension of *ibaPDA* for random access to S7 symbols and S7 operands when recording data from SIMATIC S7 CPUs. For transferring data, the *UDP*[1] network protocol is used. In this manual, only the extensions and differences are shown. For all other functions and operating options, please see the *ibaPDA* and *ibaPDA-Interface-S7-TCP/UDP* manuals.

Other documentation

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

1.1 Target group and previous knowledge

This documentation addresses qualified professionals, who are familiar with handling electrical and electronic modules as well as communication and measurement technology. A person is regarded as a professional if he/she is capable of assessing the work assigned to him/her and recognizing possible 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 of *ibaPDA-Request-S7-UDP* the following basic knowledge is required and/or useful:

❑ Windows operating system

❑ Basic knowledge *ibaPDA*

❑ Basic knowledge network technology

❑ Knowledge of configuration and operation of SIMATIC S7 PLCs

---

[1] User Datagram Protocol

1.2   Notations

In this manual, the following notations are used:

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

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:

❑ From an electric shock!

❑ Due to the improper handling of software products which are coupled to input and output procedures with control function!

⚠ WARNING

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

⚠ CAUTION

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

Note

A note specifies special requirements or actions to be observed.

Important note

Note if some special features must be observed, for example exceptions from the rule.

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.

Example

Configuration and application examples for a better understanding

2   System requirements

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

- ❑ *ibaPDA* V6.35 or higher
- ❑ Basic license for *ibaPDA*
- ❑ Additional license for *ibaPDA-Interface-S7-TCP/UDP*
- ❑ Additional license for *ibaPDA-Request-S7-UDP*
- ❑ SIMATIC S7 controller S7-300, S7-400, S7-400H, S7-1500, WinAC
- ❑ in case PC/CP connections are used:
  - ▪ SIMATIC STEP 7 or SIMATIC NET, or
  - ▪ SIMATIC TIA Portal
- ❑ SIMATIC CFC (V6.0 or higher), if signals are to be chosen by Drag & *Drop*[2].

For integrating the Request blocks in the S7 program:

- ❑ SIMATIC STEP 7 V5.4 SP5 or higher, or
- ❑ SIMATIC STEP 7 (TIA Portal) V13 SP1 or higher

For more requirements concerning the used PC hardware and the supported operating systems, please see the *ibaPDA* documentation.

System restrictions

- ❑ Restrictions SIMATIC TIA-Portal:
  - ▪ SIMATIC TIA-Portal STEP 7 V13 or higher is a prerequisite for generating address books from TIA Portal projects.
- ❑ The access to optimized data blocks of S7-1500 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 FB function blocks 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, CHAR

All 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.

---

[2] Can only be used in combination with SIMATIC STEP 7 V5.x

Licenses

Order No.	Name	Description
31.001040	ibaPDA-Interface-S7-TCP/UDP	Extension license for an ibaPDA system by a TCP/IP and UDP/IP interface Number of connections: 64
31.101040	one-step-up-Interface-S7-TCP/UDP	Extension license for the extension of an existing ibaPDA-Interface-S7-TCP/UDP interface by another 64 S7-TCP/UDP connections, a max. of 3 permitted
31.001311	ibaPDA-Request-S7-UDP	Extension license for an ibaPDA system for using the Request functionality with the ibaPDA-Interface-S7-TCP/UDP interface Number of connections: 2
31.101311	one-step-up-Request-S7-UDP	Extension license for extending an existing ibaPDA-Request-S7-UDP by another 2 Request-S7-UDP connections, a max. of 127 permitted.

Table 1:   License overview, version ibaPDA V6.35

The use of the *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

3.1 General information

The *ibaPDA-Request-S7-UDP* interface is suitable for the measurement data acquisition with random access over standard network cards via UDP. The measurement data are 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 serve for sending the actual values of the S7 operands selected by the user within *ibaPDA* cyclically for recording to *ibaPDA*. When modifying the signal selection, no modifications in the S7 program are needed.

The signals to be measured are selected either by means of the absolute operand address or by means of the symbolic name using the *ibaPDA* address book browser. This browser allows accessing all defined symbols of the connected STEP 7 project.

When using the SIMATIC CFC editor on the same PC, the signals and connectors that are to be measured can be configured via drag & drop from the control program.
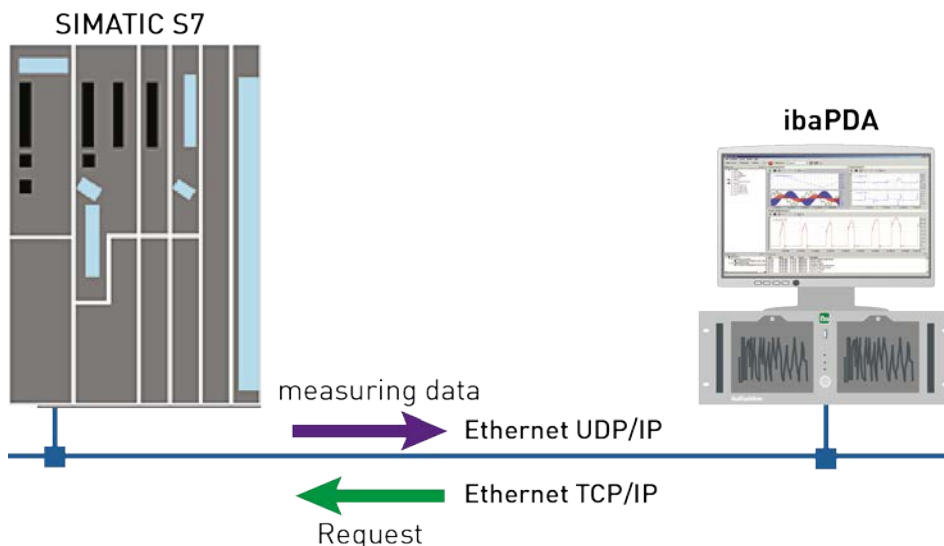
Figure 1:   System topology

On side of the SIMATIC S7, a PROFINET interface integrated on the CPU can be used 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 of a higher performance than communication processors, as 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 see the Siemens device and system manuals.

3.1.1 How does the symbolic Request work?

The S7-CPU generally works with operand addresses. This requires a mapping table between symbols and operands to be created to enable the *ibaPDA* user to select symbolic variables. When selecting a symbol, *ibaPDA* will simultaneously assign the suitable operand addresses.

The mapping table of the symbols to the operand addresses - the address book - is generated using the S7 address book generator that is integrated in *ibaPDA*.

One part of the address book (symbol table and data blocks) is generated by means of direct evaluation of the STEP 7 project. When using SIMATIC CFC, the SCL code is analyzed that is generated by the CFC Compiler. This SCL code is stored in the STEP 7 project in the "Sources" folder. One look at the SCL code shows that all temporary calculation results at the connectors of the function blocks are stored in data blocks. The address book generator uses this information to create the mapping between block connectors and data blocks.

The browser integrated in the I/O Manager of *ibaPDA* then uses the address book to select the signals.

3.1.2   Maximum amount of data

For each module, the amount of data that can be transferred is limited by different factors:

1. Maximum telegram length

For each telegram and hence for each connection, a maximum of 1466 Bytes of user data can be transferred.

2. Maximum number of pointers

The data to be transferred are defined by means of ANY pointers, which are transmitted by *ibaPDA* to the controller. Operands which belong together (e.g. consecutive addresses) are represented by a common pointer.

❑ A varied number of pointers can be managed, depending on the size of the REQ_DB. For S7-300, S7-400, WinAC controllers, you can freely select the length:
5280 Bytes: up to 128 pointers
9120 Bytes: up to 512 pointers
14240 Bytes: up to 1024 pointers (maximum)

❑ For S7-1500 controllers, there is only one fixed length available:
9120 Bytes: up to 512 pointers

Example:
If you use a REQ_DB with a length of 9120 Bytes, you can use up to 512 pointers. These pointers may occupy an address range of 1466 Bytes. If you want to acquire 512 distributed single bytes of operands, you will need all 512 pointers. Only 512 Bytes can be acquired, although the maximum telegram length of 1466 Bytes has not been reached, yet. In case all 512 Byte operands occupy consecutive addresses, you only need one pointer. In this case, 511 pointers are still available. With these pointers, you can address the remaining 1466 Bytes - 512 Bytes = 954 Bytes.
On the *S7 Request-Info* tab, under diagnostics, you can see the maximum telegram length and the current maximum number of pointers.

↗ See *Module diagnostics*, page 58.

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 (STEP 7 Version ≥ V13).

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

On the SIMATIC side, generally the following configuration steps have to be carried out:

❑ Configuration software (STEP 7 V5):
Integration of the Request blocks in the S7 program

❑ Configuration of the connection:
If a communication processor CP x43-1 is being used, a programmed connection has to be configured in NetPro. When using a PN interface integrated on the CPU, this step can be left out.

3.2.1.1   Description of the Request blocks

The communication between the S7 and *ibaPDA* is initialized and controlled with these blocks.

One set of Request blocks has to be called for each Request module (connection) in *ibaPDA*. The used blocks are part of the iba S7 library (see chapter *iba S7 library*, page 69).

Depending on the system configuration, different Request block combinations have to be used:

Request block	CPU with integrated PN interface or WinAC RTX	S7-300 CPU + CP343-1	S7-400 CPU + CP443-1	recommended call-up 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	-

Table 2:   Combinations of Request blocks

The following blocks always have to be used:

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

❑ ibaREQ_UDPact (provides the current signal values)
This block provides the current signal values in the call up cycle. The block is called internally in the blocks ibaREQ_UDPint, ibaREQ_UDPext3, or ibaREQ_UDPext4. This is why the block always has to be present in the project, but does not always has to be called separately.

❑ ibaREQ_DB (interface DB)
This DB serves as interface to *ibaPDA* and between the different Request blocks.

The following blocks are to be used depending on the S7 system configuration:

❑ ibaREQ_UDPint
This block sends the provided current signal values via an integrated PN interface.

❑ ibaREQ_UDPext3
This block sends the provided current signal values via an external communication processor CP343-1.

❑ ibaREQ_UDPext4
This block sends the provided current signal values via an external communication processor CP443-1.

The blocks ibaREQ_UDPint, ibaREQ_UDPext3 and ibaREQ_UDPext4 always have to be used alternatively.

ibaREQ_M (FB140)

Description of the formal parameters:

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the ibaPDA communication interface ibaREQ_DB
RESET	IN	BOOL	FALSE -> no reset performed (Standard) TRUE -> reset performed
INP_RANGE	IN	INT	Number of input bytes (evaluation only when initializing), 0: automatic detection (recommended)
OUT_RANGE	IN	INT	Number of output bytes (evaluation only when initializing), 0: automatic detection (recommended)
MARKER_RANGE	IN	INT	Number of marker bytes (evaluation only when initializing), 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**

Via this DB, the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

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 will be evaluated according to the length.

❑ **RESET**

Serves for manually resetting the Request blocks. All Request blocks of a combination are automatically reset together. Usually, the input does not have to be connected.

❑ **INP_RANGE**

Limits the number of input bytes to be measured.
For INP_RANGE = 0 the size of the available processor image of the inputs is determined by the Request FB itself (recommended). The evaluation is performed only during the initialization phase of the function block.

❑ **OUT_RANGE**

Limits the number of output bytes to be measured.
For OUT_RANGE = 0 the size of the available processor image of the outputs is determined by the Request FB itself (recommended). The evaluation is performed only during the initialization phase of the function block.

❑ **MARKER_RANGE**

Limits the number of marker bytes to be measured.
For MARKER_RANGE = 0 the number of available markers is determined by the Request FB itself (recommended). The evaluation is performed only during the initialization phase of the function block.

❑ **ERROR_STATUS**

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

↗ For a list of all possible error codes, please see chapter *Error codes of Request blocks*, page 96

ibaREQ_UDPact (FB145)

Description of the formal parameters:

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the ibaPDA 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**

Via this DB the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

❑ **Xchange**

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

❑ **ERROR_STATUS**

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

➔ For a list of all possible error codes, see chapter *Error codes of Request blocks*, page 96

ibaREQ_UDPint (FB146)

Description of the formal parameters:

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the ibaPDA 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**

Via this DB the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

❑ **CON_ID**

Unique reference to the connection to be set up. 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.

Value	Meaning
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 and CPU41x-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 CPU 41x-5H PN/DP (Rack 1).
B#16#18	Communication via the integrated PROFINET interface of the CPU 410-5H (Rack 1)

Table 3:   Valid values of the parameter LOCAL_DEVICE_ID

Other documentation

You find more information under the following *link https://support.industry.siemens.com/cs/de/de/view/51339682/en*:

❑ **LOCAL_PORT**

Number of the locally used port.

❑ **RESET_CON**

Serves for manually resetting the communication connection.

❑ **ERROR_STATUS**

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

↗ For a list of all possible error codes, please see chapter *Error codes of Request blocks*, page 96

❑ **ERROR_TCON**

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

0x8yyy   errorcode of TCON block will be 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 TUSEND block are output.

0x8yyy   errorcode of TUSEND block will be 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.

ibaREQ_UDPext3 (FB147)

Description of the formal parameters:

Name	Type	Data type	Description
REQ_DB	IN	Block_DB	DB of the iba 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**

Via this DB the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

❑ **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 code of the block. If there is no error, the value 0 is output.

↗ For a list of all possible error codes, please see chapter *Error codes of Request blocks*, page 96.

❑ **ERROR_SEND**

Error code when sending. The standard error codes of the AG_SEND modules are output.

0x8yyy   errorcode of AG_SEND 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.

ibaREQ_UDPext4 (FB148)

Description of the formal parameters:

Name	Type	Data type	Description
REQ_DB	IN	Block_DB	DB of the ibaPDA 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**

Via this DB, the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

❏ **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 code of the block. If there is no error, the value 0 is issued.

➚ For a list of all possible error codes, please see chapter *Error codes of Request blocks*, page 96.

❏ **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   errorcode 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.

3.2.1.2   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 (see chapter *iba S7 library*, page 69) to the blocks folder of your STEP 7 project.

Important note

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

The blocks ibaREQ_UDPint, ibaREQ_UDPext3 and ibaREQ_UDPext4 have to adapted when

- 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 notes on the adaptation, see chapter *Adaptation to the renumbered system functions*, page 80.

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

The following blocks are required:

- ibaREQ_M (FB140)
- ibaREQ_UDPact (FB145)
- ibaREQ_UDPint (FB146)
- ibaREQ_DB (DB15)
- ibaUDT_UDPact (UDT145)

For each Request module, the following steps have to be carried out:

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

```
                                    DB140
                                  "IDB_REQ-M"
                                    FB140
                          ibaPDA-Request-S7 management block
                                   "ibaREQ_M"

                    ... ── EN

        DB11
     Interface-DB
   ibaPDA-S7-Request
     "ibaREQ_DB512" ── REQ_DB

        M38.0
   "ibaREQ-M_RESET" ── RESET

                    0 ── INP_RANGE

                    0 ── OUT_RANGE                  ERROR_STATUS ── ...

                    0 ── MARKER_RANGE                        ENO ──
```
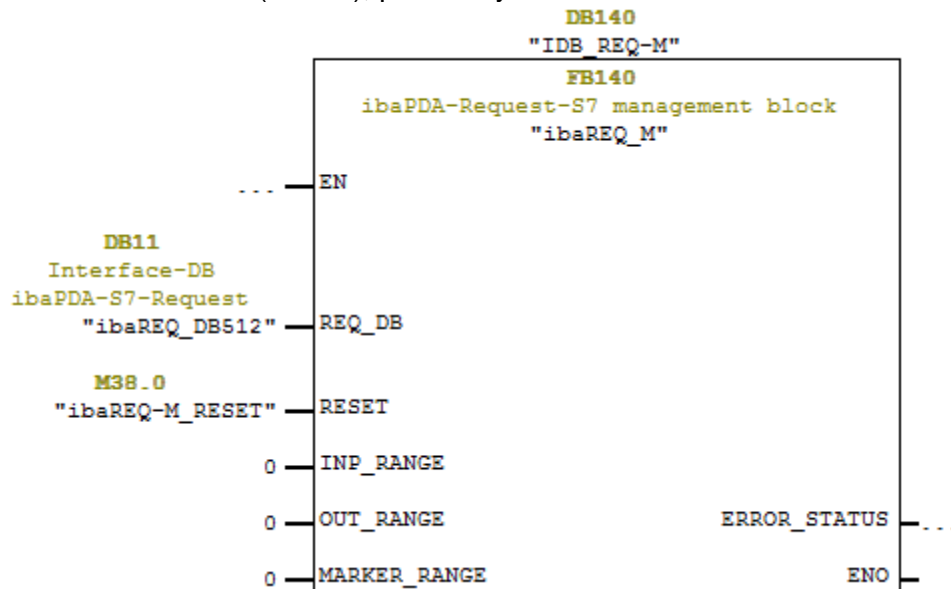
2. Call the ibaREQ_UDPint (FB146), preferably within a cyclic interrupt OB (OB3x)

```
                                    DB146
                              "IDB_REQ-UDPint"
                                    FB146
                          ibaPDA-Request-S7 block for sending
                            data via an internal PN-Interface
                                  "ibaREQ_UDPint"

                    ... ── EN

                                                 ERROR_STATUS ── ...
        DB15
  ibaPDA-Request-S7 –                            ERROR_TCON ── ...
     Interface-DB
        "ibaREQ_DB" ── REQ_DB                    COUNT_TCON ── ...

                    1 ── CON_ID                   ERROR_TSEND ── ...

             B#16#5 ── LOCAL_DEVICE_ID            COUNT_TSEND ── ...

             L#2001 ── LOCAL_PORT                 LOST_SAMPLES ── ...

                   ... ██ RESET_CON                        ENO ──
```
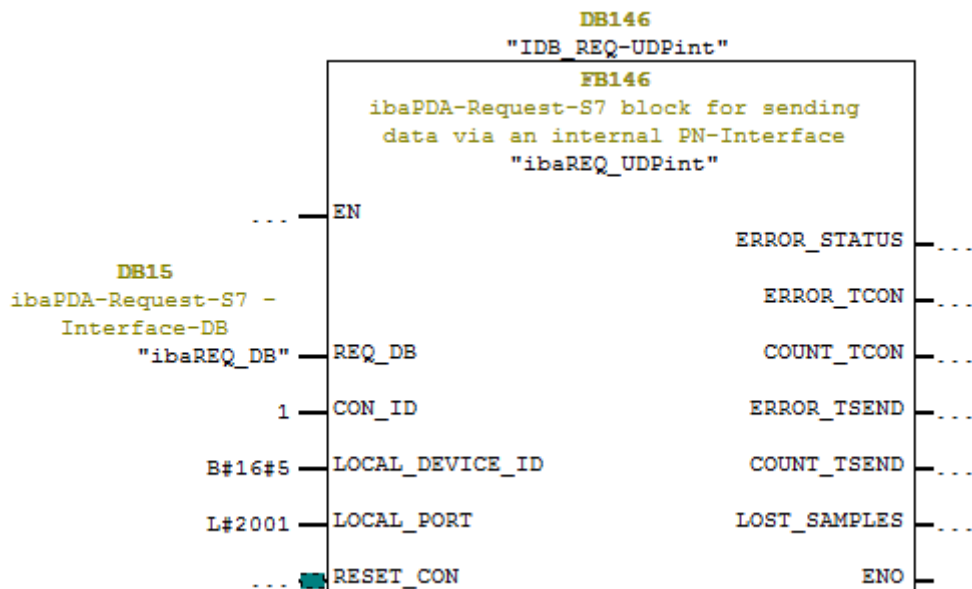
For each additional Request module:

❑ 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 for each Request module with a new DB number (input REQ_DB).

❑ Within a cyclic interrupt OB (OB3x), the ibaREQ_UDPint (FB146) has to be called for each Request module once more with the new DB number (input REQ_DB).

❑ Please consider that all instance data blocks have to be unique and that the values for the CON_ID and LOCAL_PORT parameters are assigned uniquely.

Finish:

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

**CPU S7-300 with CP343-1**

The following function blocks are required:

- ibaREQ_M
- ibaREQ_UDPact
- ibaREQ_UDPext3
- ibaREQ_DB
- ibaUDT_UDPact (UDT145)

For each Request module, the following steps have to be carried out:

1. Configure a new connection in NetPro.
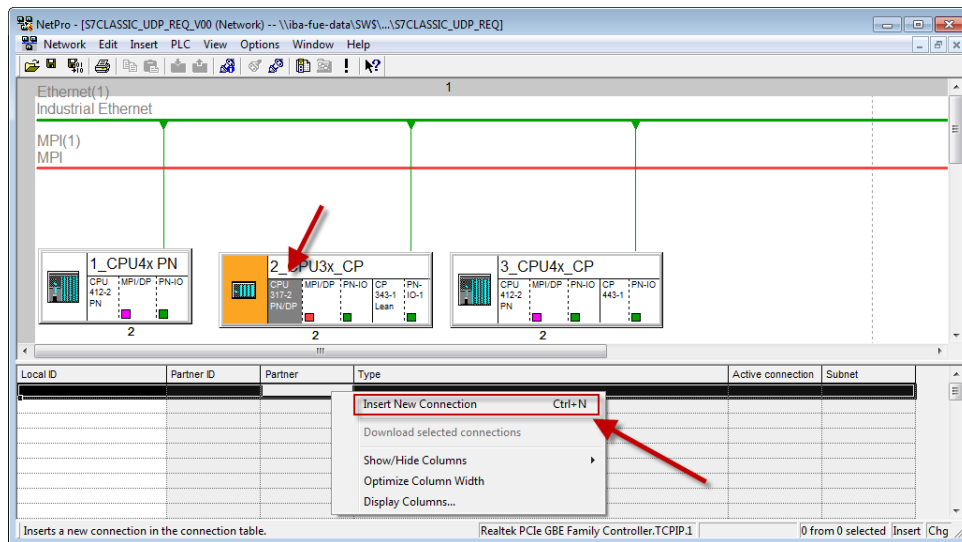
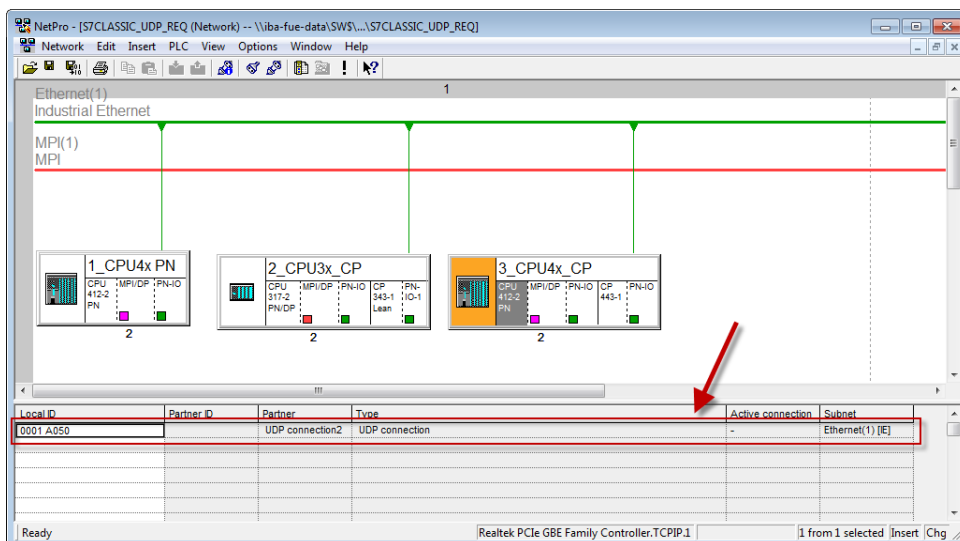**2.** Select the connection partner "unspecified" and connection type "UDP connection".
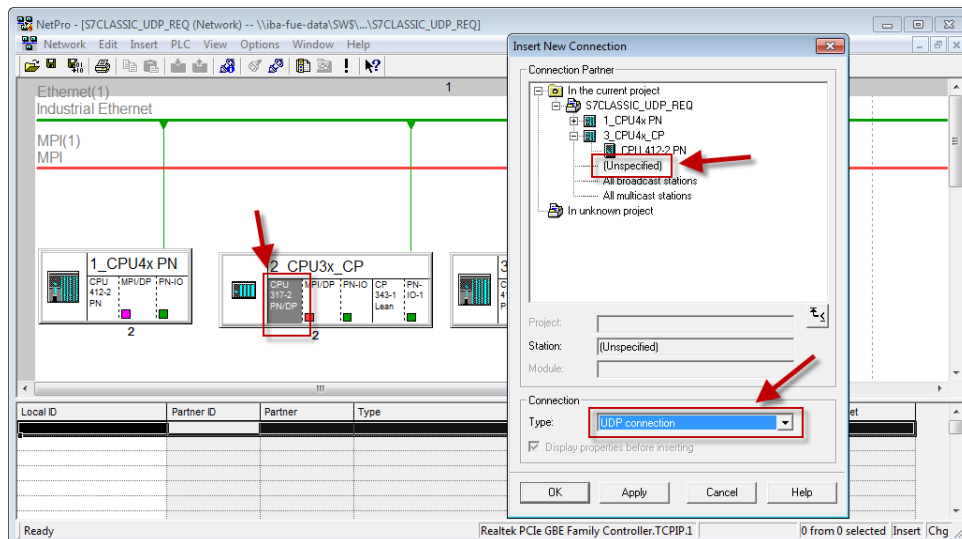
3. The automatically assigned function block parameter for the connection ID (ID) and the hardware starting address (LADDR) are needed later in step 7.
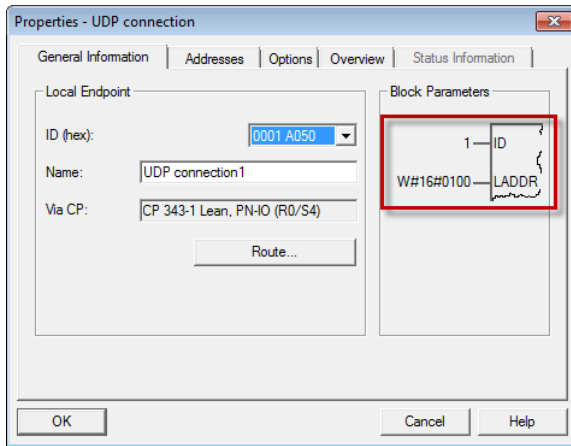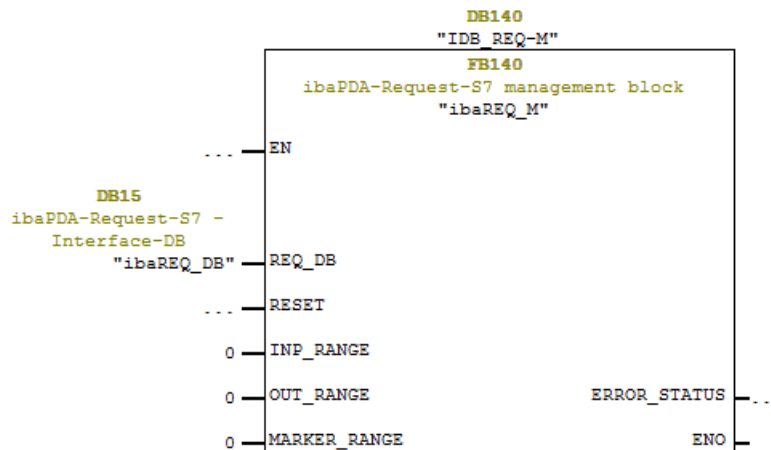
4. Enter the IP address of the *ibaPDA* computer as partner IP address and the configured port number (standard: 4170) and select a unique local port number.
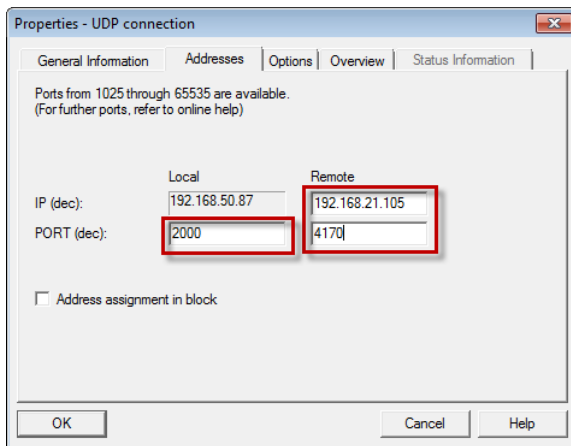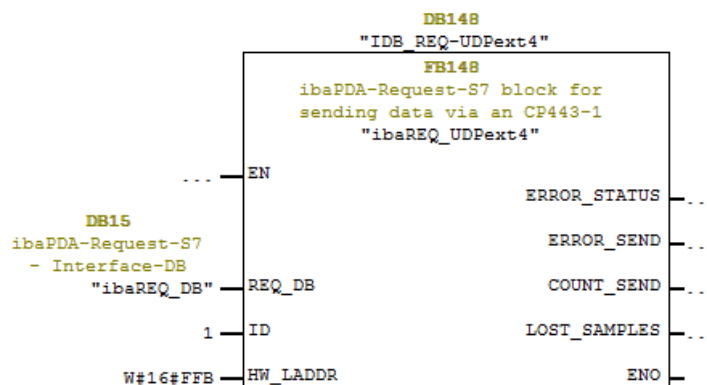

5. The connection table of the CPU now displays the newly configured connection.
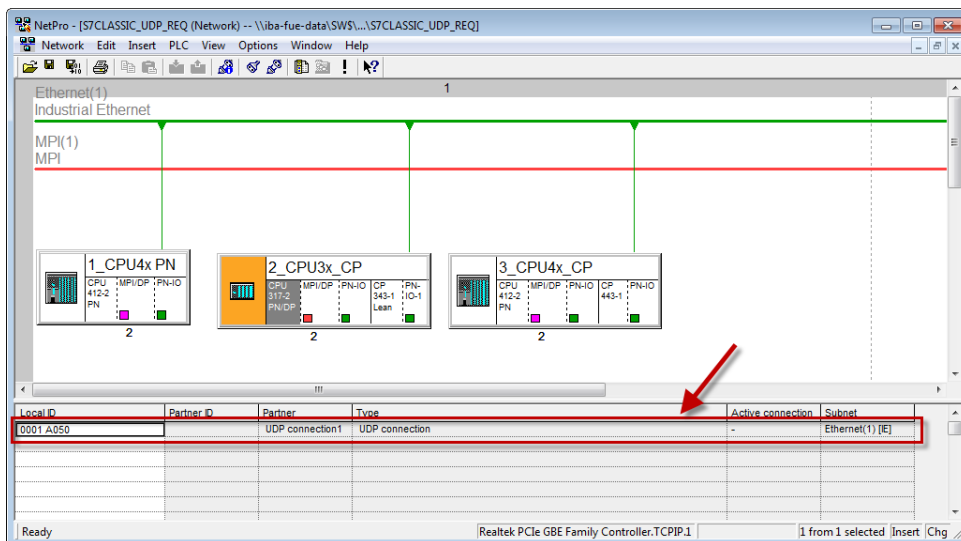
6. Call of ibaREQ_M (FB140), preferably within the OB1.

```
                                            DB140
                                          "IDB_REQ-M"
                                            FB140
                                  ibaPDA-Request-S7 management block
                                            "ibaREQ_M"
                    ... ──  EN

         DB15
   ibaPDA-Request-S7 -
      Interface-DB
       "ibaREQ_DB" ──  REQ_DB

                    ... ──  RESET

                      0 ──  INP_RANGE

                      0 ──  OUT_RANGE                  ERROR_STATUS ── ...

                      0 ──  MARKER_RANGE                        ENO
```
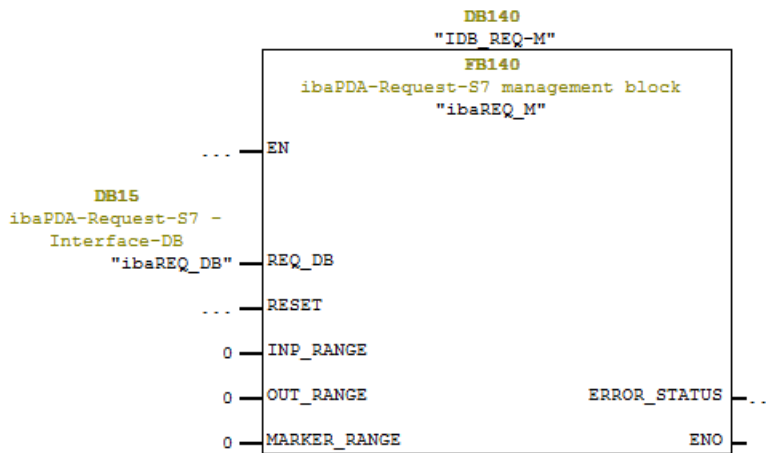
7. Call of ibaREQ_UDPext3 (FB147), preferably within a cyclic interrupt OB (OB3x)

```
                                            DB147
                                        "IDB_REQ-UDPext3"
                                            FB147
                                  ibaPDA-Request-S7 block for sending
                                       data via an CP343-1
                                          "ibaREQ_UDPext3"
                    ... ──  EN
                                                      ERROR_STATUS ── ...
         DB15
   ibaPDA-Request-S7 -                                 ERROR_SEND ── ...
      Interface-DB
       "ibaREQ_DB" ──  REQ_DB                          COUNT_SEND ── ...

                      1 ──  ID                        LOST_SAMPLES ── ...

              W#16#100 ──  HW_LADDR                           ENO
```
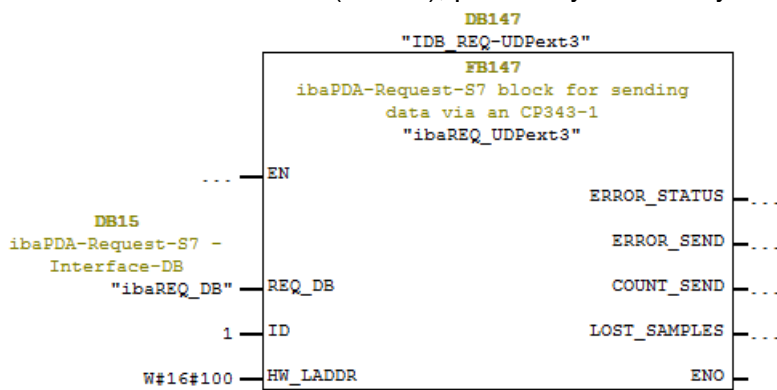
For each additional Request module:

❑ In the blocks folder, one 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 for each Request module a separate connection (steps 1-4). Assign different local port numbers.

❑ Within the OB1 the ibaREQ_M (FB140) has to be called once more for each Request module with a new DB number (input REQ_DB)   (step 6).

❑ Within a cyclic interrupt alarm call OB (OB3x), the ibaREQ_UDPext3 (FB147) has to be called once more for each Request module with the new DB number (input REQ_DB) (step 7).

❑ Please consider that all instance data blocks have to be unique and that the values for the ID and HW_LADDR parameters are to be assigned uniquely.

Finish:

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

CPU S7-400 with CP443-1

The following blocks are required:

- ibaREQ_M
- ibaREQ_UDPact
- ibaREQ_UDPext4
- ibaREQ_DB
- ibaUDT_UDPact (UDT145)

For each Request module, the following steps have to be carried out:

1. Configure a new connection in NetPro.
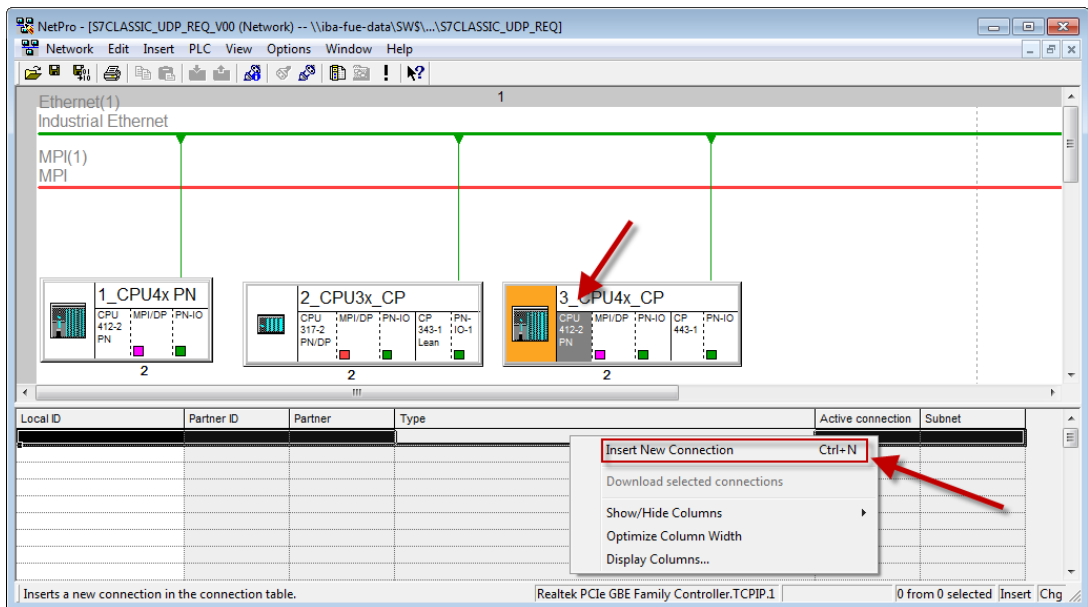
2. Select the connection partner "unspecified" and connection type "UDP connection".
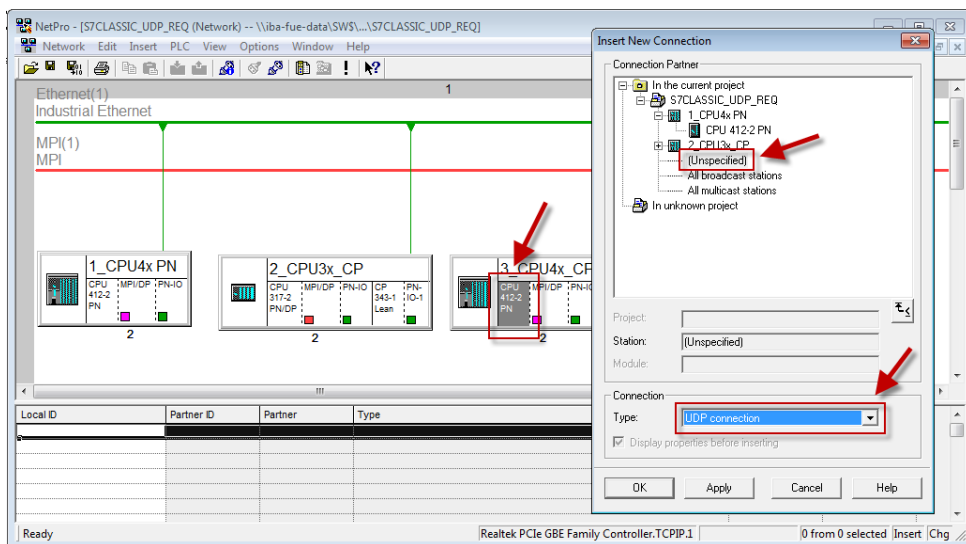
3. The automatically assigned function block parameters are needed later in step 7.
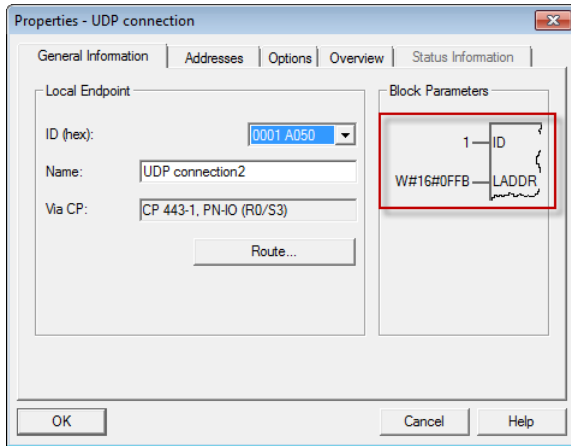
4. Enter the IP address of the *ibaPDA* computer as partner IP address and the configured port number (standard: 4170) and select a unique local port number.
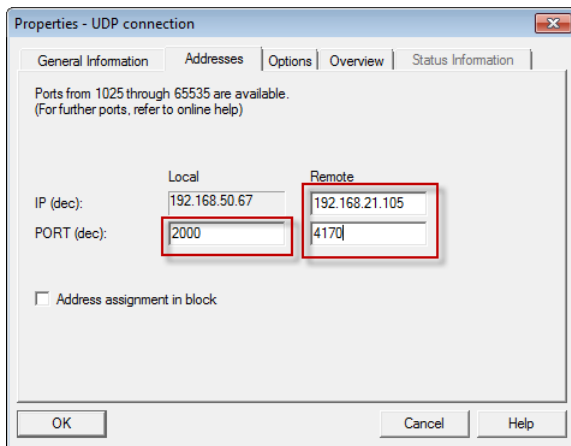
5. The connection table of the CPU now displays the newly configured connection.

6. Call the ibaREQ_M (FB140), preferably within the OB1.

```
                                    DB140
                                  "IDB_REQ-M"
                                    FB140
                          ibaPDA-Request-S7 management block
                                  "ibaREQ_M"
                  ...  ─ EN

        DB15
    ibaPDA-Request-S7 -
      Interface-DB
      "ibaREQ_DB" ─ REQ_DB

                  ...  ─ RESET

                    0 ─ INP_RANGE

                    0 ─ OUT_RANGE                    ERROR_STATUS ─ ...

                    0 ─ MARKER_RANGE                          ENO
```

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

```
                                    DB148
                               "IDB_REQ-UDPext4"
                                    FB148
                          ibaPDA-Request-S7 block for
                          sending data via an CP443-1
                               "ibaREQ_UDPext4"
                  ...  ─ EN                           ERROR_STATUS ─ ...

        DB15
    ibaPDA-Request-S7                                  ERROR_SEND ─ ...
      - Interface-DB
      "ibaREQ_DB" ─ REQ_DB                             COUNT_SEND ─ ...

                    1 ─ ID                            LOST_SAMPLES ─ ...

            W#16#FFB ─ HW_LADDR                               ENO
```

For each additional Request module:

❑ In the blocks folder, one 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 for each Request module a separate connection (steps 1-4). Assign different local port numbers.

❑ Within the OB1, the ibaREQ_M (FB140) has to be called once more for each Request module with the new DB number (input REQ_DB) (step 6).

❑ Within a cyclic interrupt OB (OB3x), the ibaREQ_UDPext4 (FB148) has to be called once more for each Request module with a new DB number (input REQ_DB) (step 7).

❑ Please consider that all instance data blocks have to be unique and that the values for the parameters ID and HW_LADDR are assigned uniquely.

Finish:

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

3.2.2 Configuration SIMATIC S7-1500

Basically, the following configuration steps have to be carried out on the SIMATIC TIA Portal side:

❑ Configuration Software   (STEP 7 ≥ V13 SP1):
Integration of the Request blocks in the S7 program

❑ Device configuration:
Setting the CPU protection properties

3.2.2.1 Description of the Request blocks

With these blocks, the communication between the S7 and *ibaPDA* is initialized and controlled. A 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 69).

Request block	S7-1500 CPU with integrated PN interface	S7-1500 CPU with CP1543-1 or CM1542-1	recommended call up level
ibaREQ_M (FB1400)	X	X	OB1
ibaREQ_UDP2 (FB1406)	X	X	OB3x
ibaREQ_UDPact (1410)	X	X	OB3x
ibaREQ_UDP (FB1405)	X	X	OB3x
ibaREQ_DB (DB15)	X	X	-
ibaREQ_DB-Interface	X	X	-

Table 4:   Combination of Request blocks

❑ ibaREQ_M (Management)
The block realizes the communication with *ibaPDA*. Ideally, the block is called in the OB1.

❑ ibaREQ_UDP2 (provides and sends the current signal values)
The block provides the actual signal values in the send cycle. Ideally, the block is called in a cyclic interrupt OB.

❑ ibaREQ_UDPact
The block is used internally by ibaREQ_UDP2.

❑ ibaREQ_UDP
Similar function like ibaREQ_UDP2. This block is available for backward compatibility reasons. We recommend using the block ibaREQ_UDP2.

❑ ibaRQ_DB (interface DB)
This DB serves as interface to *ibaPDA* and between the different Request blocks.

ibaREQ_M (FB1400)

Description of the formal parameters:

Name	Type	Data type	Description
REQ_DB	IN	DB_ANY	DB of the ibaPDA communication interface ibaREQ_DB
RESET	IN	BOOL	TRUE -> perform reset
CPU_HW_ID	IN	HW_IO	Hardware ID of local CPU
ERROR_STATUS	OUT	WORD	Error code

The following SIMATIC standard blocks are used internally:

❑ GET_IM_DATA (FB801)

Detailed description:

❑ **REQ_DB**

Via this DB, the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

The length of the data block is fix.

❑ **RESET**

Serves for manually resetting 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.
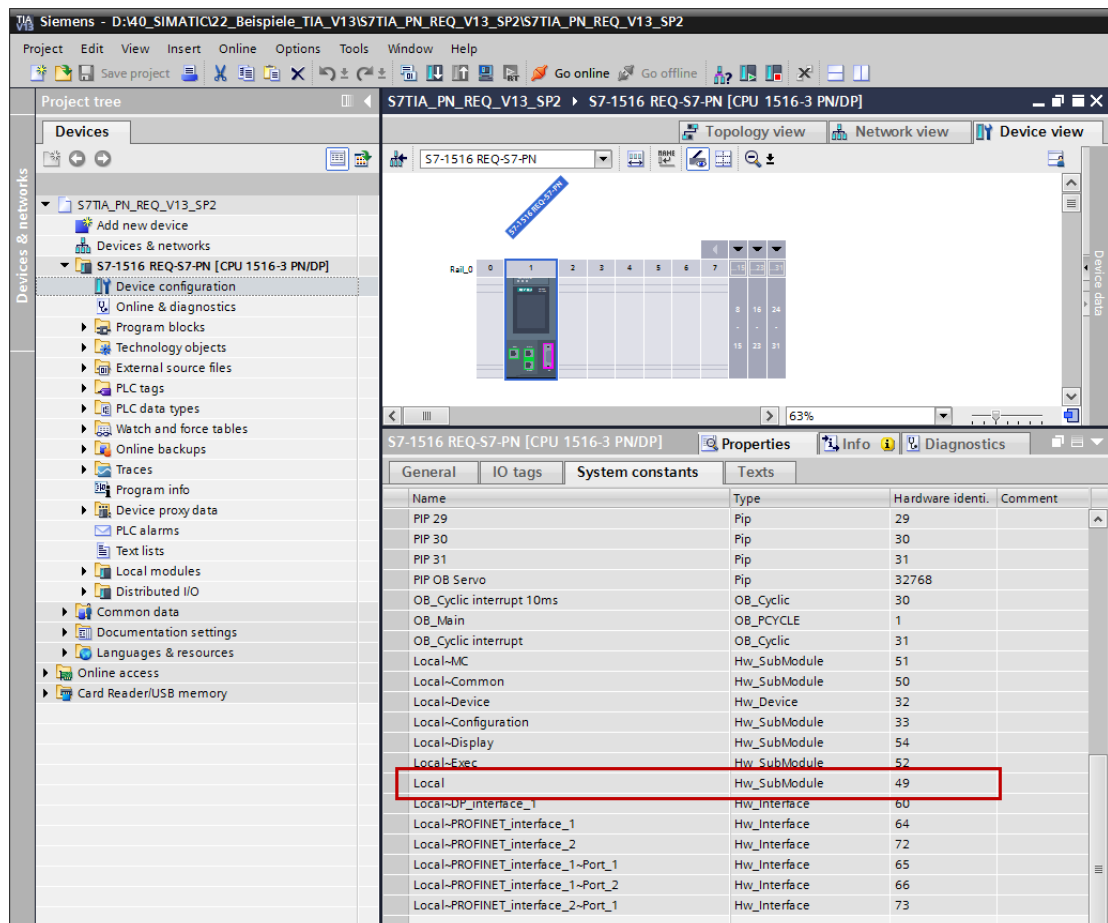
❑ **ERROR_STATUS**

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

↗ For a list of all possible error codes, see chapter *Error codes of Request blocks*, page 96.

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	TRUE -> reset of the communication connection
REQ_DB	INOUT	DB_ANY	DB of the ibaPDA 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 the system constant of the interface and not of a port, or of the IO system is to be used.

❑ **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**
Serves for manually resetting the communication connection.

❑ **REQ_DB**
Via this DB, the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

❑ **ERROR_STATUS**
Internal error code of the block. If there is no error, the value 0 will be issued.

↗ For a list of all possible error codes, see chapter *Error codes of Request blocks*, page 96

❑ **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.

ibaREQ_UDP (FB1405)

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	TRUE -> reset of the communication connection
REQ_DB	INOUT	DB_ANY	DB of the ibaPDA communication interface ibaREQ_DB
ERROR_STATUS	OUT	WORD	Internal error code
ERROR_TSEND	OUT	WORD	Error code send block
Error code send block	OUT	WORD	Telegram counter of the send block
LOST_SAMPLES	OUT	UNIT	Counter for lost measurement values

The following SIMATIC standard blocks are used internally:

❑ TSEND_C (FB1030)

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 the system constant of the interface and not of a port, or of the IO system is to be used.

❑ **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**

Serves for manually resetting the communication connection.

❑ **REQ_DB**

Via this DB, the data are exchanged with *ibaPDA*. For all Request blocks belonging together, the identical DB has to be configured.

❑ **ERROR_STATUS**

Internal error code of the block. If there is no error, the value 0 will be issued.

↗ For a list of all possible error codes, see chapter *Error codes of Request blocks*, page 96

❑ **ERROR_TSEND**

Error code when sending

Standard error codes of the TSEND_C block are issued.

0x8yyy   errorcode of TSEND_C block will be passed

❑ **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.

3.2.2.2 Configuration in STEP 7

In the following, we describe the configuration of the Request blocks in STEP 7 ≥ V13 SP1.

Copy the required blocks from the iba S7 library (see chapter *iba S7 library*, page 69) to the blocks folder of your STEP 7 project.

The following blocks are required:

- ibaREQ_M (FB1400)
- ibaREQ_UDP2 (FB1406)
- ibaREQ_UDPact (FB1410)
- ibaREQ_DB (DB15)
- ibaREQ_DB-Interface (PLC data type)

For each Request module, the following steps have to be carried out:

1. Call the ibaREQ_M (FB1400) preferably within the OB1
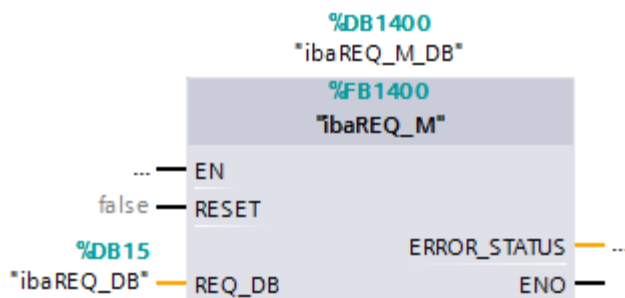
2. Enable "Retain" for the complete instance block, you have just created.
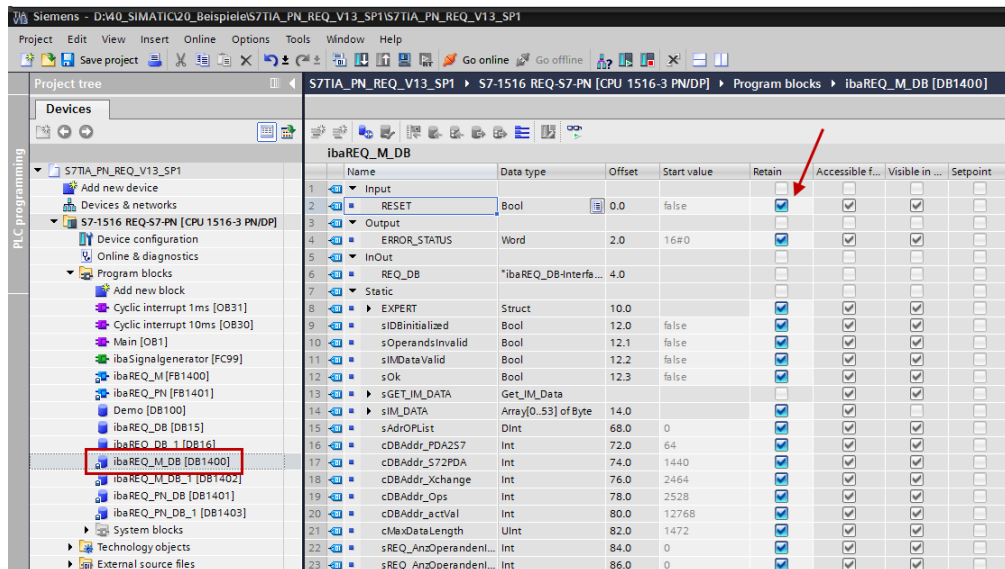
3. Call the ibaREQ_UDP2 (FB1406), preferably within a cyclic interrupt OB (OB3x)
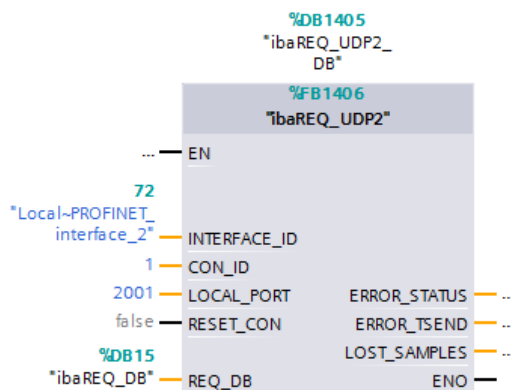
For each additional Request module:

❑ In the blocks folder, an 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 for each Request module with the new DB number once more.

❑ Within a cyclic interrupt (OB3x), the ibaREQ_UDP2 (FB1406) has to be called for each Request module with the new DB number once more.

❑ Please consider that all instance data blocks have to be unique and that the values for the CON_ID and LOCAL_PORT parameters are assigned uniquely.

Finish:

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

3.2.2.3 Device configuration

The following settings have to be done in the device configuration of the CPU:
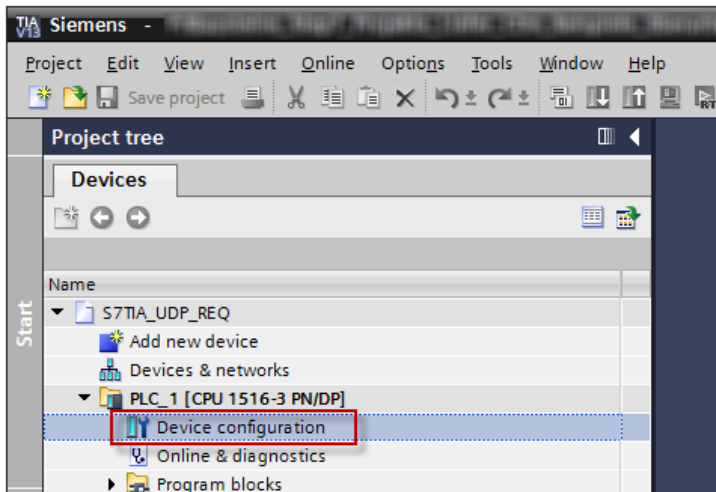
Figure 2:   Device configuration

1. Disable ibaREQ_DB (DB15) Properties – Attributes – Optimized block access
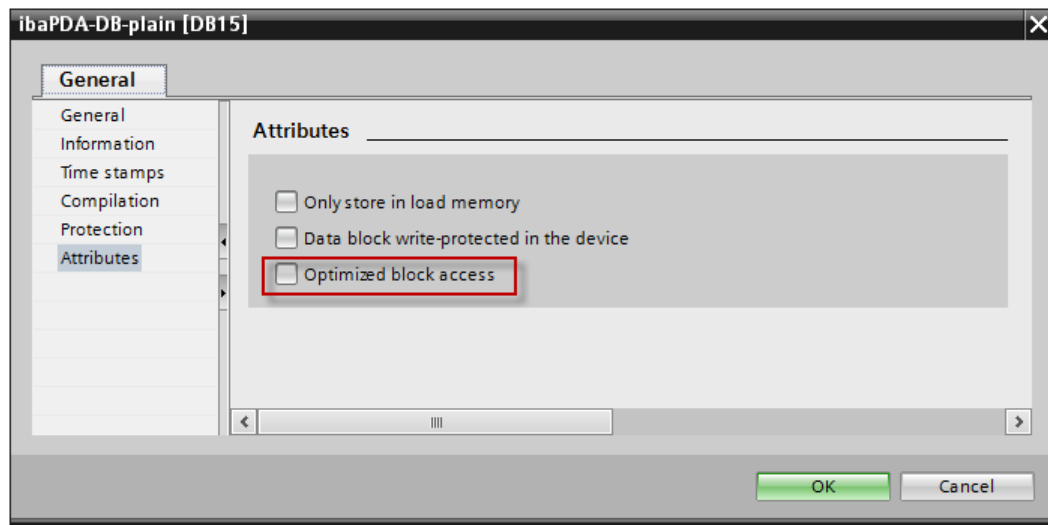
When using the connection mode TCP/IP (not TCP/IP S7-1x00) with S7-1500 CPUs, the following settings at the device configuration have to be made in addition:

2. Properties – General – Protection – Connection mechanisms: Permit access with PUT/GET communication
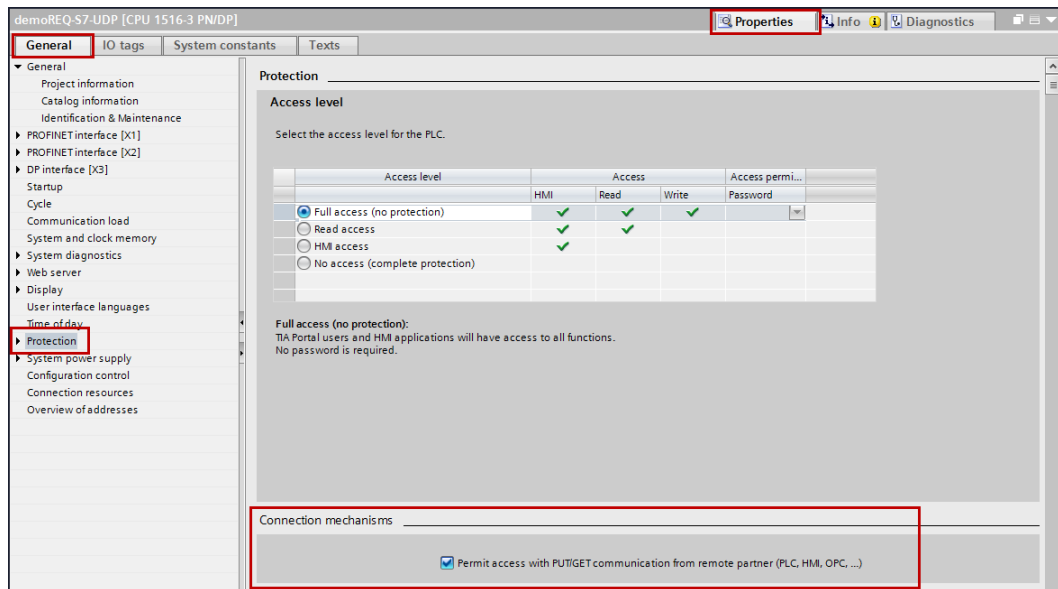
Access protection S7-1500

An access protection can be enabled for an S7-1500 CPU. The following dependancies on *ibaPDA* apply:

Access level	CPU access	ibaPDA reads symbolic from CPU	S7-Request 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

Table 5:   Access protection for S7-1500

Important 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.

3.3   Configuration and engineering ibaPDA

3.3.1   General interface settings

The interface *ibaPDA-Interface-S7-TCP/UDP* is configured in the *ibaPDA* "I/O Manager". If all system requirements are met, the "S7 TCP/UDP" data interface is displayed in the interface tree. *ibaPDA-Request-S7-UDP* is a module ("S7 Request") of this interface.
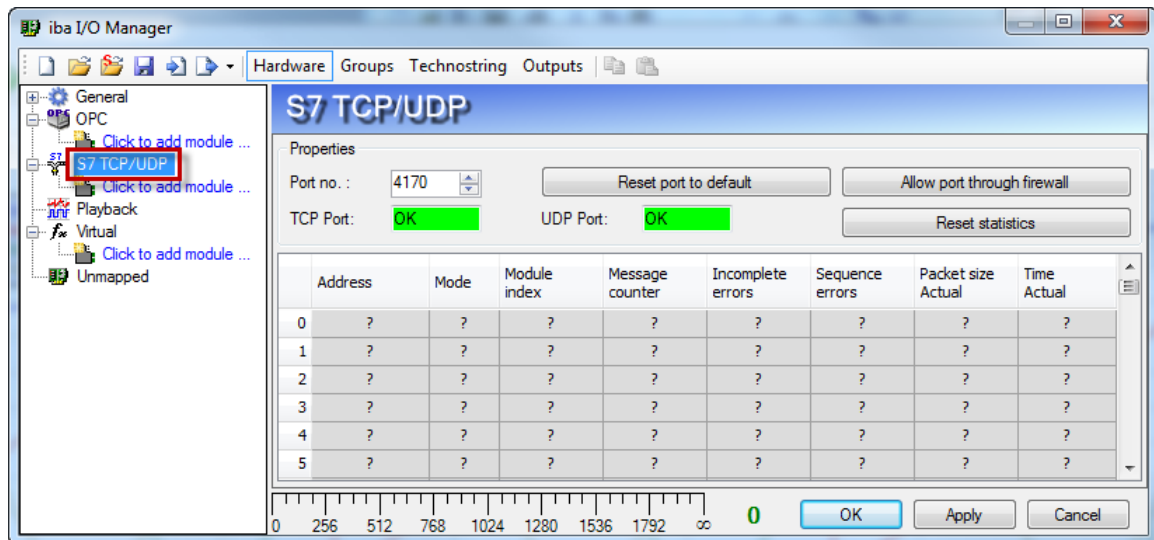
Figure 3:   Interface in the I/O Manager

The interface has the following functions and configuration options:

❑ Port-no.
Port used on side of the PC

The port no. has to be used identically in the S7 connection configuration.

By means of <Reset port to default>, the setting can be reset under "Port No." to the default setting.

Via <Allow port through firewall>, the Firewall exceptions can be reactivated. This might be necessary e.g. after a modification of the Port no. The settings are defined as default already in course of the installation.

With <Reset statistics>, you can reset the counter.

In the table, all active connections are entered.

With a double-click on a row, you jump to the configuration of the corresponding module.

Other documentation

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

↗ For information about the connection diagnostics, see the chapter *Diagnostics*, page 65.

3.3.2 Add module

Add a module by clicking below the interface. Select the desired module type and click on <OK>.
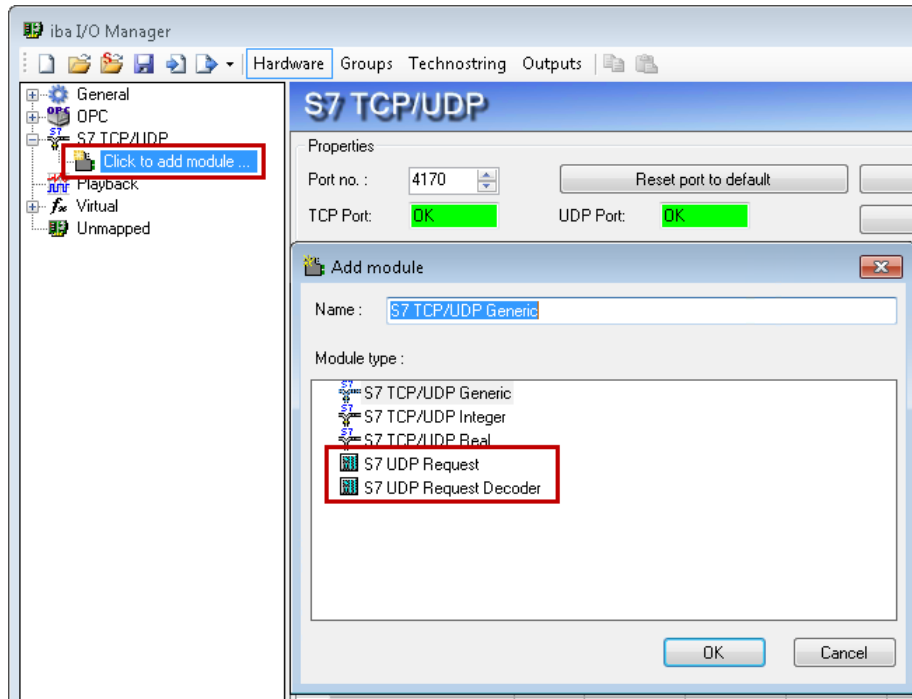
Figure 4:   Add module

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

Table 6:   Module overview of the Request-S7-UDP interface

3.3.3   General module settings

All modules have the following common setting options:

Figure 5:   General module settings

Basic settings

❑ Module type (read only)
Display of the module type

❑ Locked
A module can be locked for preventing accidental or unauthorized changes of the module settings.

❑ Enabled
Disabled modules are excluded from signal acquisition.

❑ Name
Here, the clear text name of the module designation has to be entered.

❑ Module No.
Consecutive number that is assigned by *ibaPDA* in ascending order and can be changed by the user. This number serves to identify signals uniquely, e.g. in expressions and *ibaAnalyzer*.

❑ Timebase

All signals of this module are acquired with this timebase.

❑ Use name as prefix

Puts the module name in front of the signal name.

Module Layout

❑ No of analog/digital signals

Defines the number of configurable analog/digital signals in the signal tables. A maximum of 1024 signals for each are allowed.

S7

❑ CPU Name

Select the S7-CPU that is connected to this module. When selecting a S7-CPU, you can select the signals symbolically. Otherwise, the signals are selected via the S7 operand.

This requires that address books have already been generated. Otherwise, the selection list is empty. When clicking in the selection list on „Create address book…", you get directly to the address book generator (see chapter Selection via the operand symbols).

Connection

❑ Module index (read only)

Internal reference number of the module

❑ Auto enable/disable

If TRUE, the acquisition is started even when no connection to the S7-CPU can be established. The module is disabled. During the acquisition, *ibaPDA* tries to reconnect to the S7-CPU. If the connection can be established successfully, the acquisition is restarted.

If FALSE, the acquisition is not started, in case no connection could be established to the configured CPU.

3.3.4   Connection settings

On the *Connection* tab, the connection to the controller is configured.

The following controllers, connection modes and selection methods are supported:

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

Table 7:   Possible connections

Depending on the connection mode different settings have to be made.

3.3.4.1 Connection mode TCP/IP

This mode activates a connection via the standard network interface of the PC.
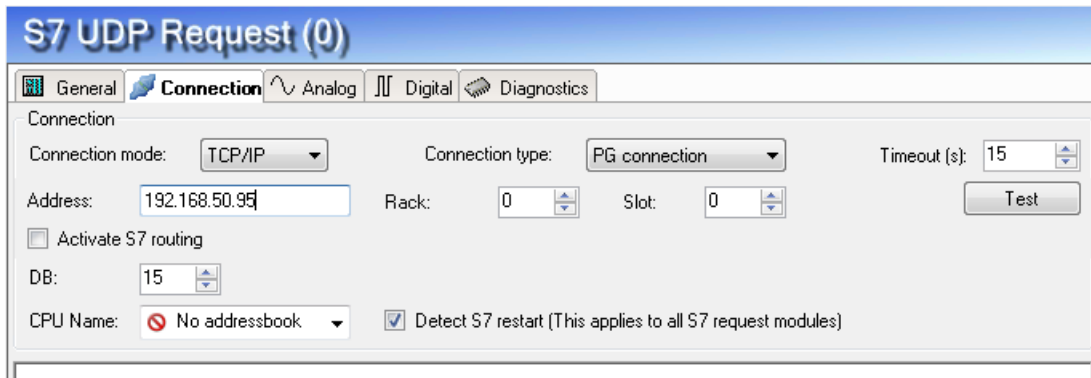
Figure 6:   Connection configuration TCP/IP

❑ CPU Name
Selection of the linked address book

❑ Connection mode
Selection of the TCP/IP connection mode

❑ Timeout
Timeout of the connection

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

❑ Address
IP address of the controller

❑ Rack
Rack number of the controller (default: 0)

❑ Slot
Number of the slot of the controller in the rack
(for S7-1500 CPUs: 0)

❑ Test

Connection test to the CPU and available diagnostic data are issued
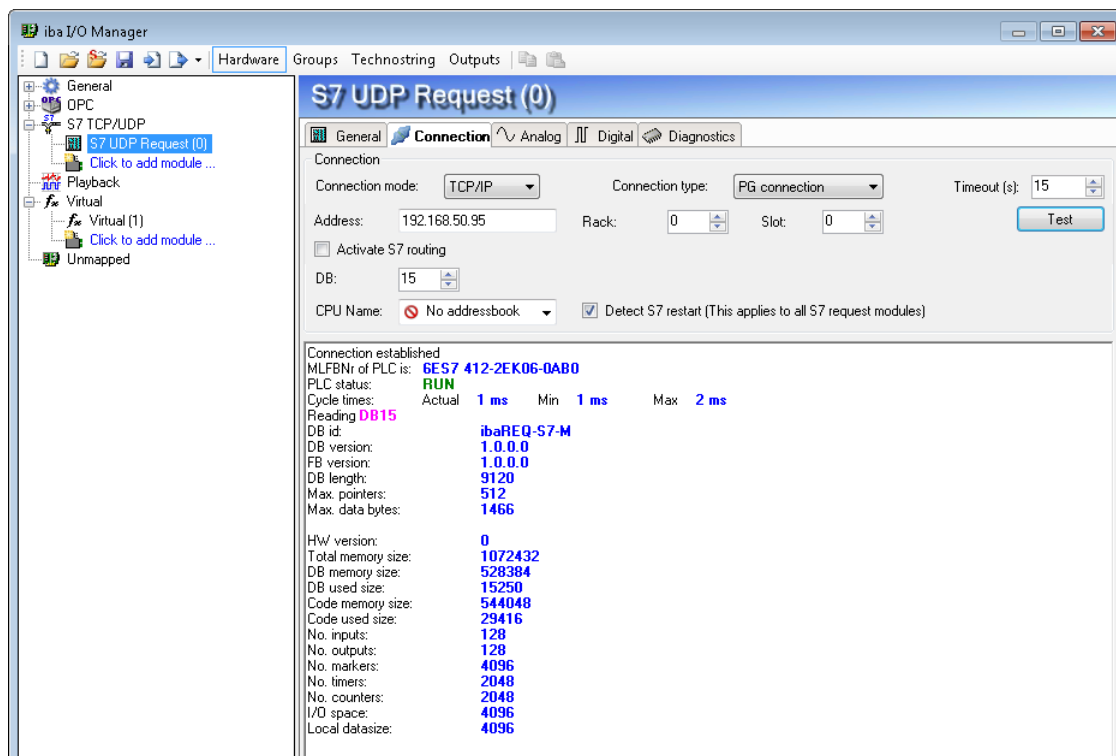
Figure 7:   Connection test/ diagnostic data

Tip

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

Please check if...

- 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 position

❑ Activate S7 routing (see also chapter *S7 routing*, page 86):

Activate this option, if the S7-CPU and the *ibaPDA*-PC 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 the device acting as gateway: Enter the IP address of the Gateway
- S7 Subnet ID of target net: Enter subnet-ID from STEP 7 NetPro

❑ 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 will restart the data acquisition. The configuration data are transferred again. This does not affect a warm restart of the CPU.

> **Note**
>
> **Port 102** in the firewall must be released for the communication with the CPU for signal request.

3.3.4.2   Connection mode PC/CP

This mode activates a connection over the interface card of the PC, which is configured using SIMATIC Net.

The interfaces configured in SIMATIC Net can be used, e. g.:

- MPI adapter (COM)
- MPI adapter (USB)
- PROFIBUS (CP5611)
- TCPIP (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 will be sufficient.
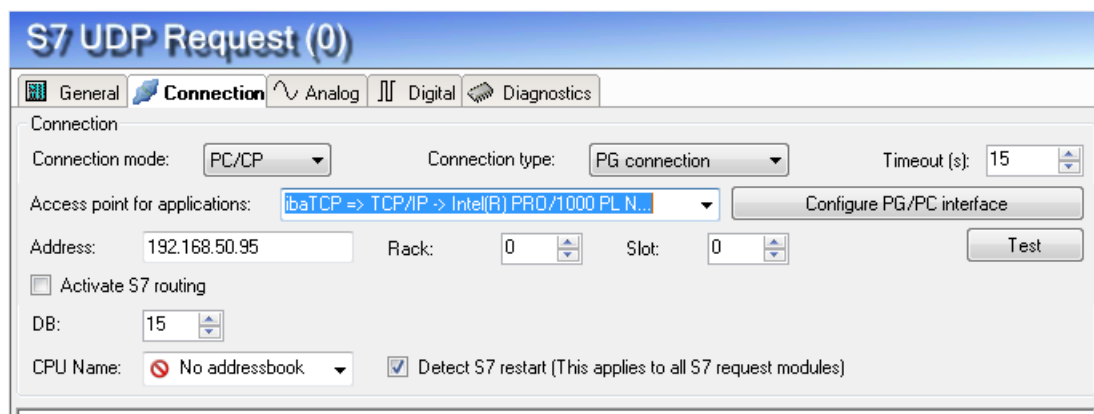
Figure 8:   Connection configuration PC/CP

❑ CPU Name
Selection of the linked address book

❑ 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
Timeout of the connection

❑ Access point for applications
Selection of the access point that is to be used

↗ For notes on creating and adapting an access point, see *Setting PG/PC interface / defining new access point*, page 82
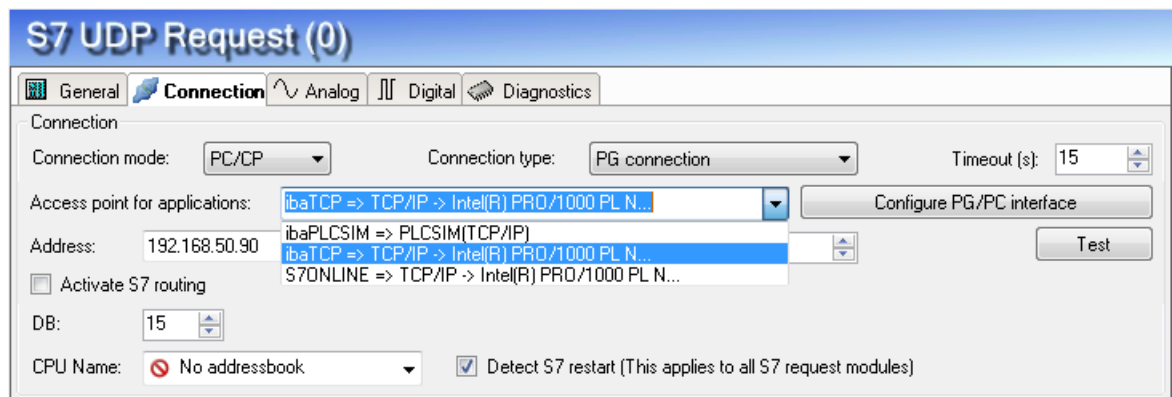
Figure 9:   Select access point

Note

The access points that are available under SIMATIC Net are configured using the Siemens tool
"PG/PC interface settings".

In case *ibaPDA-Request-S7-UDP* and SIMATIC Manager run on the same computer, iba recommends configuring a special access point for *ibaPDA* for the connection between *ibaPDA-Request-S7-UDP* and SIMATIC S7 via PC/CP connections. A separate access point for *ibaPDA-Request-S7-UDP* prevents the risk that access for *ibaPDA* is impaired when changing the standard access point in the SIMATIC Manager.

❑ Configure PG/PC interface
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
Number of the slot of the controller in the rack
(for S7-1500 CPUs: 0)

❑ Test

A connection test to the CPU is performed and available diagnostic data are issued.

Figure 10:   Connection test/ diagnostic data

Tip

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

Please check if...

- 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 position

❑ Activate S7 routing (see also chapter *S7 routing*, page 86):

Activate this option, if the S7-CPU and the *ibaPDA*-PC 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 the device acting as gateway: Enter the IP address of the Gateway

- S7 Subnet ID of target net: Enter subnet-ID from STEP 7 NetPro

❑ 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 will restart the data acquisition. The configuration data are transferred again. This does not affect a warm restart of the CPU.

3.3.4.3   Connection mode TCP/IP S7-1x00

This mode activates a connection over the standard network interface of the PC and can only be used in combination with S7-1500 CPUs.
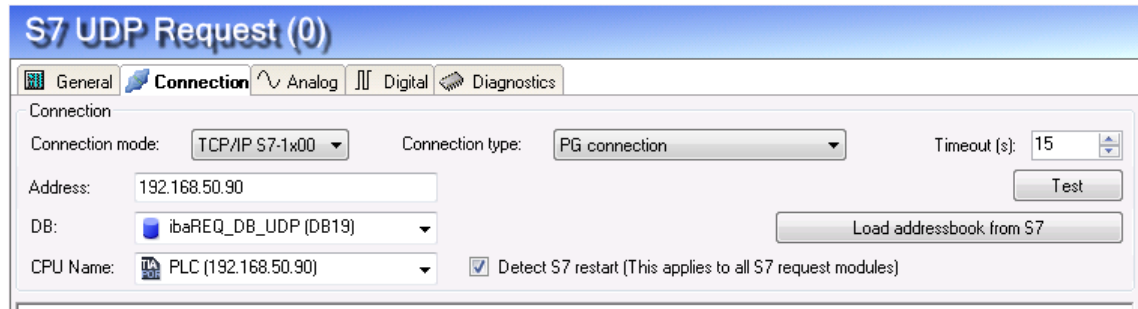
Figure 11:   Connection configuration TCP/IP S7-1x00

❑   CPU Name
Selection of the linked address book; only address books of the "TIA Portal" type can be selected.

❑   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
Timeout of the connection

❑   Address
IP address of the controller

❑ Test

A connection test to the CPU is performed and available diagnostic data are issued.
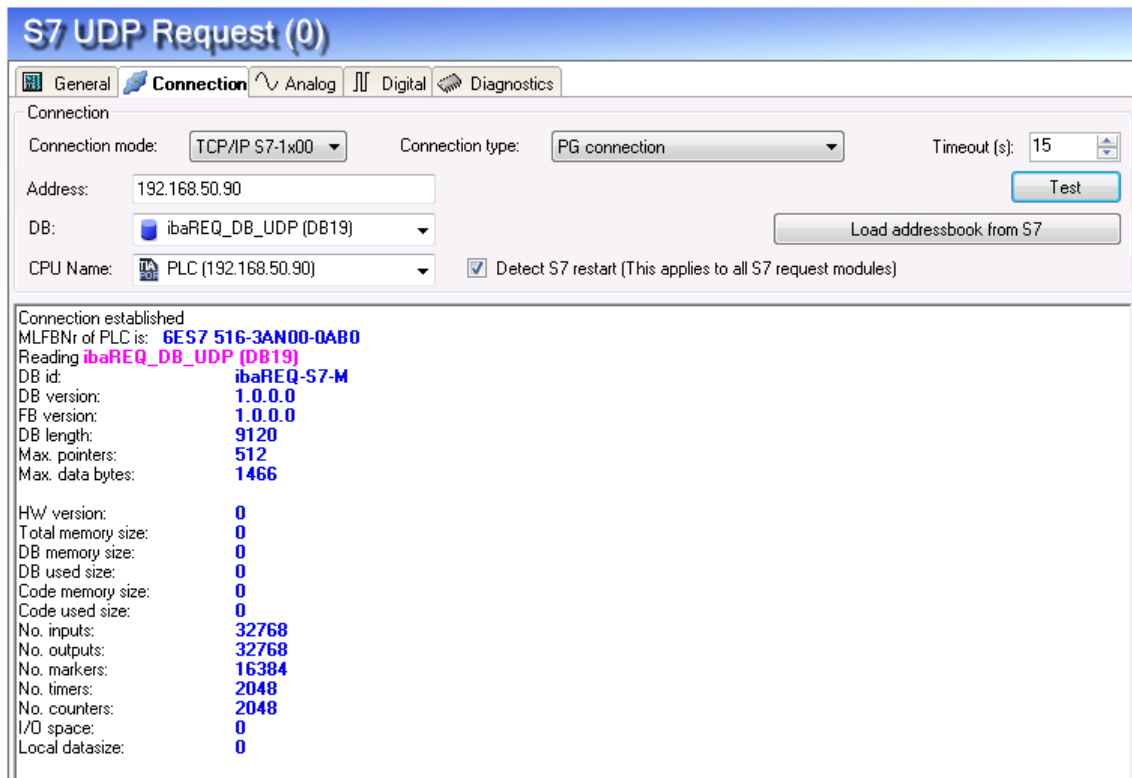
Figure 12:   Connection test/ diagnostic data

❑ Load addressbook from S7

By clicking on this button *ibaPDA* will read the list of symbols directly from the PLC and store it in an addressbook for further use in the symbol browser.
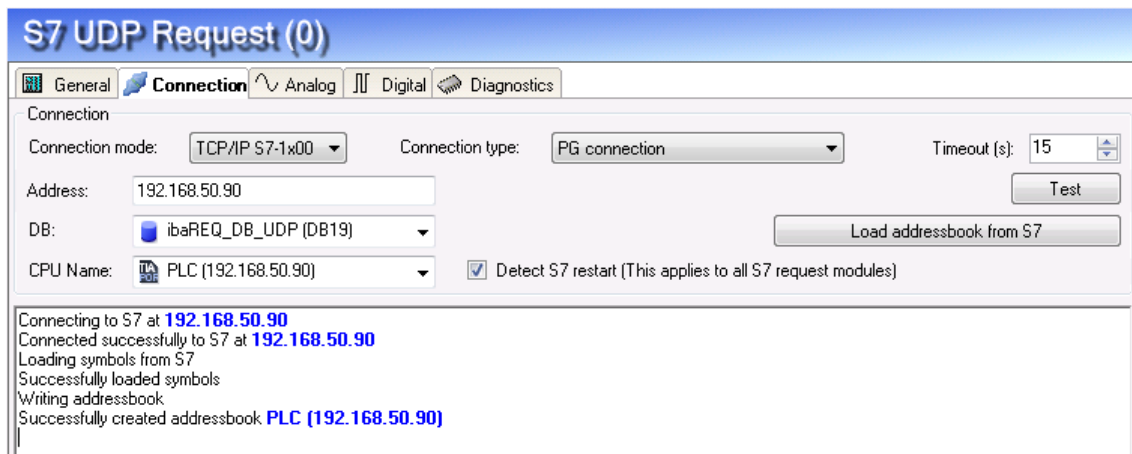
In case of an S7-1500, please consider the possibility of an enabled access protection.

↗ For more information see *Device configuration*, page 34

Tip

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

Please check if...

- 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 position

❑ 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 will restart the data acquisition. The configuration data are transferred again. This does not affect a warm restart of the CPU.

Note

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

Note

Port 102 in the firewall must be released for the communication with the CPU for signal request.

3.3.5   Signal configuration

The signals to be measured are selected in the I/O Manager, either by means of the absolute operand address or the symbolic name provided by the *ibaPDA* address book browser.

When using the SIMATIC CFC editor on the same PC, the signals and connectors to be measured can be configured from the PLC program via drag & drop in *ibaPDA*.

There are 3 options for selecting measurement values:

1   Selection via the absolute address of the S7 operands.

2   Selection via the S7 symbol addresses (symbol table and symbols from data blocks)

3   Selection of the CFC connectors (when programming the CPU with SIMATIC CFC)

SIMATIC CPU	Access via absolute address	Access via symbol	Access via CFC connectors[3]
S7-300	X	X	X
S7-400	X	X	X
WinAC	X	X	X
S7-1500	X	X	

Table 8:   Possible types of selections

Supported operand ranges:

Operand ranges	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[4]*

Table 9:   Supported operand ranges

---

[3] Required is the use of the SIMATIC STEP 7 options package S7-CFC
[4] The access to optimized data blocks of S7-1500 controllers is not supported.

3.3.5.1 Selection via the absolute address of the operands

There are 2 ways for selecting the measurement values via the operand address.

1 On the one hand, you can click on the *General* tab of the module on the "Select S7 operands" link in order to open the S7 operand editor.
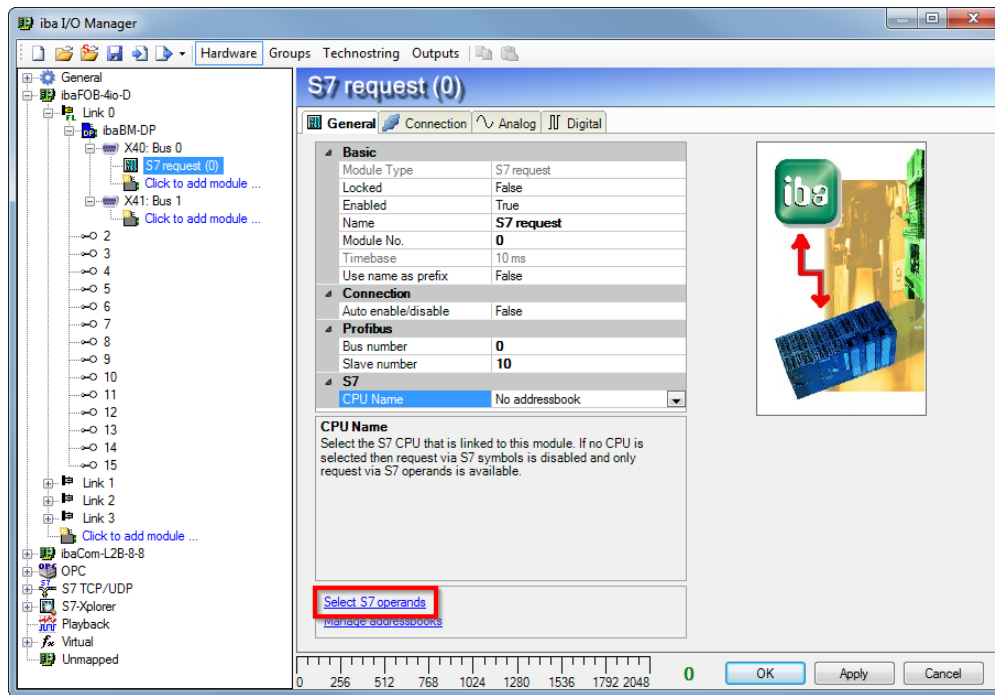
Figure 13: Selecting S7 operands

1 On the other hand, you can click on the *Analog* or *Digital* tab of the module in one field of the "S7 Operand" column.
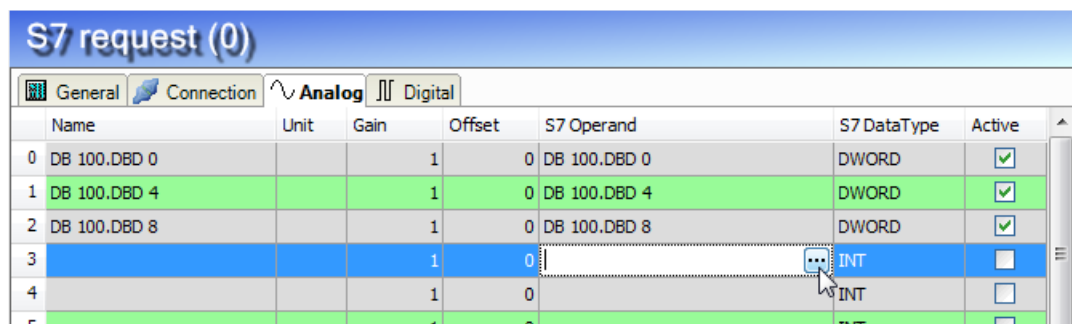
Figure 14: Selecting S7 operand

In both cases, a dialog box is opened, in which you can select the S7 operand to be measured.
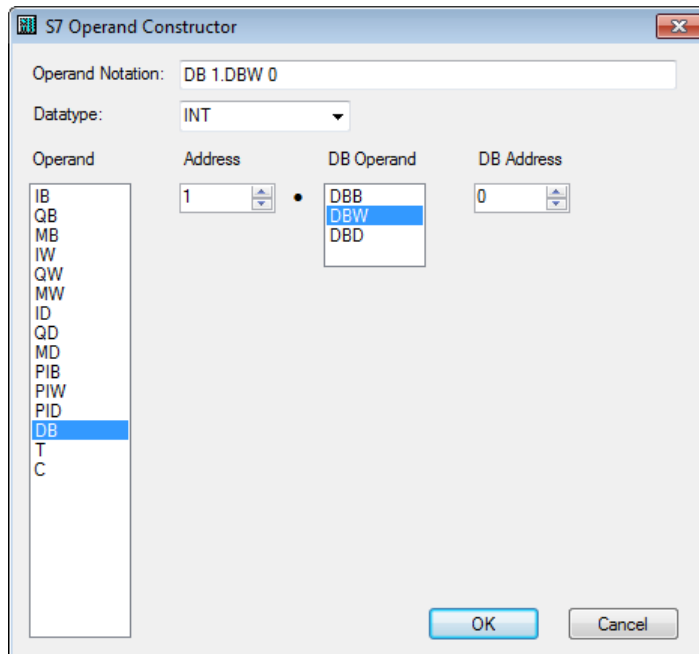
Figure 15:   S7 operand constructor

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.

Figure 16:   Operand names

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 operand symbols

This method of access requires that the signals to be measured already have an entry in the S7 symbol table, the PLC variable list or in a data block and that an address book has been created (see chapter *Address books*, page 60).

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

You can integrate an address book into the module as follows:

1. Select the S7-CPU you want to assign this module to in the drop down menu on the *General* tab of the module under "CPU Name".
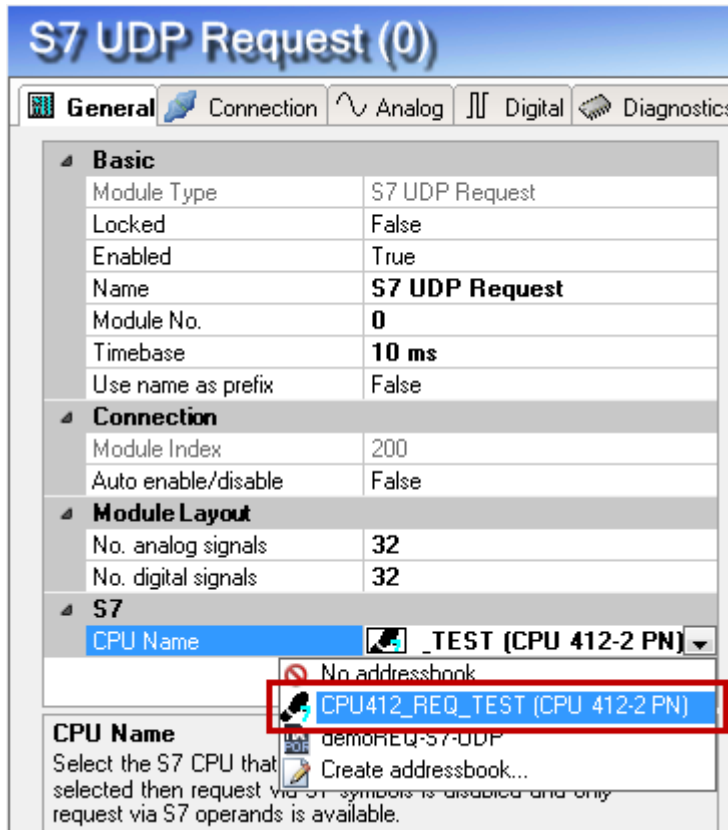
2. An additional "S7 symbol" column is displayed on the *Analog* and *Digital* tab.
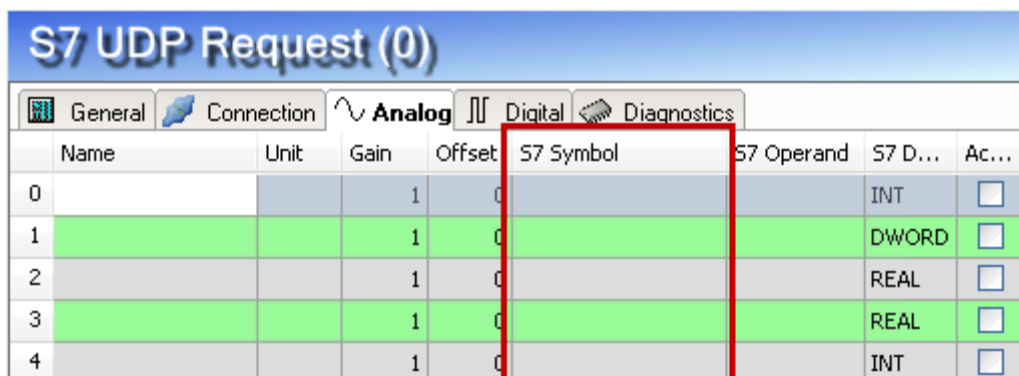
Now, you can access the symbol addresses via the *S7 CFC and symbol browser* (short: symbol browser). The symbol browser can be opened in different ways:

2   Click on the link „Select S7 symbols" on the *General* tab of the Request module. The selected signals are entered automatically in the appropriate table "Analog" or "Digital". The symbol browser stays open until it is closed with <OK>. This way, several signals can be added successively.

3   On the *Analog* or *Digital* tabs click in a cell of the "S7 Symbol" column. You can only select the symbols with the data type matching the table. After every selection, the symbol browser is closed.

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, the individual data blocks and their variables are displayed.

- Symbol table:
  On the *Symbols* tab, the entries of the S7 symbol table can be selected.

- *Search* tab:
  You can search the variable via a part of the name.
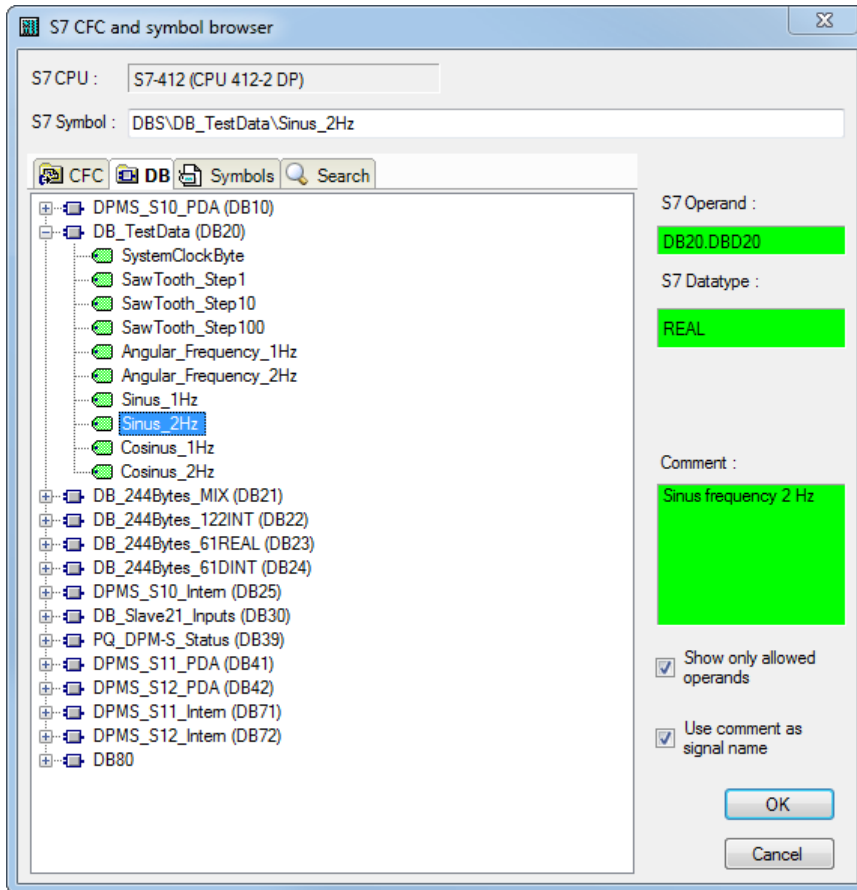
Figure 17:   S7-CFC and symbol browser

After having selected the variable, the operand address, the data type and a comment are displayed on the right side.

The variables are shown in the following colors:

Green	The operand is valid and can be transferred to the module setting with <OK>.
Yellow	The operand has a data type that does not match the selected column, 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 *ibaPDA* (e.g. STRING), or the operand is a constant.

You can hide all non-valid variables, by clicking on the option "Show only allowed operands".
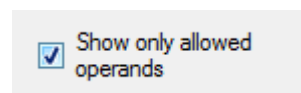
Usually, the symbolic signals name from STEP 7 is adopted as signal name in the I/O Manager. Optionally, you can adopt the STEP 7 signal comment as signal name in the I/O Manager by selecting the option "Use comment as signal name".
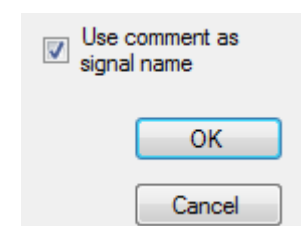
3.3.5.3 Measuring the CFC connectors

For selecting CFC connectors for the measurement process, proceed basically as described in chapter *Selection via the operand symbols*, page 51, until opening the *S7-CFC and symbol browser*.

Note

In case the connectors are not displayed on the *CFC* tab, then possibly the SCL sources have not been translated in the STEP 7 project.

To solve this problem, make sure that the following options are selected in the dialog box for compiling the program in the SIMATIC software:
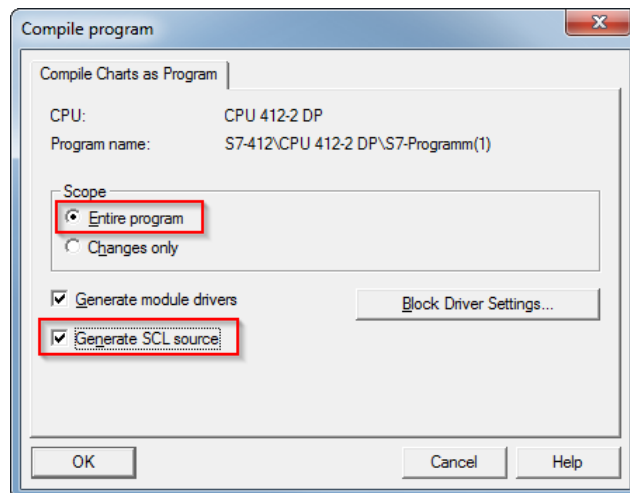
Afterwards, go back and generate the address books again.

Important note

When compiling a CFC program, generated DB addresses are assigned automatically to the connectors in STEP 7. Depending on the scope of program changes adopted between two compilation runs, connectors may have assigned different DB addresses.

In that case, the address book for *ibaPDA* must be newly generated, too. The symbolically configured signals are checked automatically in the I/O Manager and the associated absolute S7 operands are updated.

In the *S7-CFC and symbol browser* you can select the *CFC* tab instead of the *Symbol* tab and select the signals. The connectors are listed hierarchically by chart name, module name and connector name:
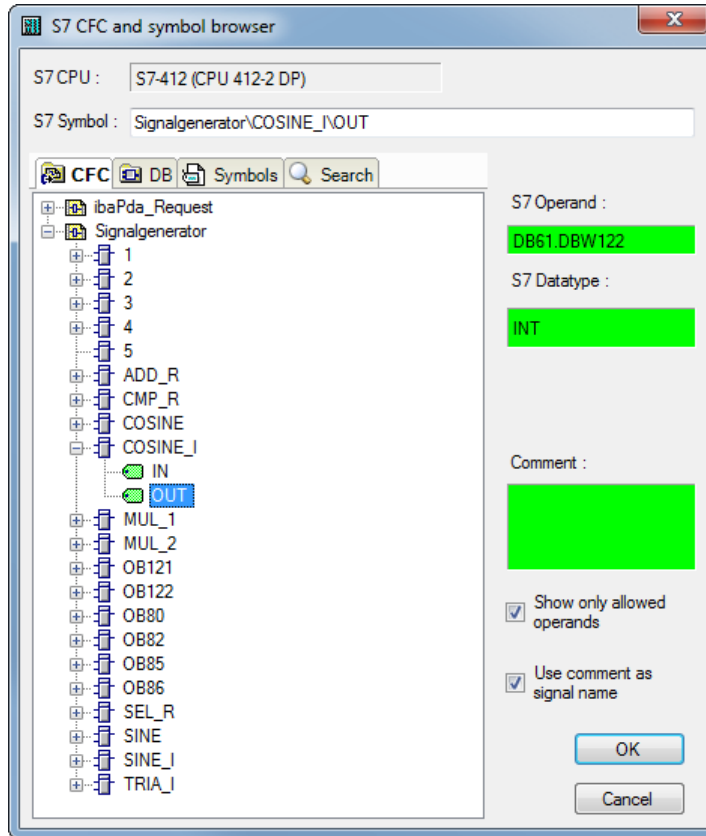
Figure 18:   S7-CFC and symbol browser

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*.

For this purpose, display the CFC editor on the same PC in the background as full screen and the *ibaPDA* client in the window mode in the foreground. Then, start the I/O Manager.
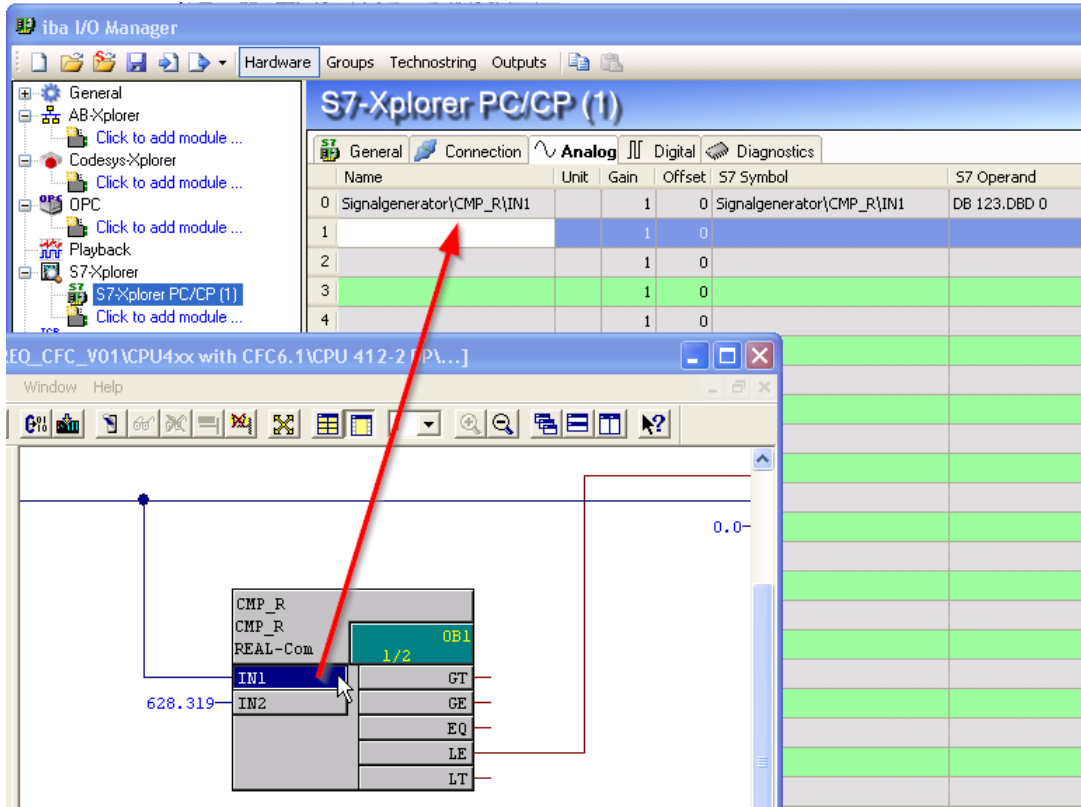
Figure 19:   Selecting signals via drag & drop

Click the connector in the CFC chart, hold the mouse key down, drag it onto the I/O Manager and drop it in the desired signal row. Now, the CFC connector has been entered as measurement signal in the signal table of the Request module.
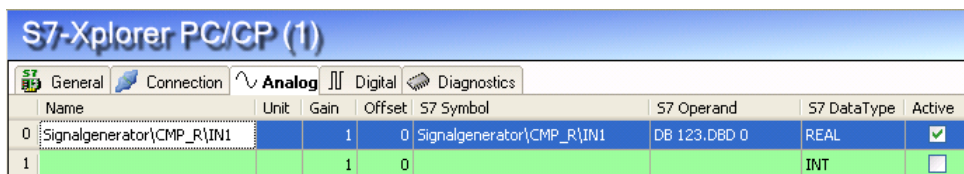
Figure 20:   Added signal

3.3.6   Module S7 Request

With the "S7 Request" module, analog and digital signals can be acquired.

In *ibaPDA*, up to 1024 analog and 1024 digital signals can be defined. Together, these may occupy a max. of 1466 Byte (max. length of the user data of an UDP telegram).

3.3.7   Module S7 Request Decoder

With the "S7 Request Decoder", up to 11728 digital signals can be acquired that are sent in form of a max. of 733 words (1466 Byte).

The operand to be addressed (absolute or symbolic) is of the WORD data type. Individual digital signals cannot be addressed.

On the *Digital* tab, you can directly enter the words as basic signals for the decoding via absolute S7 operands. S7 symbols can be used too by using address books.

The signals selected in S7 CFC and symbol browser are applied and the columns "Name", "S7 symbol", "S7 Operand" and "Data Type" are filled in automatically.

Click the <+> symbol to the left of each line to open a list in which you can enter a name for each of the 16 digital values.
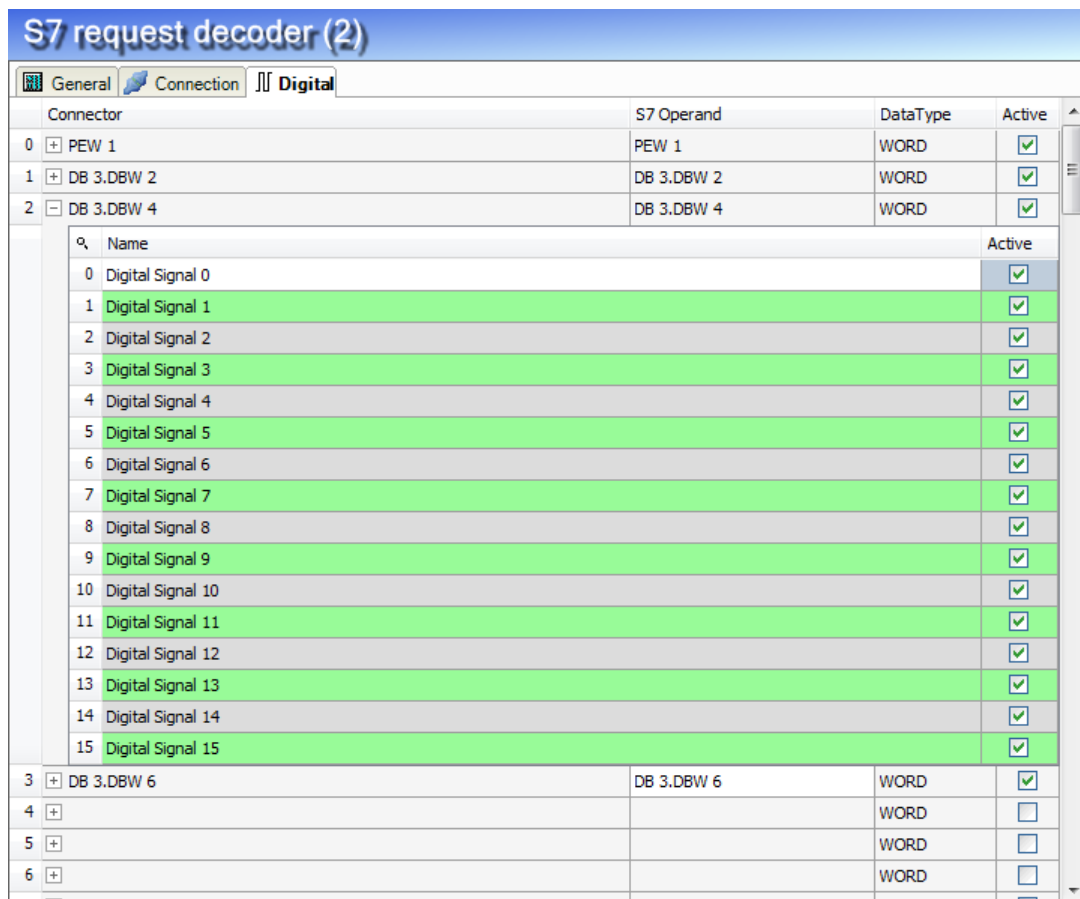
Figure 21:   Modul S7 Request Decoder

3.3.8   Module diagnostics

On the *Diagnostics* tab all configured operands are listed as a table with their data type and actual value.

Figure 22:   Module diagnostics analog values

Figure 23:   Module diagnostics digital values

On the *S7 request info* tab the data sent to the S7 CPU, and the data that are reported back as well as the general diagnostics data are displayed.

Figure 24:   Module diagnostics S7 request info

❑ 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 38)

❑ 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 telegrams received last
  ▪ Min: shortest time
  ▪ Max: longest time

❑ <Reset>
Resetting the min. and max. values.

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

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

❑ 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. One address book can be used in more than one module.
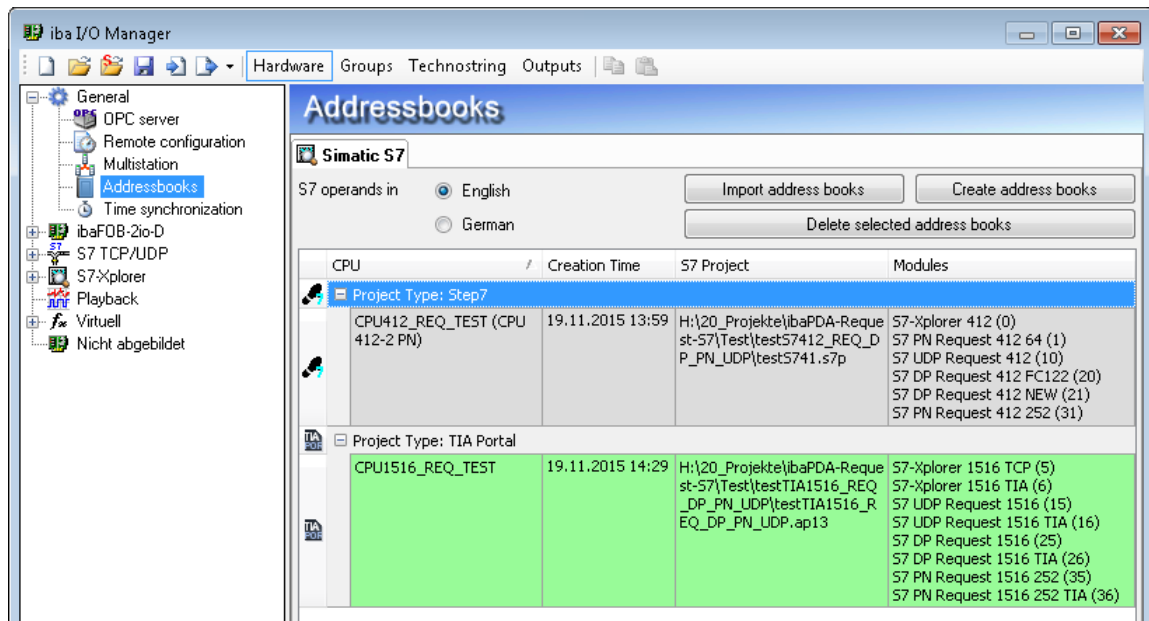

Figure 25:   S7 address book

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

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

- 📖 TIA Portal: TIA Portal Project

❑ S7 operands in English / German
Selection of the signal mnemonics

❑ Import address books
Import of address books which have already been created

❑ Create address books
Creates new address books from STEP 7 projects

❑ Delete selected address books
Deletes address books from a table

❑ Table
List of all address books that are currently available in the system with name, creation data, path of the STEP 7 project and location where the address book is used.

Creating an address book

An S7 address book can either be created offline from an S7 project (both STEP 7 CLASSIC and TIA-Portal) or online directly from a CPU (S7-1200 / 1500 only).

Offline from S7 project

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

Open the S7 address book generator with the <Create address books> button. Optionally, you can also open it under "S7 - CPU Name" on the *General* tab in the module configuration. Select "Create addressbook..." in the dropdown menu.
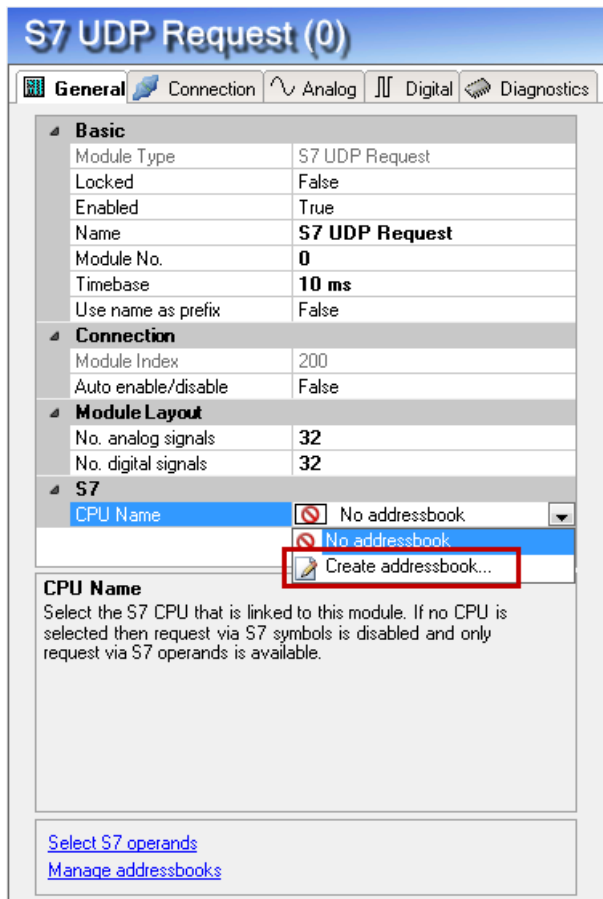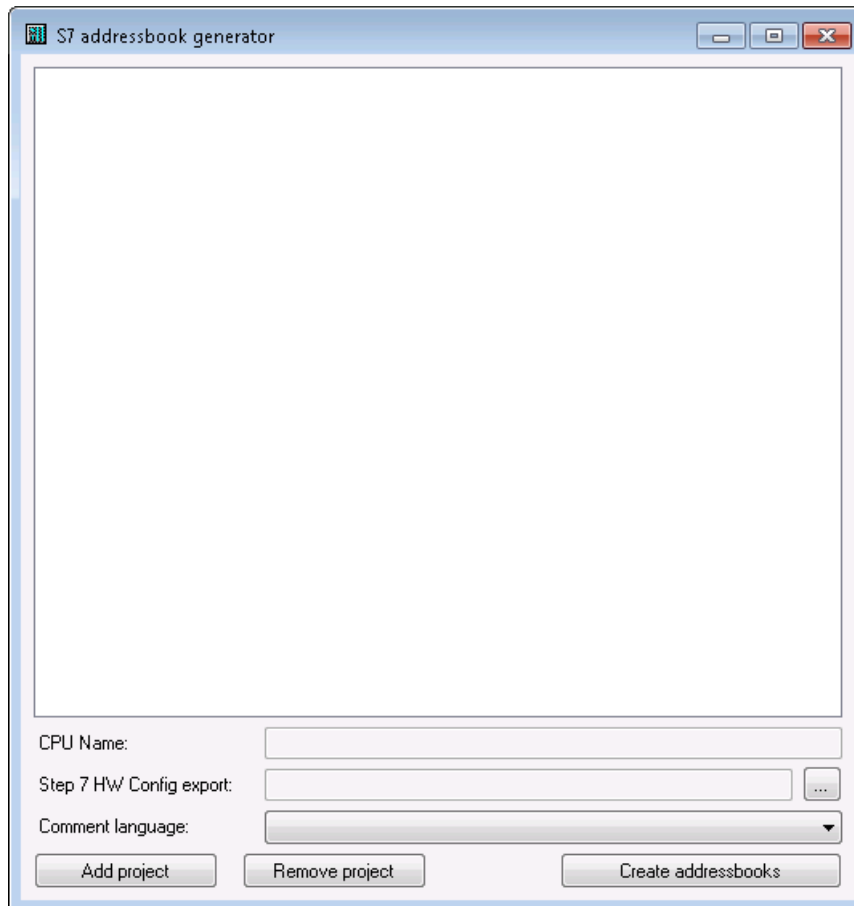

Figure 26:   Create S7 address book from module configuration

How to proceed in the address book generator:

1. Click on the <Add project> button

❑ CPU Name
CPU designation

❑ Step 7 HW Config export
optional selection of a HW Config export file (useful when using an iba busmonitor in sniffing mode)

❑ Comment language
Imported comment language can be selected (only available in SIMATIC TIA Portal projects)

❑ Add project
Adds a new project to the list

❑ Remove project
Removes the marked project from the list

❑ Create addressbooks
Creates address books from the selected projects

2. Select a project file in the file browser

3. Now, the STEP 7 project with all configured CPUs is displayed. Mark the CPUs you want to create the address books from and click on the button <Create addressbooks>.
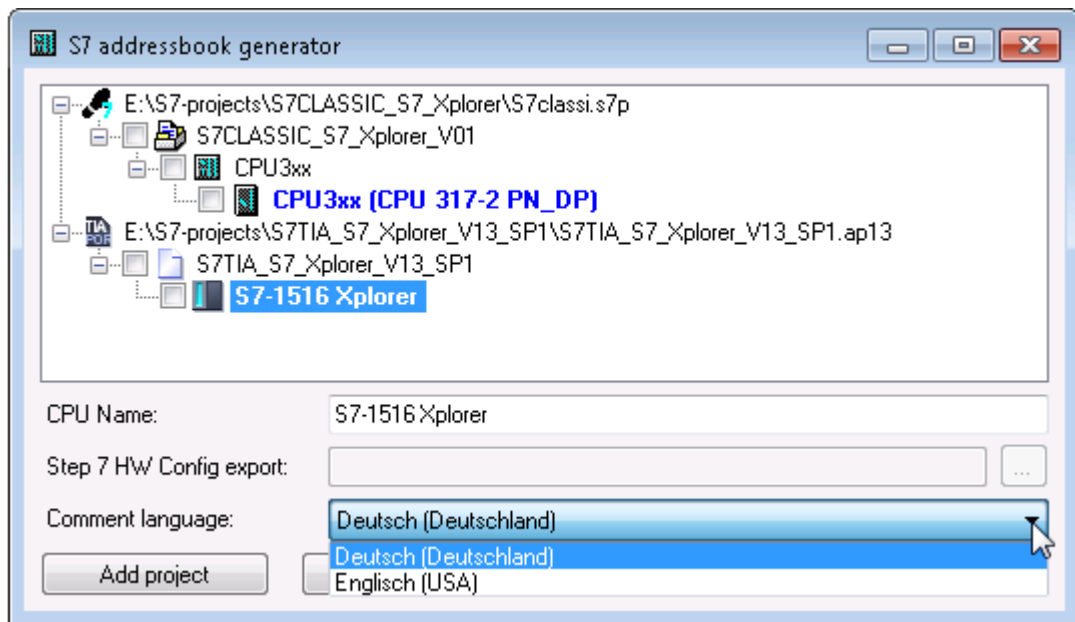
4. Now, you can select the generated address book in the selection list.

Note

The entry in the "CPU Name" field can be overwritten. 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.

Note

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

Online from S7-1200 / 1500 CPU

Online address books can be created 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 button <Load addressbook from S7> to load the address book.

The CPU name of the address book will be given automatically.
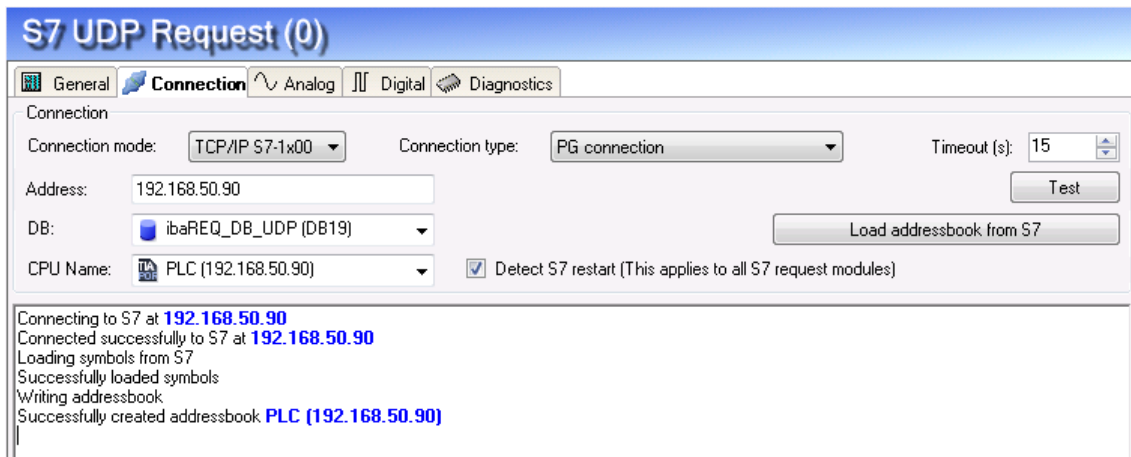
Figure 27: Load address book online from CPU

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

4   Diagnostics

4.1   Checking the license

If the "S7 UDP Request" modules are not available in the signal tree, you can either check in *ibaPDA* in the I/O Manager under *General - Settings - License Info* or in the *ibaPDA* service status application if your license is detected properly. The number of licensed connections is shown in brackets.
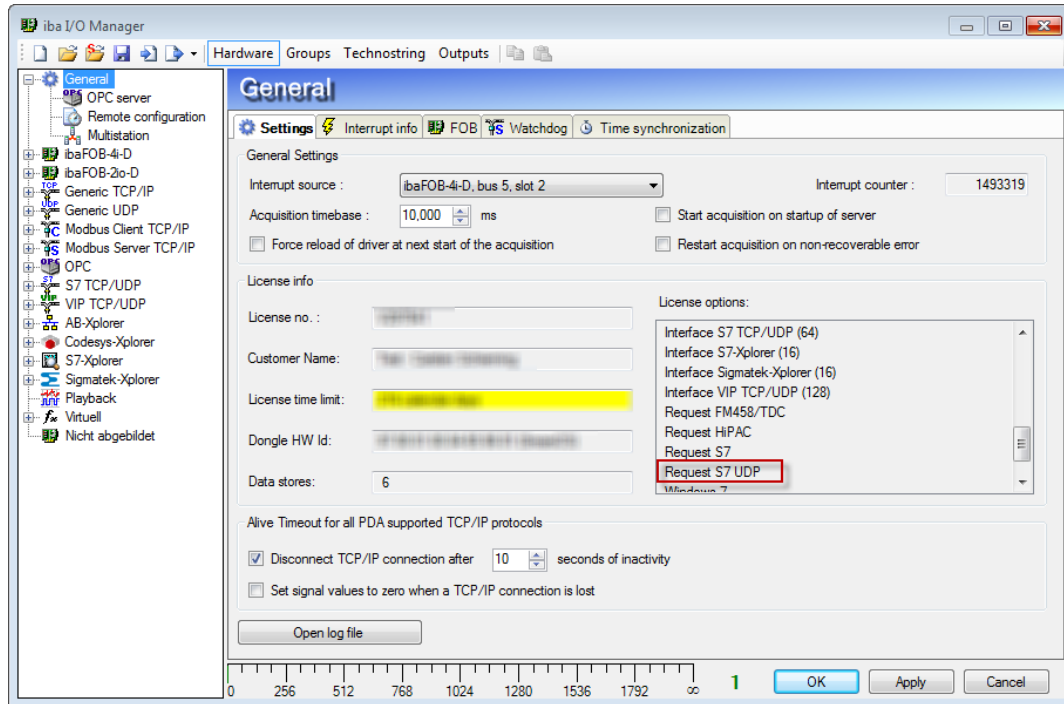

Figure 28:   Checking the license

4.2   Log files

For many interfaces, there is an <Open log file> button in the specific interface overview in the I/O manager.

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

In the file system on the harddisk, you will find the log files in the program path of the *ibaPDA* server (...\Programs\iba\ibaPDA\Server\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** (EtherNet/IP connections log)
- **AbEthLog.txt** (log of Allen-Bradley Ethernet connections)

4.3   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.

Open a Windows command prompt.

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

With an existing connection you receive several replies.

Figure 29:   PING successful

With not existing connection you receive error messages.

Figure 30:   PING unsuccessful.

4.4   Connection table

All Ethernet based interfaces have a table in the I/O Manager, that shows the status of the individual connections. Every row represents a connection.

Depending on the type of the connection, the columns contain different values and information.

Usually, the target systems to which the connection is established are identified in the first row (left) with their name or their IP address.

The table shows the package sizes and error counters of the individual connections during data acquisition. Click on the <Reset counter> button for resetting the counters and the calculation of the response times.
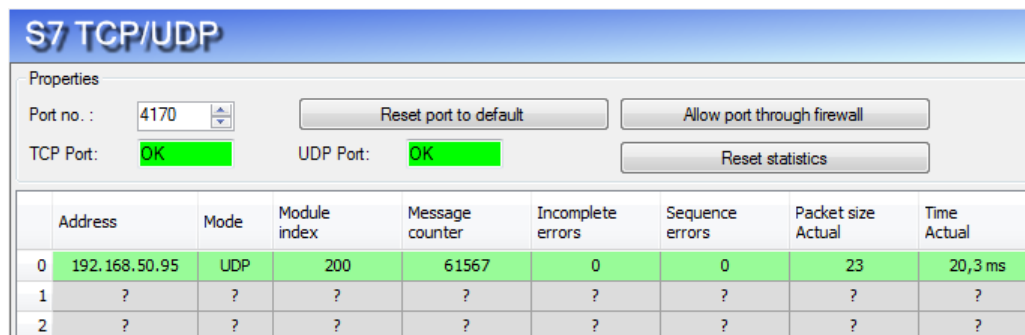
In case of communication problems, check the module indices, the package sizes and if there are sequence errors.

Each UDP telegram sent successfully, gets a consecutive sequence number. If *ibaPDA* detects an error in the sequence, the sequence error counter is incremented.

The background color of the cells provides additional information:

Color	Meaning
Green	The connection is OK and the data are read.
Orange	The connection is OK, but there is an error (e.g. wrong module index, configured update time is not reached,...)
Red	The connection has failed or was interrupted.
Grey	No connection is configured.

Table 10:   Color code for the background colors of the connection table

Figure 31:   Connection table

4.5   Module diagnostics

You will find a diagnostic help with a tabular display of the actual analog and digital values and the data types on the *Diagnostics* tab of each S7 Request module.

🡵 See chapter *Module diagnostics*, page 58.

5   Appendix

5.1   iba S7 library

The iba S7 library is available in two versions:

❑ SIMATIC Manager : STEP 7 ≥ V5.5

❑ SIMATIC TIA-Portal : STEP 7 ≥ V13 SP1

5.1.1   iba S7 library for SIMATIC manager

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

You find the iba S7 library as an archived file on the DVD "iba Software & Manuals" under \04_Libraries_and_Examples\10_Libraries\01_SIMATIC_S7_

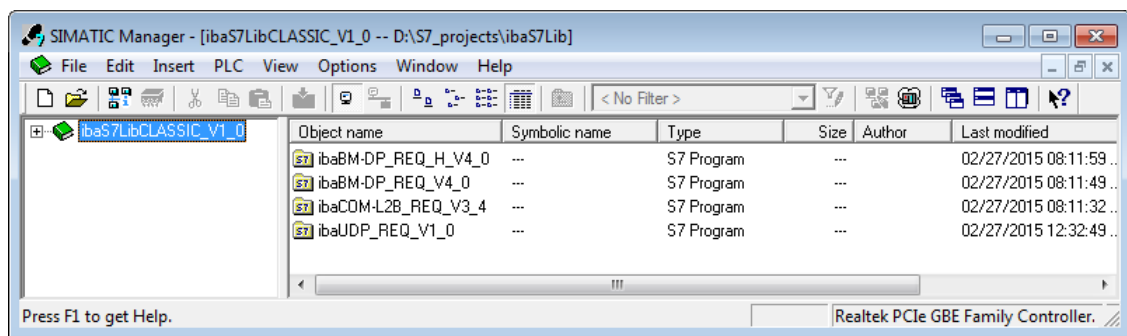Figure 32:   iba S7 library for SIMATIC Manager

The following blocks are available:

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_DB	DB15	
	ibaUDT_UDPact	UDT145	
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

iba connection	Block name	Block no.	Note
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 11:   ibaS7LibCLASSIC block overview

5.1.1.1   Integrating the library

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.
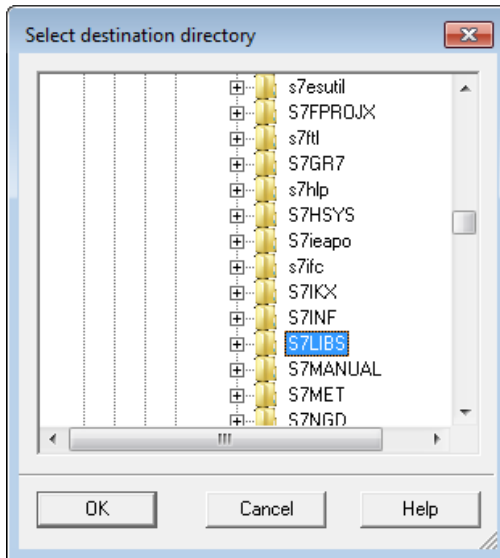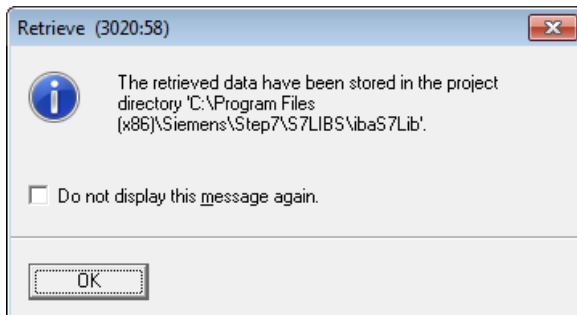
❑ Select the menu „File – Retrieve...“
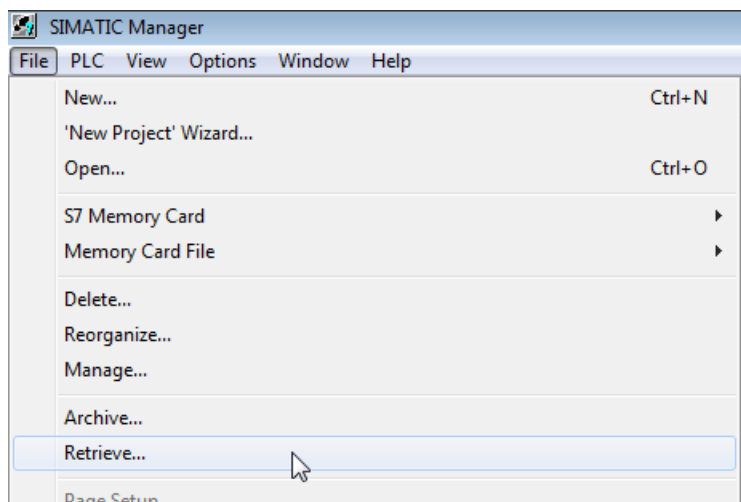
❑ Choose the archive file of the iba S7 library and select a storage location for the extracted library in a next step:
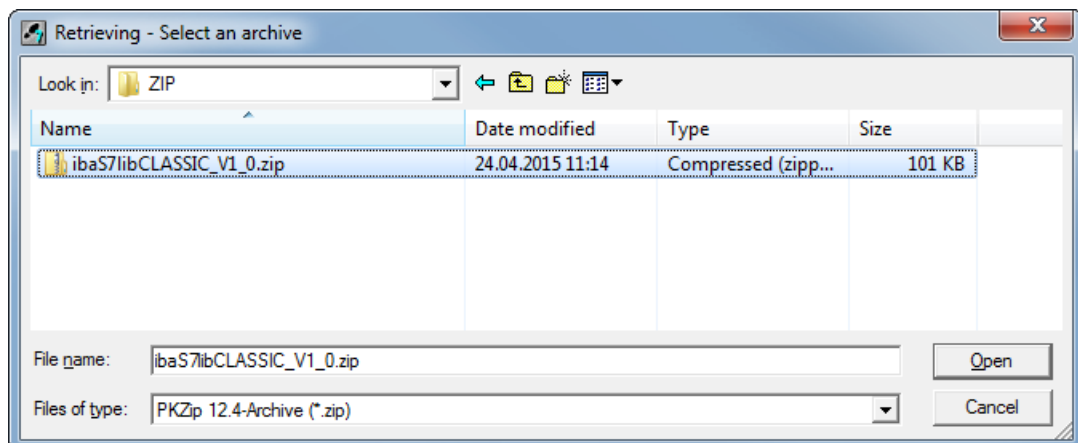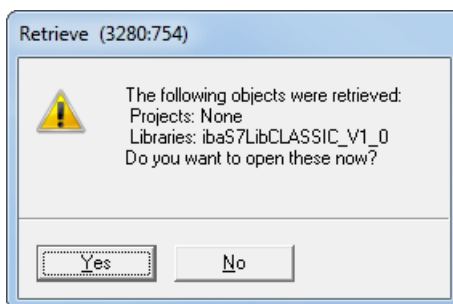
❑ Confirm the message for a successful extraction.

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

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

5.1.1.2 Using the blocks

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

1. Display the block library on the left and drag the required blocks to the opened destination block.
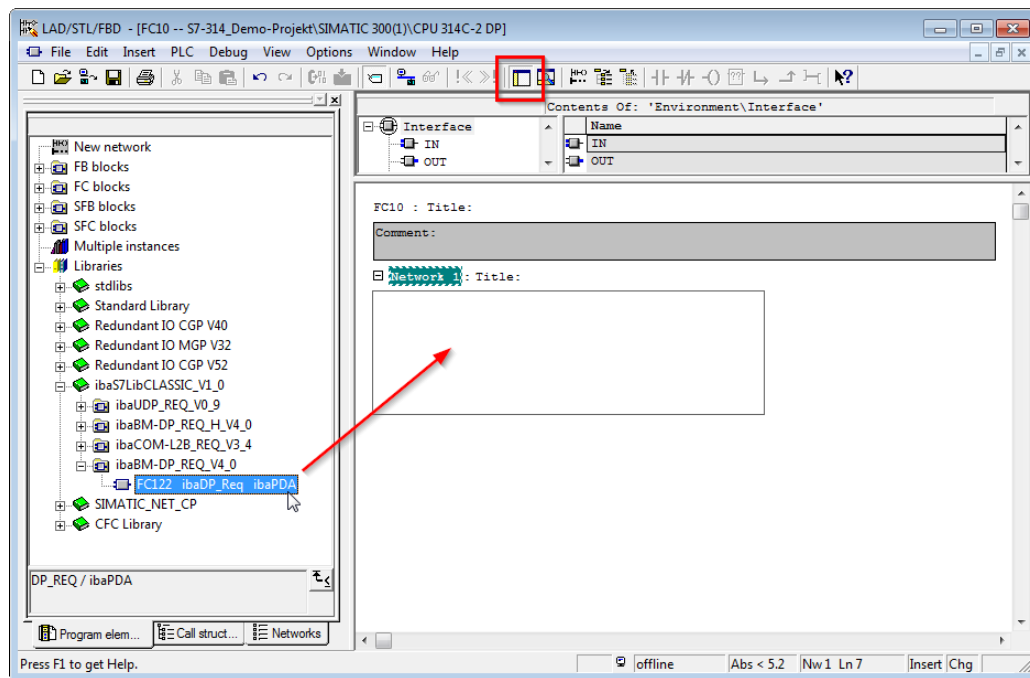
2. Open the library via "File - Open - Libraries", and the required destination project in parallel. With the <Tile horizontal> button, both projects can be displayed side by side. The blocks can be copied either via drag or copy & paste.
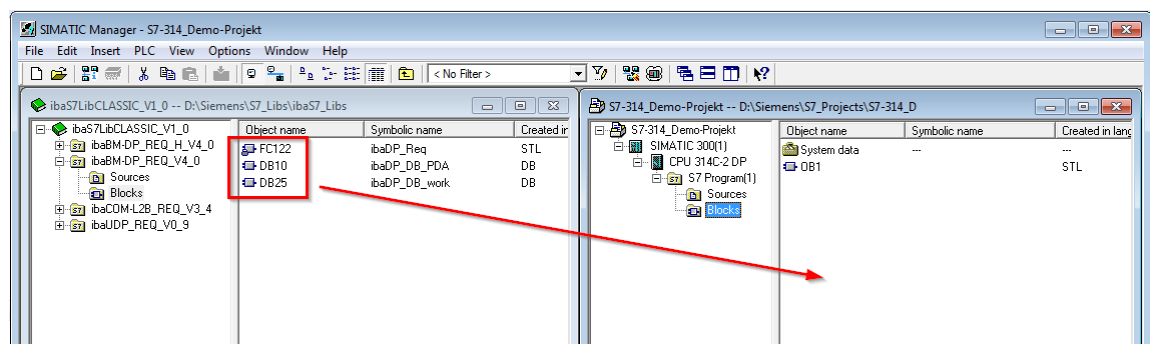
The blocks may now be used in the destination project.

5.1.2   iba S7 library for SIMATIC TIA portal

The iba S7 library for SIMATIC TIA Portal („ibaS7LibTIA_Vx_y") is suited for the use with SIMATIC TIA Portal V13 SP1 or higher. It contains the Request blocks described in the manual, which are required for the use of i*baPDA-Reque*st-S7-UDP.

You find the iba S7 library as project archive on the DVD "iba Software & Manuals" under **\04_Libraries_and_Examples\10_Libraries\01_SIMATIC_S7**
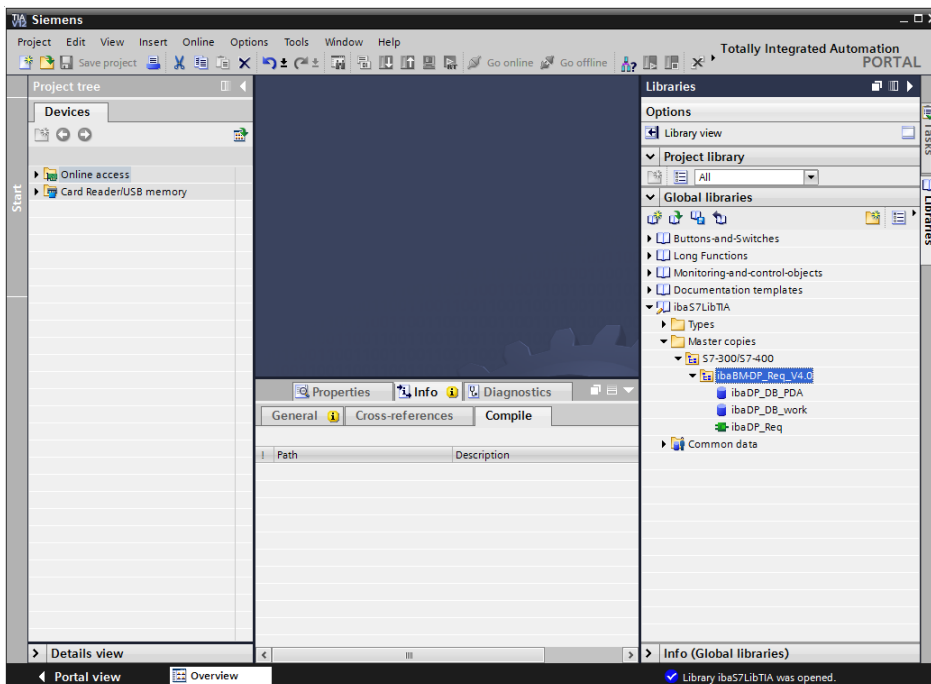
Figure 33:   iba S7 library for SIMATIC TIA Portal

The following components are contained:

iba connection	Block name	Block no.	Note
Target platform S7-300, S7-400, WinAC			
ibaBM-DP	ibaDP_Req	FC122	
	ibaDP_DB_PDA	DB10	
	ibaDP_DB_work	DB25	
ibaBM-PN	ibaREQ_M	FB140	
	ibaREQ_PN	FB141	
	ibaREQ_DB	DB15	
	ibaUDT_UDPact	UDT145	
ibaCom-L2B	ibaL2B_Init	FC111	
	ibaL2B_Req	FC112	
	ibaL2B_Req_CP	FC113	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

| iba connection	Block name	Block no.	Note
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	
Target platform S7-1500			
ibaBM-DP	ibaREQ_M	FB1400	
	ibaREQ_DP	FB1402	
	ibaREQ_DB	DB15	
	ibaREQ_DB-Interface		
ibaBM-PN	ibaREQ_M	FB1400	
	ibaREQ_PN	FB1401	
	ibaREQ_DB	DB15	
	ibaREQ_DB-Interface		
ibaPDA-Interface-S7-TCP/UDP	ibaREQ_M	FB1400	
	ibaREQ_UDP	FB1405	
	ibaREQ_DB	DB15	
	ibaREQ_DB-Interface		

Table 12:   ibaS7LibTIA block overview

Note

TIA Portal libraries are version dependent and not upwardly compatible, i. e. a library saved with TIA Portal V13 cannot be used with TIA Portal V12.

5.1.2.1   Integrating the library

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.

❑ Select the "Retrieve library..." command on the "Libraries" tab in the context menu (right mouse-button).
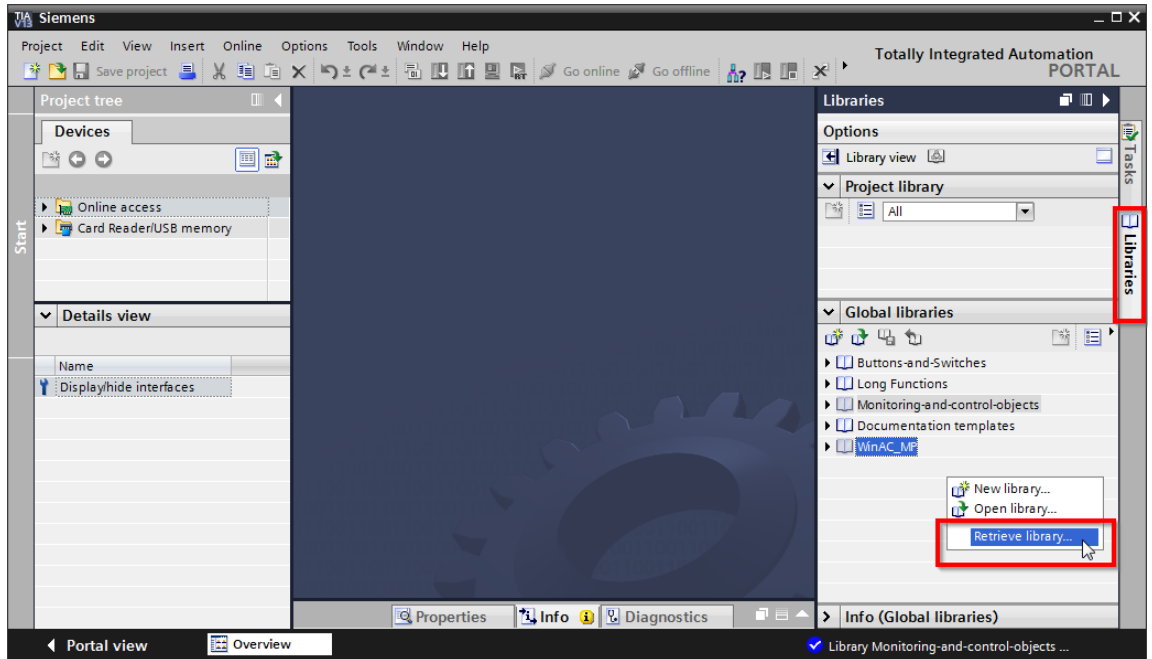
❑ Select the archive file of the iba S7 library and select the storage location for the extracted library in a next step:
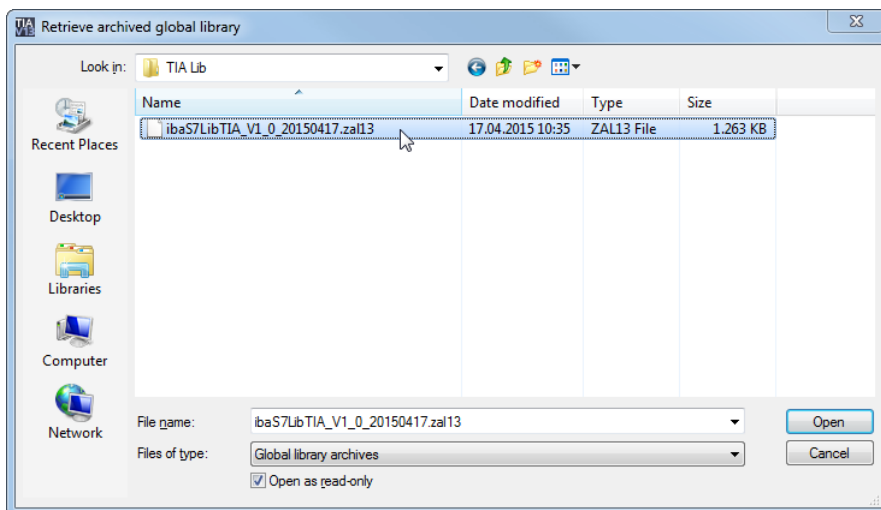
❑ Now, the library is integrated.
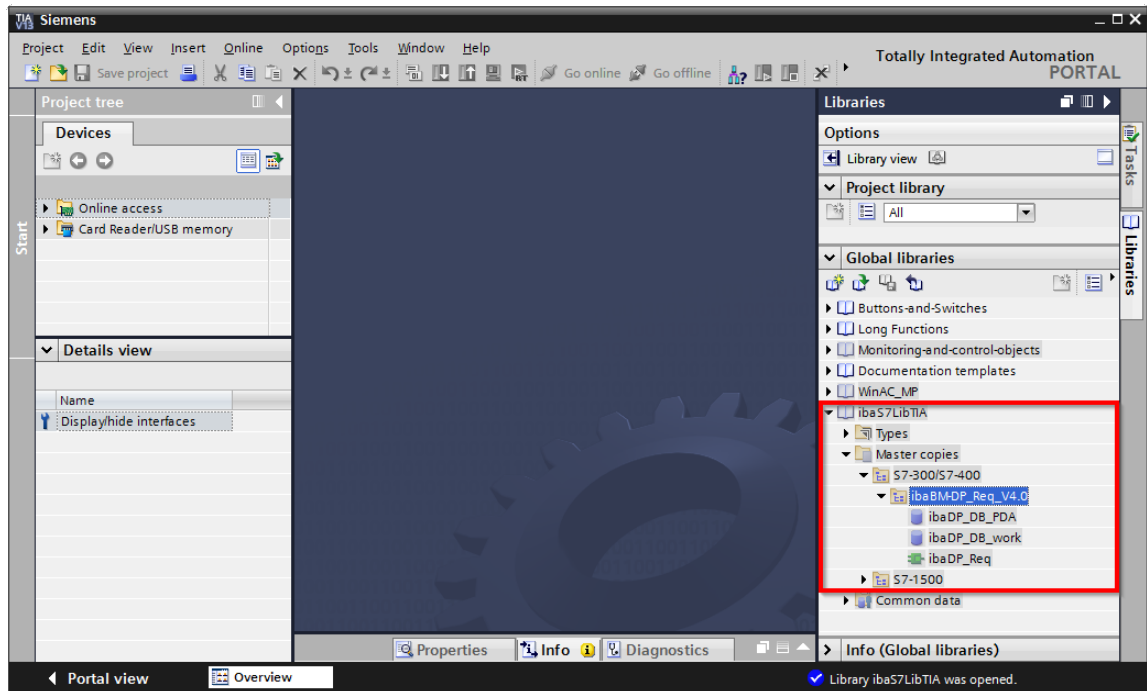
5.1.2.2   Using the blocks

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

1. Display the library and drag the required block to the opened destination block.
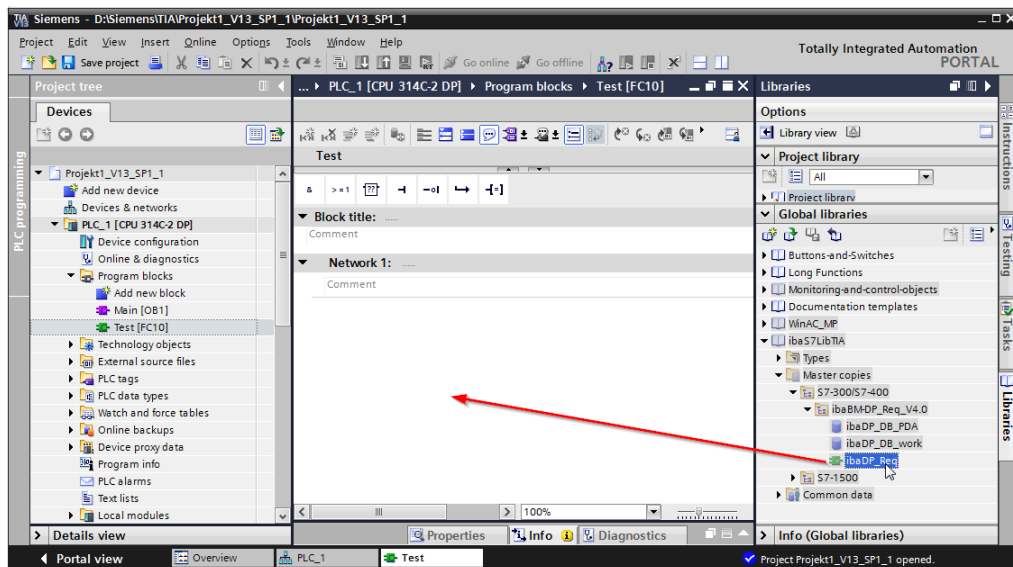
2. Drag or copy the blocks to the program blocks directory in the project navigation.

The blocks can now be called within a project block.

3. Drag or copy the PLC data type to the PLC data type directory in the project navigation.
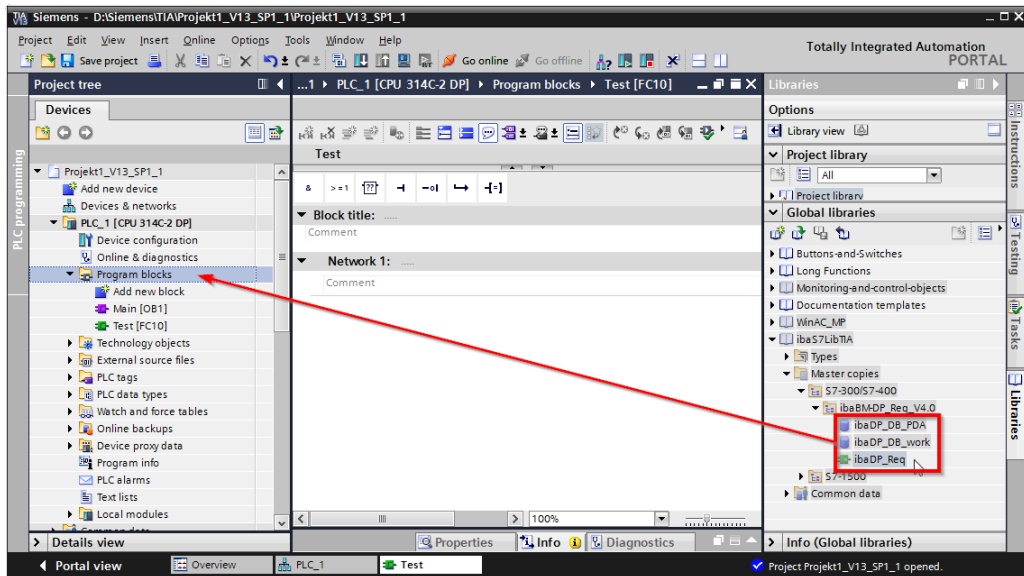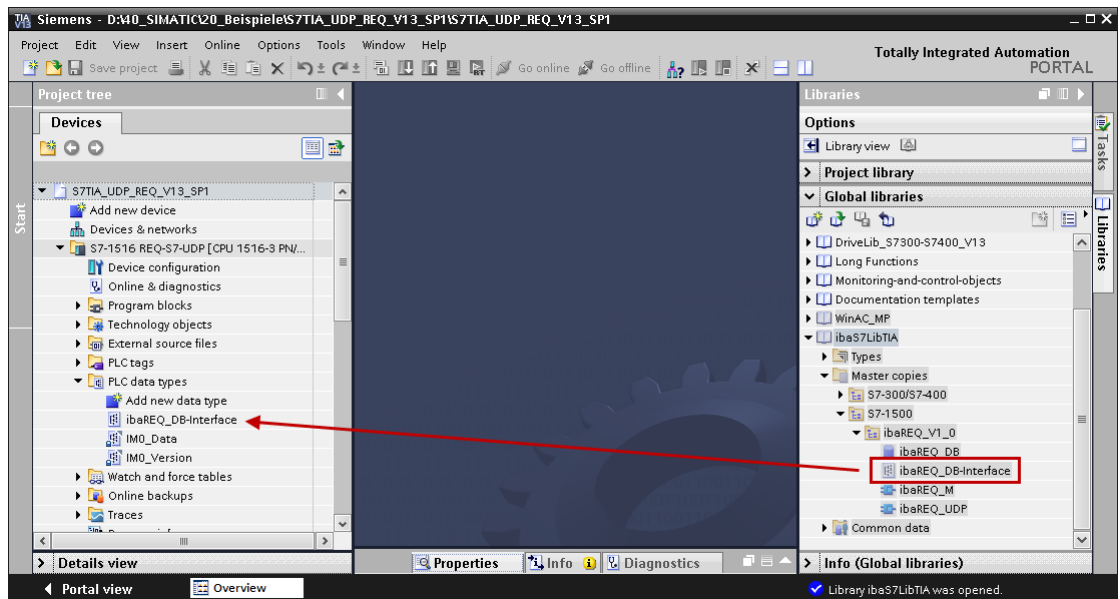
5.2   Application examples

You find application examples for several different configurations on the DVD "iba Software & Manuals" under:

\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

Table 13:   Application examples on the DVD

5.3   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. They 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

5.4   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	ibaAG, 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	ibaAG, ibaS7Lib

Table 14:   Subordinate blocks

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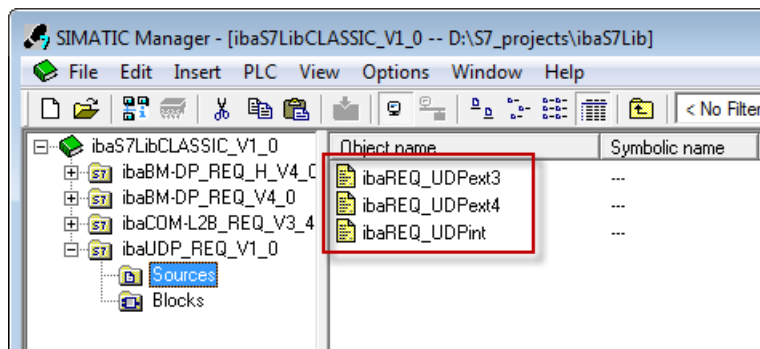

Figure 34:   Function block sources in the ibaS7 function block library.

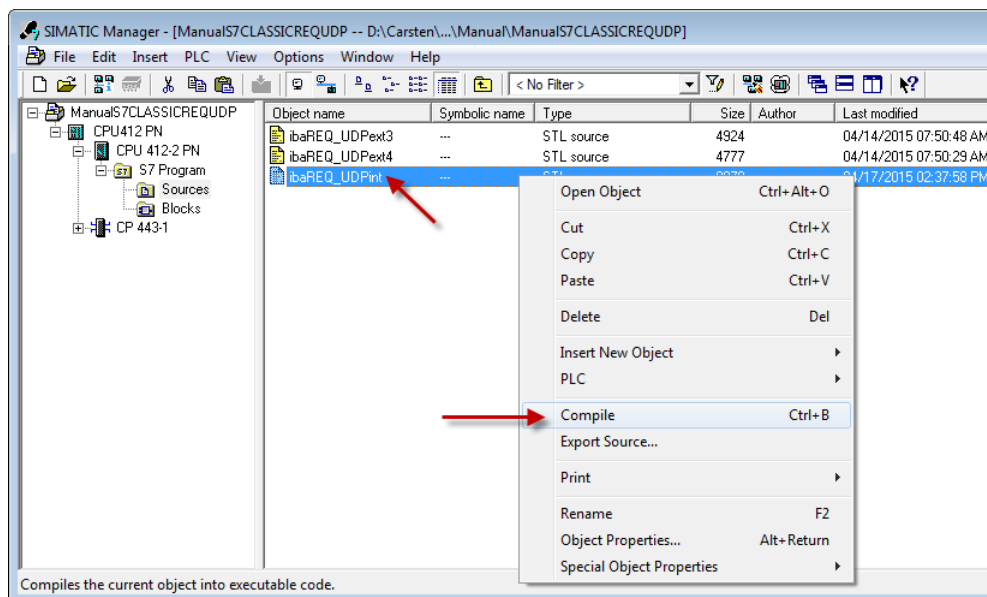❑ Make a new translation for all sources of the function blocks you use.

Figure 35:   Translating the function block sources

Important note

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

5.5   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 will be changed again in the SIMATIC Manager, the measurement does no longer work because *ibaPDA* no longer has any 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, a new access point should be defined.

There is the <Configure PG/PC interface> button in the dialog window of the PC/CP module. It can be used to open the dialog for configuring the PG/PC interface.

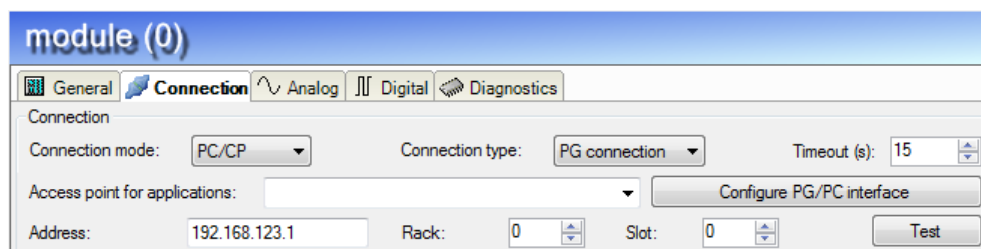The setting for the SIMATIC Manager will also be changed.

Figure 36:   Configure PG/PC interface

Procedure

1. Open the dialog box with the <Configure PG/PC interface> button.
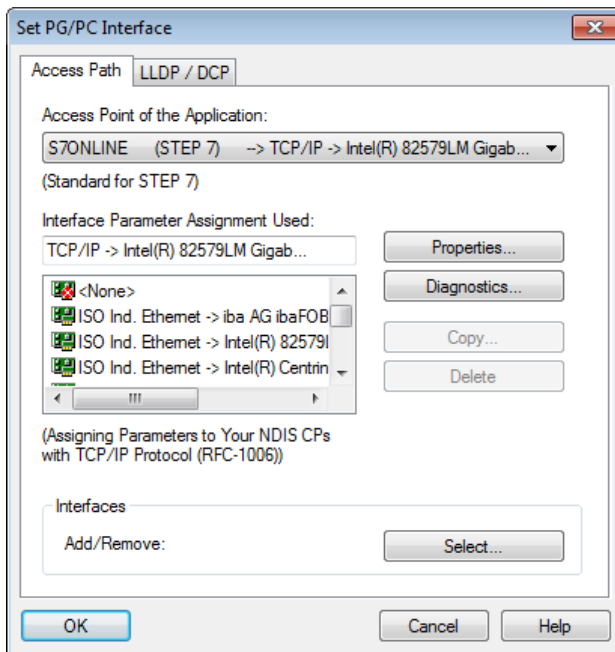
2. Select the row <Add/Delete> under "Access Point of the Application".
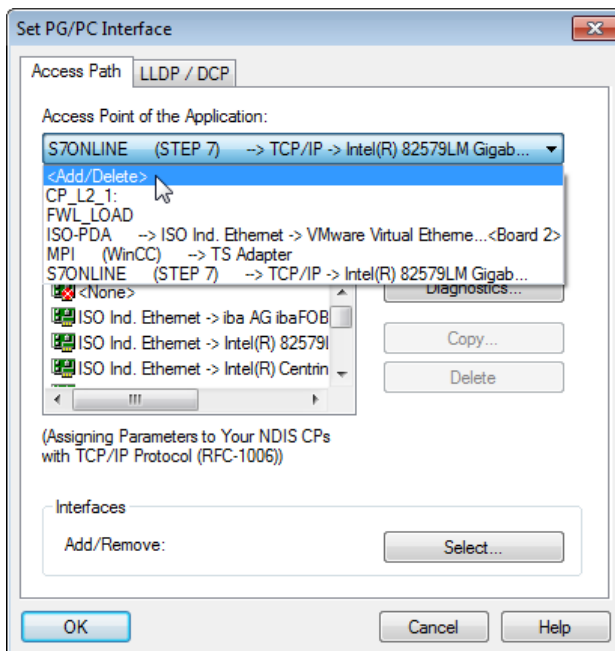
3. Define a new access point; enter a name, e.g. PDA and optionally a description for a better understanding, click on <Add> and <Close>.
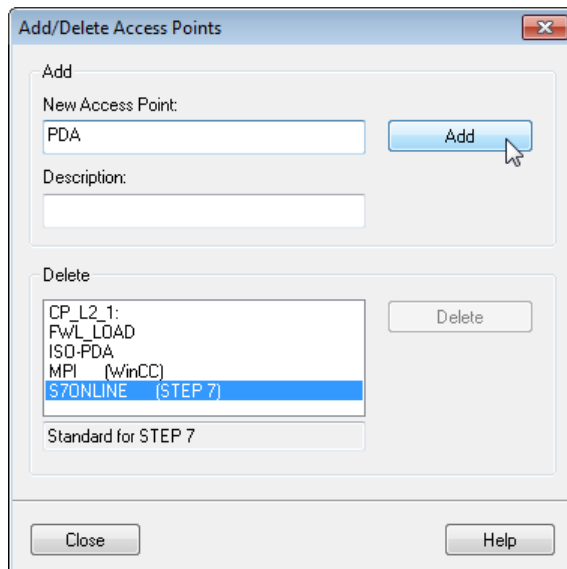
4. Add an interface to the access point, e.g. "CP5622 (MPI)" and exit with <OK>.
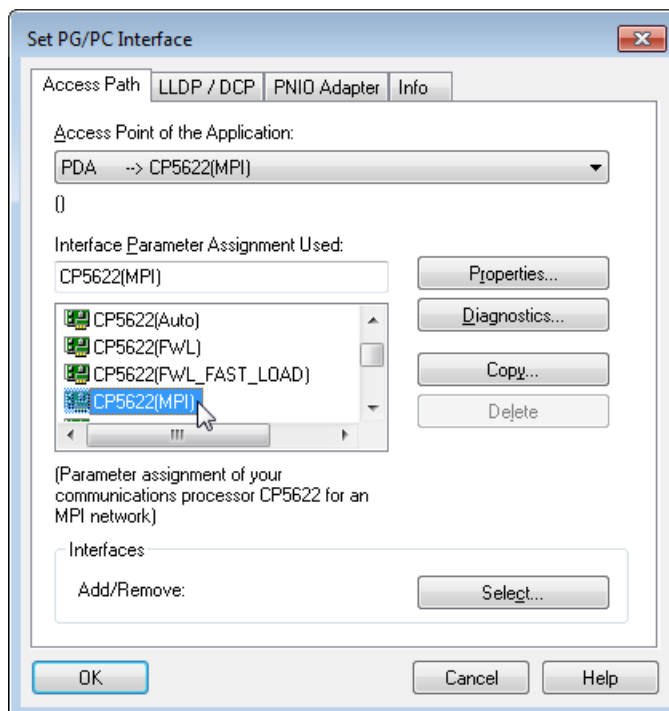
The newly defined access (e.g. PDA --> CP5622.MPI.1) is displayed subsequently in the connection dialog of *ibaPDA* under "Access points for applications".
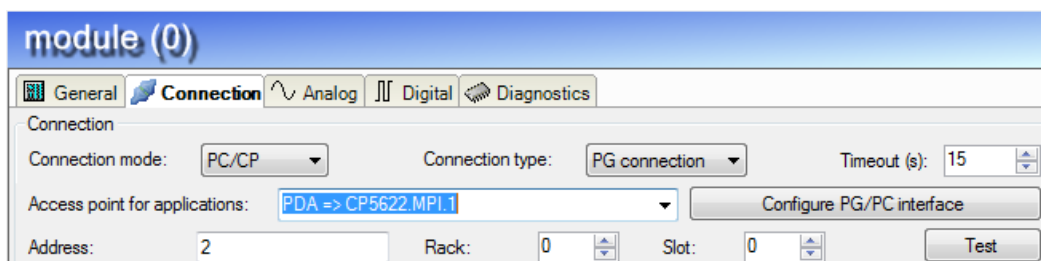
Figure 37:   Set access point

Notes on the different access points

Depending on which access points have been configured in the Engineering PC, 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 and slot number of the CP in the module configuration dialog. If you do not know the rack and/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 and slot number. For the rack and slot number, use the <Scan> button.

Bus system (PROFIBUS or MPI)

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

5.6   S7 routing

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

The "S7 routing" function is not to be confused with "IP routing".

The following constellation is to clarify this:

Figure 38:   S7 routing, system topology example

The engineering computer (also with *ibaPDA*) is to access the CPU412 control. The computer and the control system are not directly connected with each other via a joint network/bus. The connection is to take place via the CPU314C controller. "Forwarding" the communication in this controller is called "S7 routing".

In this example, the engineering computer and CPU314C are also in two different (logical) subnets. For a communication connection, an (IP) router needs to be used. This is completely independent of the "S7 routing" function and is not to be confused with it.

5.6.1   Routing from Ethernet to Ethernet

The function "S7-Routing" should not be confused with "IP-Routing".
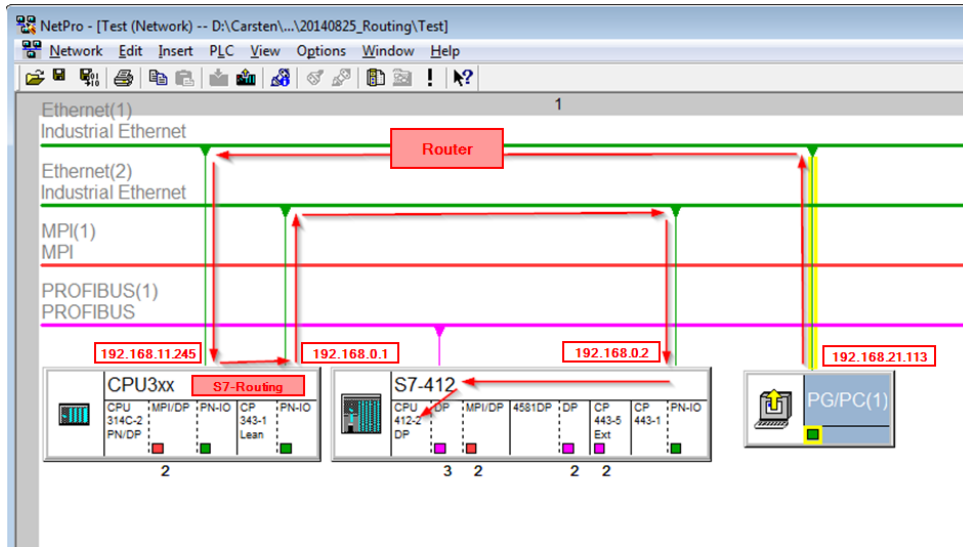
The following constellation is to clarify this:
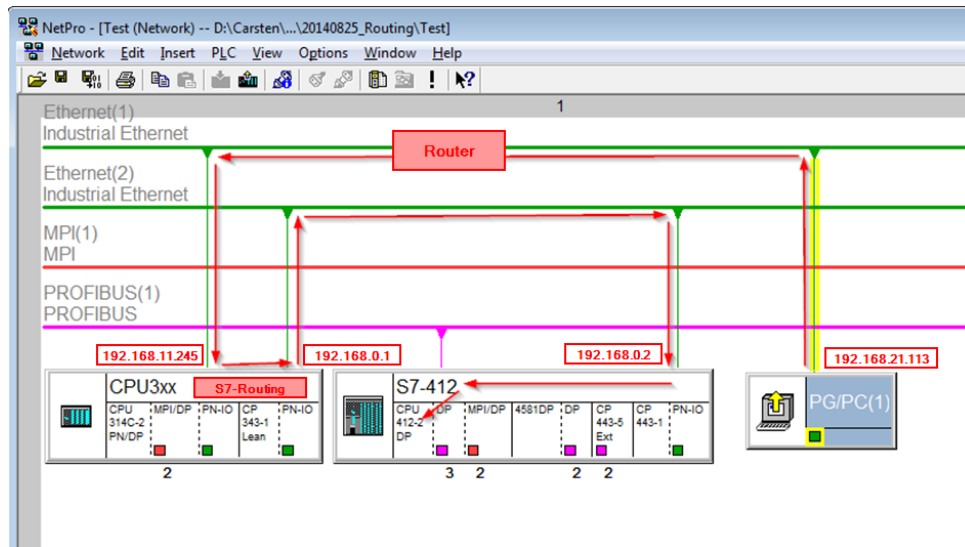
Figure 39:   S7-Routing, system topology example

The engineering computer (also with ibaPDA) is to access the CPU412 control. The computer and the control system are not directly connected with each other via a joint network/bus. The connection is to take place via the CPU314C controller. "Forwarding" the communication in this controller is called "S7 routing".

In this example, the engineering computer and CPU314C are also in two different (logical) subnets. For a communication connection, an (IP) router needs to be used. This is completely independent of the "S7 routing" function and is not to be confused with it.

5.6.1.1 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. These are not required for using *ibaPDA*.

Inserting a PG/PC station:
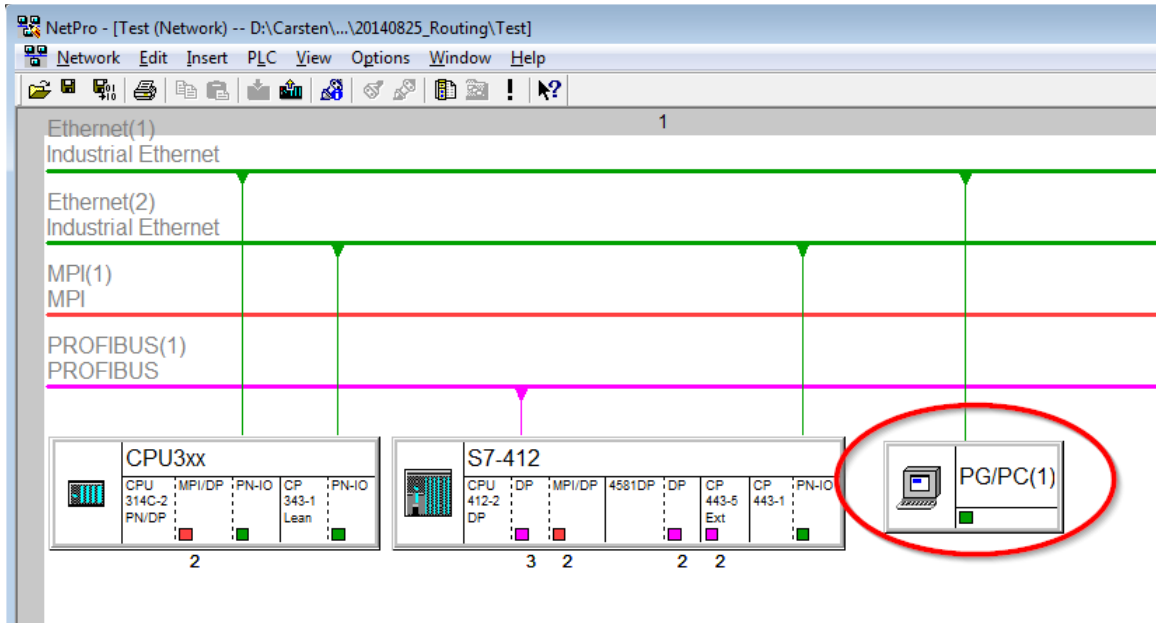
Figure 40: NetPro configuration

Assigning an interface (network interface card):
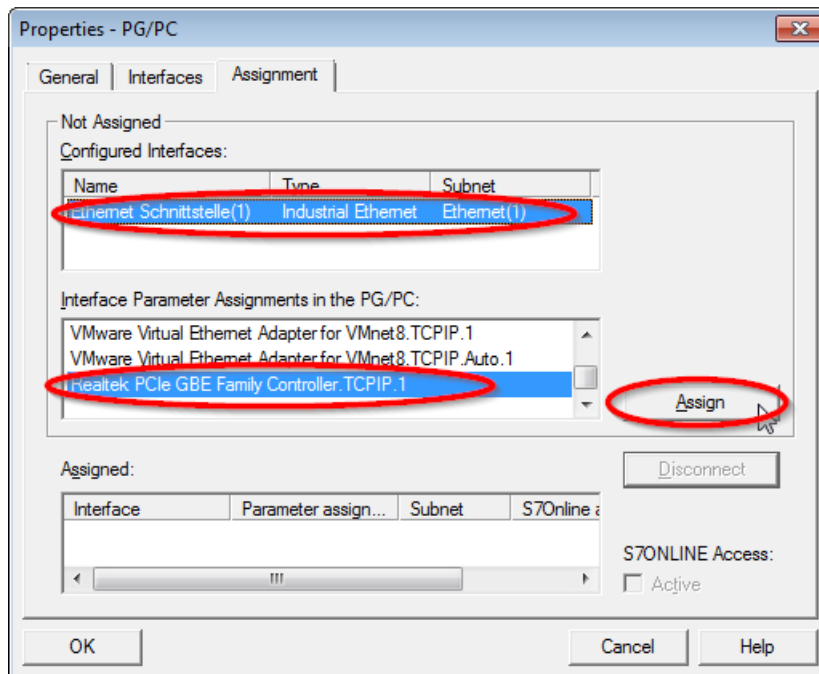
Figure 41: PG/PC interface assigned

Result:
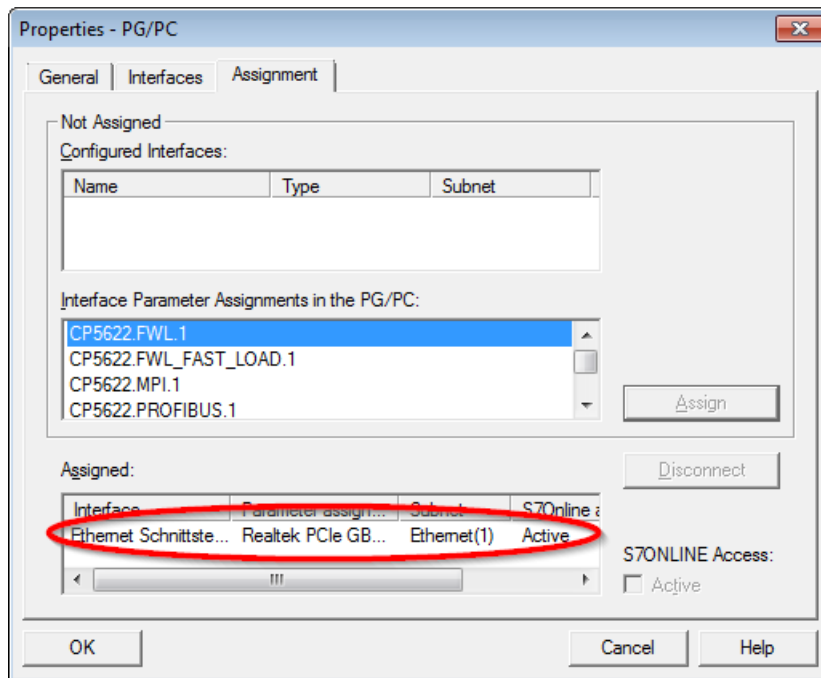
Figure 42:   Interface assigned

The connection line of PG/PC to the network should be highlighted yellow now.

The following figure shows the communication channel using arrows (these are not shown in SIMATIC NetPro).
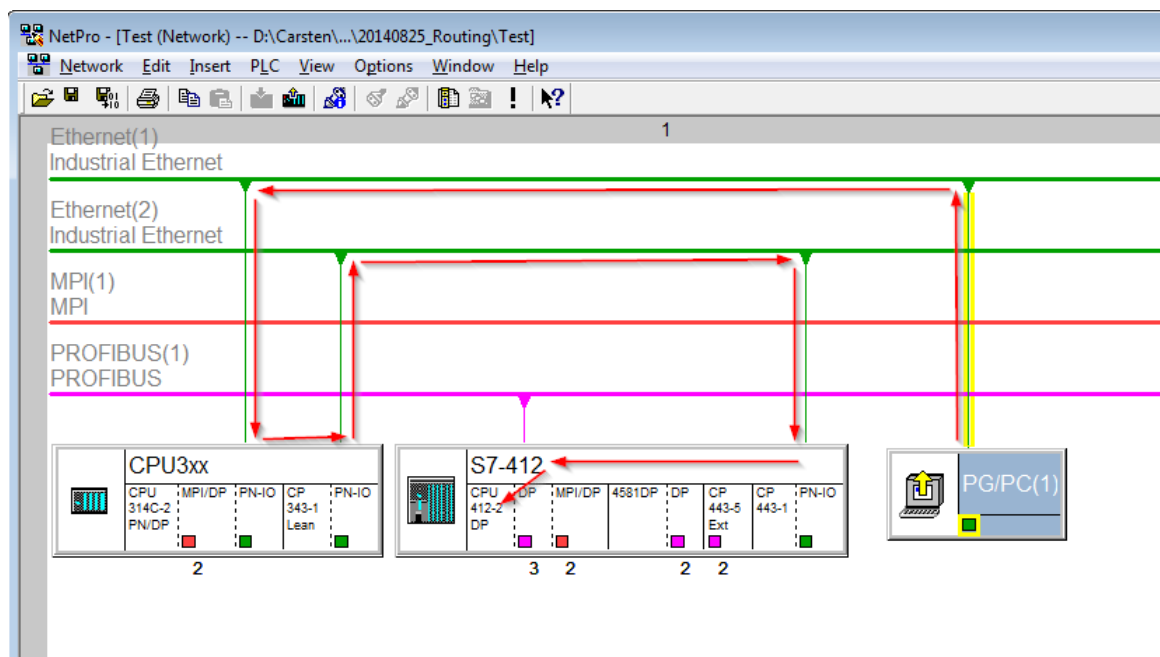
Figure 43:   Communication channel

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

5.6.1.2 ibaPDA configuration

The following entries have to be made:
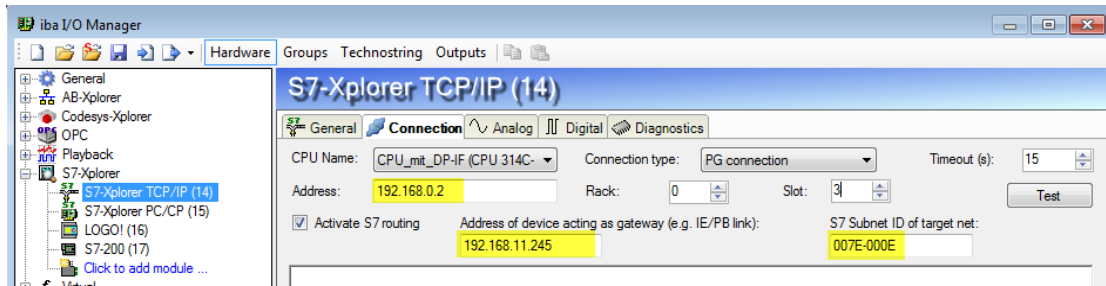
Figure 44:   Enable S7 routing

❑ **Activate S7 routing**
Enable to use S7 routing Address

❑ **Address**
Address of the target control (here CPU412)

❑ **Address of device acting as gateway**
Enter address of the gateway (here CPU314C)

❑ **S7 subnet ID of target net**
Enter subnet ID from STEP 7 NetPro

You can identify the S7 subnet ID in NetPro. For doing so, right-click on the secondary bus system and open the "Properties".

Figure 45:   Determine S7 subnet ID

Figure 46:   S7 subnet ID

Other documentation

For more information about the S7-Routing, please see:

Which modules support the "S7 Routing" function in S7 subnets?

Web http://support.automation.siemens.com/ww/view/en/584459:

What requirements must be fulfilled and what should I watch out for when executing Routing?

Web http://support.automation.siemens.com/ww/view/en/2383206:

How do you enable cross-project S7 Routing in the TIA Portal and in STEP 7 V5.x?

Web https://support.industry.siemens.com/cs/ww/en/view/109474569:

5.6.2   Routing from Ethernet to PROFIBUS

The following access path shall be established:
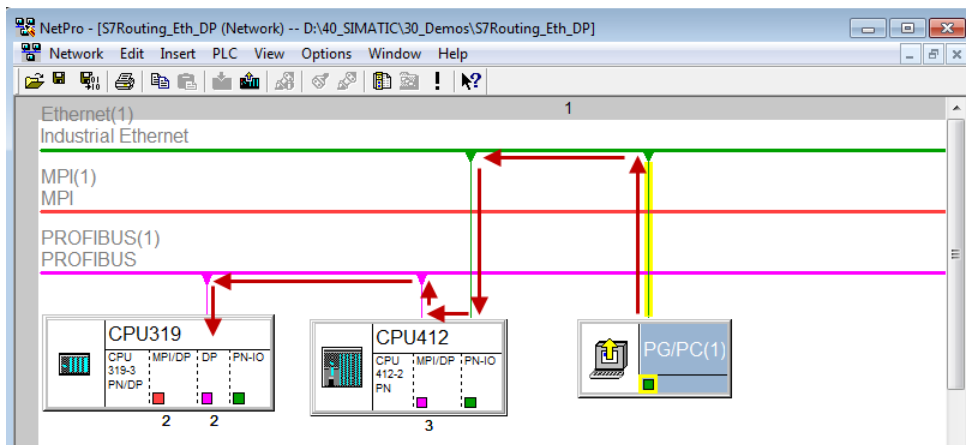
Figure 47:   S7-Routing, example for system topology Ethernet-PROFIBUS-Routing

The engineering computer (also with ibaPDA) is to access the CPU319 control. The computer and the control system are not directly connected with each other via a joint network/bus. The connection is to take place via the CPU412 controller. "Forwarding" the communication in this controller is called "S7 routing".

5.6.2.1   Configuration of STEP 7 / NetPro

The following configuration steps are required exclusively for accessing the subordinate control CPU319 by the programming software SIMATIC STEP 7. These steps are not required for the use of ibaPDA.

Insert a PG/PC station:
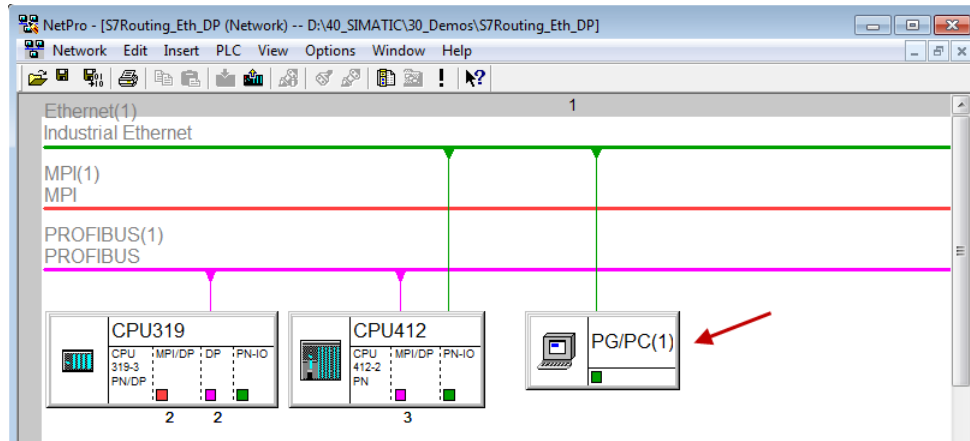
Figure 48:   Configuration NetPro

Assign an interface (network adapter):
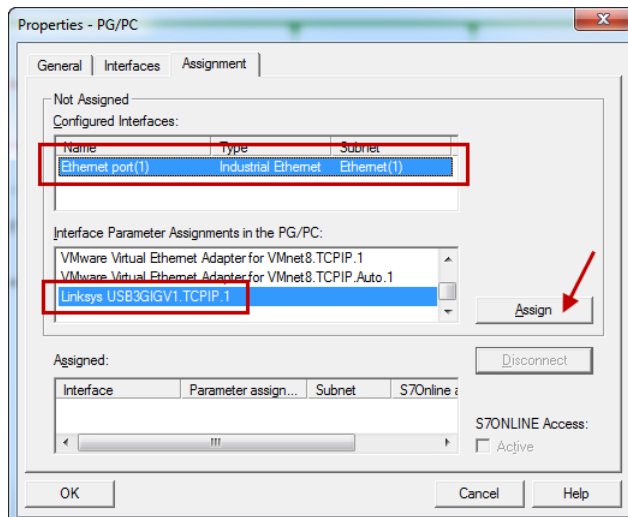
Figure 49:   Assigning a PG/PC interface

Result:
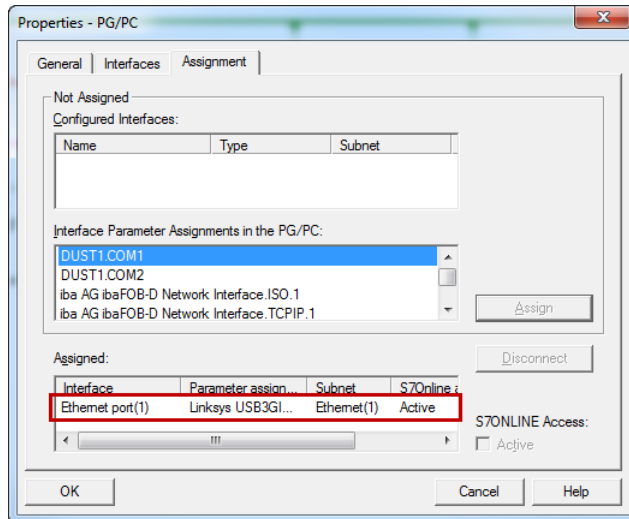
Figure 50:   Interface assigned

The connecting line between PG/PC and the network should be highlighted yellow now.

In the following figure the communication path is represented by arrows (which are not displayed in SIMATIC NetPro).
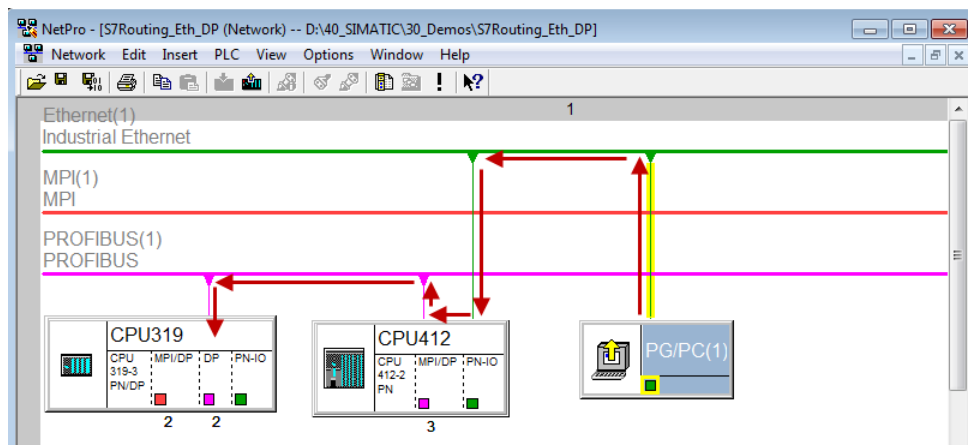
Figure 51:   Communication path

Finally, load all HW configs and connection data by NetPro.

5.6.2.2   Configuring ibaPDA

Make the following settings:
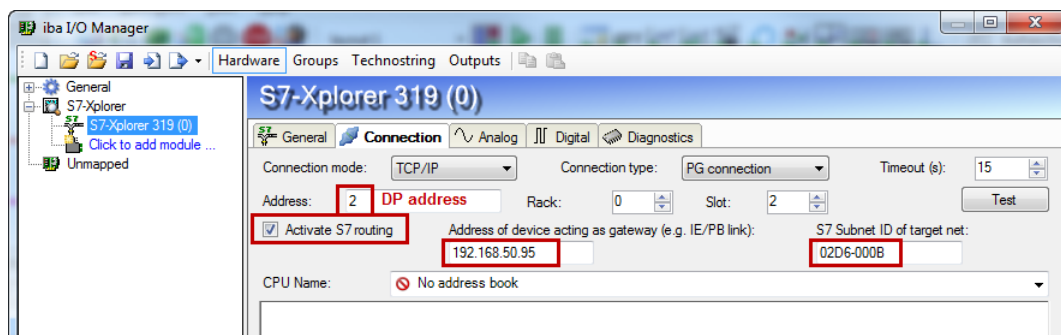
Figure 52:   Activate S7 routing

❑ Activate S7 routing
Enable to use S7 routing address

❑ Address
Address of the target control (here CPU319)

❑ Address of device acting as gateway
Enter address of the gateway (here CPU412)

❑ S7 subnet ID of target net
Enter subnet ID from STEP 7 NetPro

You can identify the S7 subnet ID in NetPro. For doing so, right-click on the secondary bus system and open the "Properties".
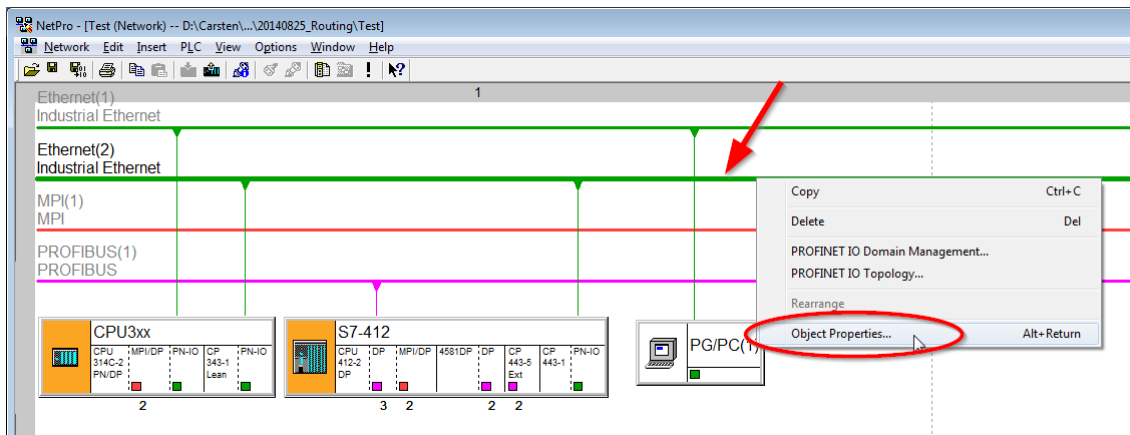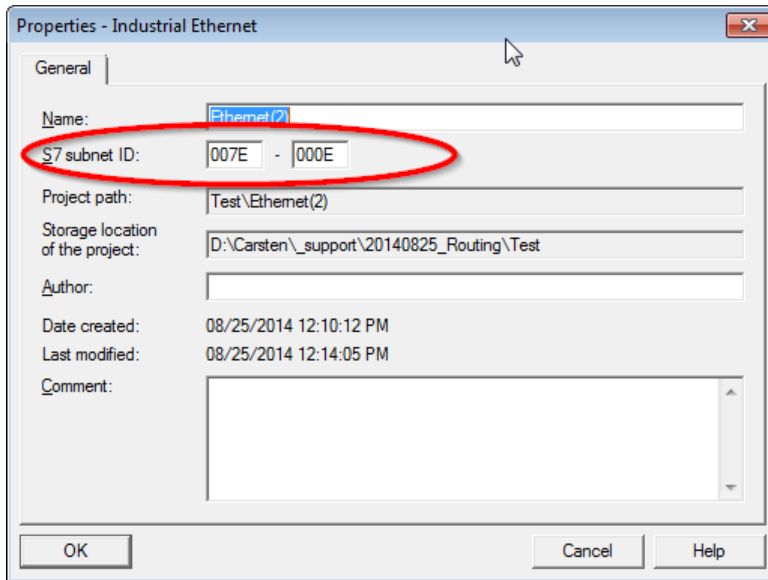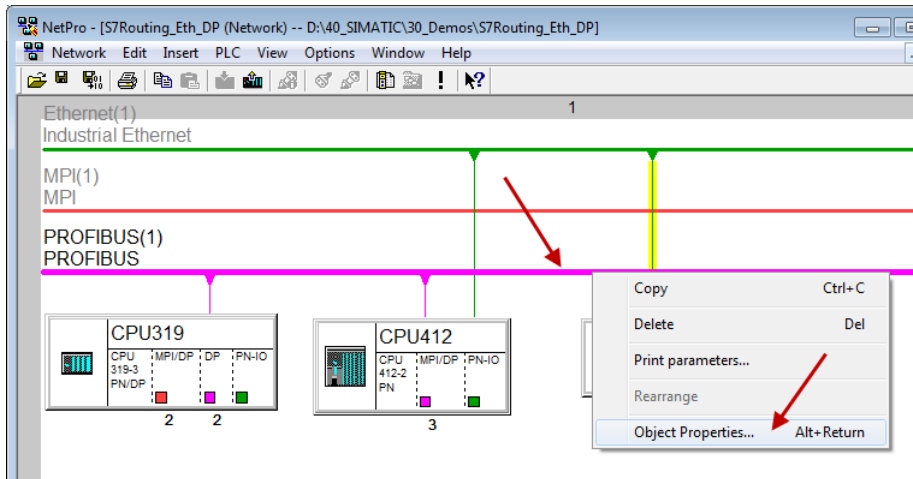

Figure 53:   Determine S7 subnet ID
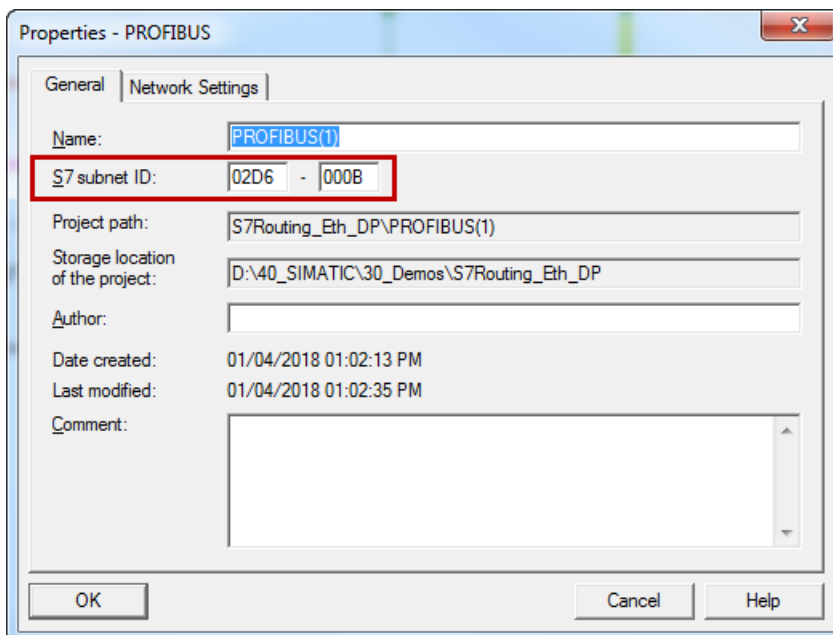
Figure 54:   S7 subnet ID

Other documentation

For more information about the S7-Routing, please see:

Which modules support the "S7 Routing" function in S7 subnets?

Web http://support.automation.siemens.com/ww/view/en/584459:

What requirements must be fulfilled and what should I watch out for when executing Routing?

Web http://support.automation.siemens.com/ww/view/en/2383206:

How do you enable cross-project S7 Routing in the TIA Portal and in STEP 7 V5.x?

Web https://support.industry.siemens.com/cs/ww/en/view/109474569:

5.7   Error codes of Request blocks

The Request blocks deliver the following possible error codes:

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
41	too many pointers (ibaREQ_DB to 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_UDPact does not match with ibaREQ_M (ID)
301	version of ibaREQ_UDPact does not match with ibaREQ_M (FB)
302	version of ibaREQ_UDPact does not match with ibaREQ_M (DB)
303	type of transmit agent does not match with configured request type in ibaPDA
305	PROFIBUS DP slave hardware configuration is invalid
306	configured peripherial 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

Value ERROR_STATUS	Description
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	errorcode of inner TUSEND / AG_SEND / AG_LSEND

Table 15:   Error codes of Request blocks

S7-1500

For the ibaREQ-UDP Request block, the following error codes may appear:

ERROR_TSEND (W#16#...)	
80A1	Connection or port already being used by user. Communication error: - 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.
80A3	The nested "T_DIAG" instruction has reported that the connection has closed.
80A4	IP address of the remote endpoint of the connection is invalid or it matches the IP address of the local partner.
80A7	Communication error: You called the instruction with COM_RST = 1 before the send job was complete.
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.
80B6	Parameter assignment error in the connection_type parameter of the data block for connection description.
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.
8085	The LEN parameter is larger than the highest permitted value.
8086	The ID parameter within the CONNECT parameter is outside the permitted range.
8087	Maximum number of connections reached; no additional connection possible.
8088	The value at the LEN parameter does not correspond to the receive area set at the DATA parameter.
8091	Maximum nesting depth exceeded.
809A	The CONNECT parameter points to a field that does not correspond to the length of the connection description.

809B	InterfaceID is invalid. It is either zero or it does not point to a local CPU interface or a CP.
80C3	All connection resources are in use. A block with this ID is already being processed in a different priority group.
80C4	Temporary communication error: - The connection cannot be established at this time. - The interface is receiving new parameters or the connection is being established. - The configured connection is currently being removed by a "TDISCON" instruction. - The connection used is being terminated by a call with COM_RST = 1.
80C6	Remote network error. Remote partner cannot be reached.

Other documentation

For more information, please see the SIEMENS documentation about the TSEND_C block.

S7-300/400

For the ibaREQ-UDP Request block, the following error codes may appear:

ERROR_TCON (W#16#...)	
8086	The ID parameter is outside the permitted range.
8087	Maximal number of connections reached; no additional connection possible
8089	The parameter CONNECT parameter does not point to a data block.
809A	The parameter CONNECT parameter points to a field that does not match the length of the connection description (UDT65).
809B	The local_device_id in the connection description does not match the CPU.
80A0	Group error for error codes W#16#80A1 and W#16#80A2
80A1	Connection or port is already occupied by user
80A2	Local or remote port is occupied by the system
80A3	Attempt being made to re-establish an existing connection
80A4	IP address of the remote connection end point is invalid, it may match the local IP address
80A7	Communications error: you have called TDISCON before TCON was complete. TDISCON must first complexly terminate the connection referenced by the ID.
80B2	The parameter CONNECT parameter points to a data block that was generated with the keyword UNLINKED.
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
80B5	Error in active_est parameter (UDT 65) in the UDP protocol variant
80B6	Parameter assignment error relating to the connection_type parameter (UDT 65)

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
80B8	Parameter ID in the local connection description (UDT 65) and parameter ID are different
80C3	All connection resources are in use.
80C4	Temporary communications error: · - The connection cannot be established at this time. · - The interface is receiving new parameters.· - The configured connection is currently being removed by a TDISCON.· - The H system is connecting and updating.

Other documentation

For more information, please see the SIEMENS documentation about the TCON block.

ERROR_TSEND (W#16#...)	
8085	LEN parameter has the value 0 or is greater than the largest permitted value
8086	The ID parameter is not in the permitted address range
8088	LEN parameter is larger than the memory area specified in DATA
8089	ADDR parameter does not point to a data block
80A1	Communication error:· - The specified connection between user program and communications layer of the operating system has not yet been established.· - The specified connection between the user program and the communication level of the operating system is currently being terminated. Transmission over this connection is not possible.· - The interface is being reinitialized.
80A4	IP address of the remote connection end point is invalid, it is possible that it matches the local IP address.
80B3	The set protocol variant (connection_type parameter in the connection description) is not UDP. Please use FB 63 "TSEND".· ADDR parameter: Invalid settings for port no.
80C3	A block with this ID is already being processed in a different priority class.· Internal lack of resources.
80C4	Temporary communications error:· - The connection between the user program and the communication level of the operating system cannot be established at this time. · - The interface is receiving new parameters.

Other documentation

For more information, please see the SIEMENS documentation about the TSEND block.

More error messages

ERROR TCON

ERROR and STATUS parameters

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Connection successfully established.
0	7000	No job processing active
0	7001	Start job execution, establish connection.
0	7002	Connection is being established (REQ irrelevant).
1	8085	Connection ID (ID parameter) is already being used by a configured connection.
1	8086	The ID parameter is outside the valid range.
1	8087	Maximum number of connections reached; no additional connection possible
1	8089	The CONNECT parameter does not point to a connection description or the connection description was created manually.
1	809A	The structure at the CONNECT parameter is not supported on an integrated interface or the length is invalid.
1	809B	The element InterfaceId within the TCON_xxx structure does not reference a hardware identifier of a CPU or CM/CP interface or has the value "0".
1	80A1	The specified connection or the port is already being used.
1	80A2	Local or remote port is being 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.
1	80A3	ID is used by a connection created by the user program, which uses the same connection description at the CONNECT parameter.
1	80A4	IP address of the remote endpoint of the connection is invalid or it corresponds to the IP address of the local partner.
1	80A7	Communication error: You executed "TDISCON" before "TCON" had completed.

1	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-".
1	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).
1	80B6	Parameter assignment error in the ConnectionType parameter of the data block for connection description. • Only valid with TCON_IP_v4: 0x11, 0x0B and 0x13. • Only valid with TCON_IP_RFC: 0x0C and 0x12
1	80B7	With TCON_IP_v4: • TCP (active connection establishment): Remote port is "0". • TCP (passive connection establishment): Local port is "0". • UDP: Local port is "0". • IP address of the partner end point was set to 0.0.0.0. With TCON_IP_RFC: • Local (LocalTSelector) or remote (RemoteTSelector) T selector was specified with a length of more than 32 bytes. • For TSelLength of the T selector (local or remote), a length greater than 32 was entered. • Error in the length of the IP address of the specific connection partner. • IP address of the partner end point was set to 0.0.0.0.
1	80B8	Parameter ID in the local connection description (structure at CONNECT parameter) and parameter ID of the instruction are different.
1	80C3	All connection resources are assigned, or ports may be dynamically used by other applications or connections.
1	80C4	Temporary communication error: • 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 currently receiving new parameters. • The configured connection is currently being removed by a "TDISCON" instruction.
1	80C5	The connection partner refuses to establish the connection, has terminated the connection or actively ended it.
1	80C6	The connection partner cannot be reached (network error).
1	80C7	Execution timeout.

1	80C8	Value at the ID parameter is already being used by a connection that was created using the user program. The connection uses the identical ID, but different connection settings at the parameter CONNECT.
1	80C9	Validation of the connection partner failed. The connection partner that wants to establish the connection does not match the defined partner of the structure at the CONNECT parameter.
1	80CE	The IP address of the local interface is 0.0.0.0.
1	80D0	In connection with TCP and the active connection end point: The remote_qdn parameter is an empty string. In this case, no connection can be established.
1	80D1	The remote_qdn parameter is not a fully qualified domain name. The period at the end may be missing.
1	80D2	No DNS server address is configured.
1	80D3	The fully qualified domain name could not be resolved. Possible causes: • 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.
1	80E0	Unsuitable or poor message was received.
1	80E1	Error during the handshake. Possible causes: • Abort by the user • Security not high enough • Renewed negotiation is not supported • SSL/TLS version is not supported • Validation of the host name failed
1	80E2	Not supported / invalid certificate 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 validity period of the certificate; the certificate is invalid for the module.
1	80E3	Certificate was discarded.
1	80E4	No valid certification authority found.
1	80E5	Certificate expired.
1	80E6	Integrity errors in the Transport Layer Security Protocol
1	80E7	Not supported extension in X.509-V3 certificate
1	80E9	TLS server without server certificate is not supported.
1	80EA	DTLS (UDP) protocol is not supported.
1	80EB	A client cannot request a client certificate.
1	80EC	The server cannot perform validation based on the subjectAlternateName (only clients can do this).
1	80ED	TLSServerCertRef_m-ID invalid

Other documentation

For more information, please see the SIEMENS documentation about the TCON block.

ERROR TUSEND

ERROR	STATUS* (W#16#...)	Explanation
0	0000	Send job completed without error
0	7000	No job processing active
0	7001	Start of job processing, data being sent Note: During this processing phase, the operating system accesses the data in the DATA send area.
0	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.
1	8085	The LEN parameter has the value "0" or is greater than the highest permitted value.
1	8086	The ID parameter is not in the permitted value range.
0	8088	The LEN parameter is greater than the memory area specified in DATA.
1	8089	The parameter ADDR does not point to a data block with the structure TADDR_Param or TADDR_SEND_QDN.
1	80A1	Communication error: • 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. Transmission over this connection is not possible. • The interface is being reinitialized.
1	80B1	You changed the DATA parameter before the current job finished.
1	80A4	IP address (at the ADDR parameter) of the remote connection end point is invalid; it may correspond to the local partner's own IP address.
1	80B3	• The protocol variant (connection_type parameter in the connection description) is not set to UDP. Please use "TSEND". • Parameter ADDR: Invalid information for port no.
1	80B7	The length of the structure referenced by the parameter ADDR is not 8 bytes.
1	80C3	• A block with this ID is already being processed in a different priority class. • Internal lack of resources.
1	80C4	Temporary communication error: • The connection between the user program and the communication layer of the operating system cannot be established at this time. • New parameter settings are being assigned to the interface. • Temporarily no receive resources available at the connection partner. The connection partner is not ready to receive.
1	80C5	• Connection terminated by the communication partner. • LSAP of the remote connection partner is not released
1	80C6	Network error: • Remote partner cannot be reached. • Physical interruption on PROFIBUS
1	80C7	Execution timeout.
1	80D3	The fully qualified domain name could not be resolved. Possible causes: • 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.
1	80EA	DTLS (UDP) protocol is not supported

Other documentation

For more information, please see the SIEMENS documentation about the TUSEND block. .

6   Support and contact

Support

Phone: +49 911 97282-14

Fax: +49 911 97282-33

Email: support@iba-ag.com

Note

If you require support, indicate the serial number (iba-S/N) of the product.

Contact

Headquarters

iba AG
Koenigswarterstr. 44
90762 Fuerth
Germany

Phone: +49 911 97282-0

Fax: +49 911 97282-33

Email: iba@iba-ag.com

Contact: Mr Harald Opel

Regional and worldwide

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

www.iba-ag.com.